Panduan Pengguna

Amazon CodeCatalyst

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon CodeCatalyst: Panduan Pengguna

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara pelanggan, atau dengan cara apa pun yang menghina atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan properti dari masing-masing pemilik, yang mungkin berafiliasi, terkait dengan, atau disponsori oleh Amazon, atau tidak.

Table of Contents

Apa itu Amazon CodeCatalyst?

Amazon CodeCatalyst adalah layanan terintegrasi untuk tim pengembangan perangkat lunak yang mengadopsi praktik integrasi dan penerapan berkelanjutan ke dalam proses pengembangan perangkat lunak mereka. CodeCatalyst menempatkan alat yang Anda butuhkan semua di satu tempat. Anda dapat merencanakan pekerjaan, berkolaborasi dalam kode, dan membangun, menguji, dan menyebarkan aplikasi dengan alat integrasi/pengiriman berkelanjutan (CI/CD) berkelanjutan. Anda juga dapat mengintegrasikan AWS sumber daya dengan proyek Anda dengan menghubungkan Anda Akun AWS ke CodeCatalyst ruang Anda. Dengan mengelola semua tahapan dan aspek siklus hidup aplikasi Anda dalam satu alat, Anda dapat mengirimkan perangkat lunak dengan cepat dan percaya diri.

Di CodeCatalyst, Anda membuat ruang untuk mewakili perusahaan, departemen, atau grup Anda, dan kemudian Anda membuat proyek yang berisi sumber daya yang diperlukan untuk mendukung tim dan tugas pengembangan Anda. CodeCatalystsumber daya terstruktur di dalam proyek yang hidup di dalam ruang. Untuk membantu tim memulai dengan cepat, CodeCatalyst sediakan cetak biru proyek berbasis bahasa atau alat. Saat Anda membuat proyek dari cetak biru proyek, proyek dilengkapi dengan sumber daya seperti repositori sumber dengan kode sampel, skrip build, tindakan penerapan, server virtual atau sumber daya tanpa server, dan banyak lagi.

Apa yang bisa saya lakukan dengan CodeCatalyst?

Anda dan tim pengembangan Anda dapat menggunakan CodeCatalyst untuk melaksanakan setiap aspek pengembangan perangkat lunak, mulai dari merencanakan pekerjaan Anda hingga menyebarkan aplikasi Anda. Anda dapat menggunakan CodeCatalyst untuk:

- Iterasi dan berkolaborasi pada kode — Bekerja secara kolaboratif dengan tim Anda pada kode dengan cabang, penggabungan, permintaan tarik, dan komentar di repositori kode sumber Anda. Buat Lingkungan Pengembang untuk bekerja pada kode dengan cepat tanpa harus mengkloning atau mengatur koneksi ke repositori.
- Membangun, menguji, dan menerapkan aplikasi Anda dengan alur kerja — Konfigurasikan alur kerja dengan tindakan build, test, dan deploy untuk menangani integrasi dan pengiriman aplikasi yang berkelanjutan. Anda dapat memulai alur kerja secara manual atau mengonfigurasinya untuk memulai secara otomatis berdasarkan peristiwa seperti mendorong kode atau membuat atau menutup permintaan tarik.

- Prioritaskan pekerjaan tim Anda dengan pelacakan masalah — Gunakan masalah untuk membuat backlog dan memantau status tugas yang sedang berlangsung dengan papan. Membuat dan memelihara tumpukan item yang sehat untuk dikerjakan tim Anda adalah bagian penting dari pengembangan perangkat lunak.
- Siapkan pemantauan dan pemberitahuan — Pantau aktivitas tim dan status sumber daya, dan konfigurasikan notifikasi agar tetap up to date dengan perubahan penting.

Bagaimana saya memulai CodeCatalyst?

Jika Anda tidak memiliki ruang atau ingin mempelajari cara mengatur dan mengelola ruang, kami sarankan Anda memulai dengan Panduan CodeCatalyst Administrator Amazon.

Jika Anda baru bekerja di proyek atau ruang, kami sarankan Anda memulai dengan:

- Meninjau CodeCatalyst konsep
- Menciptakan ruang
- Membuat proyek pertama Anda dengan mengikuti langkah-langkah di Tutorial: Membuat proyek dengan cetak biru aplikasi web tiga tingkat Modern

Pelajari lebih lanjut tentang CodeCatalyst

Anda dapat mempelajari lebih lanjut tentang fungsionalitas CodeCatalyst dalam panduan pengguna ini, serta sumber daya berikut:

- AWS DevOps Blog artikel tentang Amazon CodeCatalyst
- Panduan Referensi CodeCatalyst API Amazon
- Panduan Pengembang Kit Pengembangan CodeCatalyst Aksi Amazon
- CodeCatalyst FAQ
- Testimonial

CodeCatalyst konsep

Kenali konsep-konsep kunci untuk membantu mempercepat kolaborasi dan pengembangan aplikasi Anda di Amazon CodeCatalyst. Konsep-konsep ini mencakup istilah yang digunakan dalam kontrol sumber, integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD), dan pemodelan dan konfigurasi proses rilis otomatis.

Untuk informasi konseptual tambahan, lihat topik berikut:

- Konsep repositori sumber
- Konsep alur kerja

Topik

- AWS Ruang ID Builder di CodeCatalyst
- Ruang yang mendukung federasi identitas di CodeCatalyst
- Proyek
- Cetak Biru
- Koneksi akun
- Koneksi VPC
- AWS ID Pembangun
- Profil pengguna di CodeCatalyst
- Repositori sumber
- Berkomitmen
- Lingkungan Dev
- Alur Kerja
- Tindakan
- Masalah
- Token akses pribadi (PATs)
- Koneksi pribadi
- Peran

AWS Ruang ID Builder di CodeCatalyst

Administrator ruang mengundang pengguna CodeCatalyst dengan mengirim email undangan individual dari halaman anggota. Pengguna yang diundang atau mendaftar untuk CodeCatalyst membuat AWS Builder ID mereka sendiri. Profil dikelola di AWS Builder ID dan ditampilkan sebagai nama pengguna dan informasi profil di pengaturan pengguna di CodeCatalyst.

Ruang yang mendukung federasi identitas di CodeCatalyst

Pengguna yang telah ditambahkan ke pengguna SSO dan grup untuk instans IAM Identity Center dan dikelola di toko identitas dan diundang ke ruang Anda melalui IAM Identity Center. Administrator Space menyinkronkan halaman CodeCatalyst anggota untuk pembaruan terbaru. Pengguna masuk menggunakan portal masuk SSO seperti yang diatur dalam instans Pusat Identitas IAM perusahaan. Spasi yang mendukung federasi identitas terhubung ke instance penyimpanan identitas melalui aplikasi Pusat Identitas dan pemetaannya ke ID penyimpanan identitas.

Proyek

Sebuah proyek merupakan upaya kolaboratif CodeCatalyst yang mendukung tim dan tugas pengembangan. Setelah memiliki proyek, Anda dapat menambahkan, memperbarui, atau menghapus pengguna dan sumber daya, menyesuaikan dasbor proyek Anda, dan memantau kemajuan pekerjaan tim Anda. Anda dapat memiliki beberapa proyek dalam satu ruang.

Untuk informasi lebih lanjut tentang proyek, lihatMengatur pekerjaan dengan proyek-proyek di CodeCatalyst.

Cetak Biru

Blueprint adalah synthesizer proyek yang menghasilkan dan memperluas file dukungan aplikasi dan dependensi untuk Anda, bersama dengan membuat proyek Anda di konsol. CodeCatalyst Anda memilih jenis proyek dari pilihan cetak biru, melihat file README CodeCatalyst, dan melihat pratinjau repositori proyek dan sumber daya yang akan dihasilkan. Proyek Anda dihasilkan dari konfigurasi dasar yang ditentukan oleh cetak biru. Anda mensintesis ke cetak biru proyek secara berkala, yang memperbarui file proyek Anda, seperti dependensi perangkat lunak, dan meregenerasi sumber daya. Proyek menggunakan alat yang disebut Projen untuk mensintesis proyek dengan menyinkronkan pembaruan proyek terbaru dan menghasilkan file dukungan. File-file ini dapat mencakup `package.json` `Makefileeslint`,,, dan lainnya berdasarkan jenis dan bahasa aplikasi

Anda. Cetak biru proyek dapat menghasilkan file yang mendukung AWS sumber daya seperti konstruksi CDK, templat, AWS CloudFormation dan templat. AWS Serverless Application Model

Untuk informasi selengkapnya tentang cetak biru proyek, lihat. Membuat proyek yang komprehensif dengan CodeCatalyst cetak biru

Koneksi akun

Koneksi akun mengaitkan CodeCatalyst ruang dengan Anda Akun AWS. Setelah koneksi akun Anda diatur, koneksi Akun AWS dibuat tersedia untuk ruang. Anda kemudian dapat menambahkan peran IAM CodeCatalyst sehingga dapat mengakses sumber daya di Anda Akun AWS. Anda juga dapat menggunakan peran ini untuk tindakan CodeCatalyst alur kerja Anda.

Untuk informasi selengkapnya tentang koneksi akun, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS.

Koneksi VPC

Koneksi VPC adalah CodeCatalyst sumber daya yang berisi semua konfigurasi yang diperlukan untuk alur kerja Anda untuk mengakses VPC. Administrator ruang angkasa dapat menambahkan koneksi VPC mereka sendiri di konsol CodeCatalyst Amazon atas nama anggota luar angkasa. Dengan menambahkan koneksi VPC, anggota ruang dapat menjalankan tindakan alur kerja dan membuat Lingkungan Pengembang yang mematuhi aturan jaringan dan dapat mengakses sumber daya di VPC terkait.

Untuk informasi selengkapnya tentang koneksi VPC, lihat Mengelola Awan Pribadi Virtual Amazon di Panduan CodeCatalyst Administrator.

AWS ID Pembangun

AWS Builder ID adalah identitas pribadi yang dapat Anda gunakan untuk mendaftar dan masuk CodeCatalyst dan aplikasi lain yang berpartisipasi. Ini tidak sama dengan Akun AWS. AWS Builder ID Anda mengelola metadata seperti alias pengguna dan alamat email. AWS Builder ID Anda adalah identitas unik yang mendukung pengguna di semua ruang CodeCatalyst. Untuk informasi tentang mengakses profil AWS Builder ID Anda, lihatMemperbarui profil Anda. Untuk mempelajari lebih lanjut tentang AWS Builder ID, lihat AWS Builder ID di Referensi Umum AWS.

Untuk informasi selengkapnya tentang mendaftar dan masuk, lihatSiapkan dan masuk ke CodeCatalyst.

Profil pengguna di CodeCatalyst

Anda mengakses profil CodeCatalyst pengguna Anda dengan memilih opsi profil dari drop-down di bawah inisiasi login Anda di halaman mana pun di. CodeCatalyst Anda dapat membuat token akses pribadi (PAT) dari halaman profil Anda, tetapi Anda hanya dapat melihat atau menghapus PAT menggunakan. AWS CLI Nama pengguna Anda adalah alias yang Anda pilih saat mendaftar. Anda tidak dapat mengubah nama pengguna Anda. Untuk melihat halaman profil untuk CodeCatalyst pengguna lain, buka tab Anggota untuk proyek Anda dan pilih pengguna yang sesuai.

Anda mengakses AWS Builder ID Anda dengan melihat CodeCatalyst profil Anda dan kemudian memilih untuk pergi ke AWS Builder ID. Anda akan diarahkan ke halaman profil AWS Builder ID Anda. Nama lengkap, alamat email, dan kata sandi profil Anda dikelola oleh AWS Builder ID Anda, dan Anda dapat mengedit informasi tersebut menggunakan halaman AWS Builder ID. Anda memasukkan informasi ini saat mendaftar. Ketika Anda siap untuk mengatur MFA untuk menggunakan aplikasi autentikator untuk masuk, Anda akan menggunakan halaman AWS Builder ID. Untuk informasi selengkapnya tentang melihat profil AWS Builder ID Anda, lihatMemperbarui profil Anda.

Untuk informasi selengkapnya tentang mendaftar dan masuk, lihatSiapkan dan masuk ke CodeCatalyst.

Repositori sumber

Repositori sumber adalah tempat Anda menyimpan kode dan file dengan aman untuk proyek Anda. Ini juga menyimpan riwayat versi file Anda. Secara default, repositori sumber dibagikan dengan pengguna lain dalam proyek Anda CodeCatalyst . Anda dapat memiliki lebih dari satu repositori sumber untuk sebuah proyek. Anda dapat membuat repositori sumber untuk proyek di CodeCatalyst, atau Anda dapat memilih untuk menautkan repositori sumber yang ada yang dihosting oleh layanan lain jika layanan tersebut didukung oleh ekstensi yang diinstal. Misalnya, Anda dapat menautkan GitHub repositori ke proyek setelah Anda menginstal ekstensi GitHub Repositori. Untuk informasi selengkapnya, lihat Menyimpan kode sumber di repositori untuk proyek di CodeCatalyst dan Mulai cepat: Menginstal ekstensi, menghubungkan penyedia, dan menautkan sumber daya di CodeCatalyst.

Repositori sumber juga merupakan tempat informasi konfigurasi disimpan untuk CodeCatalyst proyek Anda, seperti file konfigurasi yang mendefinisikan atribut dan tindakan alur kerja CI/CD Anda. Jika Anda membuat proyek menggunakan cetak biru, repositori sumber akan dibuat dengan informasi konfigurasi proyek yang tersimpan di dalamnya. Jika Anda membuat proyek kosong, Anda harus

membuat repositori sumber sebelum Anda dapat membuat sumber daya yang memerlukan informasi konfigurasi, seperti alur kerja.

Untuk konsep lainnya yang dapat membantu Anda bekerja dengan repositori sumber dan kontrol sumber, lihat. Konsep repositori sumber

Berkomitmen

Komit adalah perubahan ke file atau kumpulan file. Di CodeCatalyst konsol Amazon, komit menyimpan perubahan Anda dan mendorongnya ke repositori sumber. Komit mencakup informasi tentang perubahan, termasuk identitas pengguna yang melakukan perubahan, waktu dan tanggal perubahan, judul komit, dan pesan apa pun yang disertakan tentang perubahan tersebut. Untuk informasi selengkapnya, lihat Memahami perubahan kode sumber dengan komit di Amazon CodeCatalyst.

Dalam konteks repositori sumber di CodeCatalyst, komit adalah snapshot dari perubahan pada konten repositori Anda. Setiap kali pengguna melakukan dan mendorong perubahan, CodeCatalyst menyimpan informasi yang mencakup siapa yang melakukan perubahan, tanggal dan waktu komit, dan perubahan yang dibuat sebagai bagian dari komit. Anda juga dapat menambahkan tag Git ke commit untuk membantu mengidentifikasi commit tertentu.

Untuk informasi selengkapnya tentang komit, lihatMemahami perubahan kode sumber dengan komit di Amazon CodeCatalyst.

Lingkungan Dev

Lingkungan Dev adalah lingkungan pengembangan berbasis cloud yang dapat Anda gunakan CodeCatalyst untuk bekerja dengan cepat pada kode yang disimpan di repositori sumber proyek Anda. Alat proyek dan pustaka aplikasi yang disertakan dalam Lingkungan Dev Anda ditentukan oleh devfile di repositori sumber proyek Anda. Jika Anda tidak memiliki devfile di repositori sumber Anda, devfile default akan diterapkan secara otomatis. Devfile default mencakup alat untuk bahasa dan kerangka kerja pemrograman yang paling sering digunakan. Secara default, Dev Environment dikonfigurasi untuk memiliki prosesor 2-core, RAM 4 GB, dan penyimpanan persisten 16 GiB.

Alur Kerja

Alur kerja adalah prosedur otomatis yang menjelaskan cara membangun, menguji, dan menyebarkan kode Anda sebagai bagian dari sistem integrasi berkelanjutan dan pengiriman berkelanjutan (CI/

CD). Alur kerja mendefinisikan serangkaian langkah, atau tindakan, yang harus diambil selama menjalankan alur kerja. Alur kerja juga mendefinisikan peristiwa, atau pemicu, yang menyebabkan alur kerja dimulai. Untuk menyiapkan alur kerja, Anda membuat file definisi alur kerja menggunakan editor visual atau YAMM CodeCatalyst konsol.

> **Tip**
>
> Untuk melihat sekilas bagaimana Anda dapat menggunakan alur kerja dalam sebuah proyek, buat proyek dengan cetak biru. Setiap cetak biru menerapkan alur kerja yang berfungsi yang dapat Anda tinjau, jalankan, dan bereksperimen.

Untuk informasi lebih lanjut tentang alur kerja, lihat Bangun, uji, dan terapkan dengan alur kerja di CodeCatalyst.

Tindakan

Tindakan adalah blok bangunan utama alur kerja, dan mendefinisikan unit logis kerja, atau tugas, untuk dilakukan selama alur kerja dijalankan. Biasanya, alur kerja mencakup beberapa tindakan yang berjalan secara berurutan atau paralel tergantung pada cara Anda mengonfigurasinya.

Untuk informasi selengkapnya tentang tindakan, lihatMengkonfigurasi tindakan yang dilakukan alur kerja.

Masalah

Masalah adalah catatan yang melacak pekerjaan yang terkait dengan proyek Anda. Anda dapat membuat masalah untuk fitur, tugas, bug, atau badan pekerjaan lain yang terkait dengan proyek Anda. Jika Anda menggunakan pengembangan tangkas, masalah juga dapat menggambarkan kisah epik atau pengguna.

Untuk informasi selengkapnya tentang masalah, lihatLacak dan atur pekerjaan dengan masalah di CodeCatalyst.

Token akses pribadi (PATs)

Token akses pribadi (PAT) mirip dengan kata sandi. Ini terkait dengan identitas pengguna Anda untuk digunakan di semua ruang dan proyek di CodeCatalyst. Anda menggunakan PAT untuk mengakses

CodeCatalyst sumber daya yang mencakup lingkungan pengembangan terintegrasi (IDE) dan repositori sumber berbasis Git. PAT mewakili Anda CodeCatalyst dan Anda dapat mengelolanya di pengaturan pengguna Anda. Seorang pengguna dapat memiliki lebih dari satu PAT. Token akses pribadi hanya ditampilkan sekali. Sebagai praktik terbaik, pastikan untuk menyimpannya dengan aman di komputer lokal Anda. Secara default, PAT kedaluwarsa setelah satu tahun.

Untuk informasi lebih lanjut tentang PAT, lihatBerikan akses repositori pengguna dengan token akses pribadi.

Koneksi pribadi

Koneksi pribadi adalah otorisasi antara CodeCatalyst identitas Anda dan penyedia sumber eksternal Anda, seperti GitHub. Anda menggunakan koneksi pribadi untuk memungkinkan CodeCatalyst pengguna menambahkan repositori sumber pihak ketiga. Misalnya, Anda dapat menghubungkan GitHub repositori ke spasi. CodeCatalyst Aplikasi konektor yang diinstal diinstal di GitHub akun untuk digunakan dengan repositori yang ditunjuk oleh pemilik akun. Anda dapat membuat satu koneksi pribadi untuk satu identitas pengguna (CodeCatalyst alias) di semua ruang untuk jenis penyedia tertentu, seperti GitHub. Koneksi pribadi terkait dengan AWS Builder ID atau pengguna SSO Anda.

Untuk informasi selengkapnya, lihat Mengakses GitHub sumber daya dengan koneksi pribadi.

Peran

Peran mendefinisikan akses pengguna ke sumber daya untuk proyek atau ruang dan tindakan mana yang dapat dilakukan pengguna. Anda memilih peran untuk pengguna ketika Anda mengundang mereka ke proyek. Ada peran tingkat ruang dan peran tingkat proyek di. CodeCatalyst Seorang pengguna dengan peran administratif pada tingkat yang benar dapat mengubah peran yang ditetapkan. Misalnya, pengguna dengan peran administrator Project untuk proyek memiliki kontrol penuh atas proyek tersebut dan dapat mengubah peran pengguna dalam proyek tersebut. Untuk informasi tentang peran mana yang tersedia dan izin yang dimiliki setiap peran, lihatMemberikan akses dengan peran pengguna.

Untuk informasi lebih lanjut tentang peran, lihat Memberikan akses dengan peran pengguna.

Siapkan dan masuk ke CodeCatalyst

Ada dua jenis ruang yang dapat Anda atur CodeCatalyst: spasi yang mendukung pengguna AWS Builder ID, dan membuat ruang yang mendukung federasi identitas, tempat pengguna dan grup SSO dikelola di IAM Identity Center. Pengguna di ruang ID AWS Pembangun masuk CodeCatalyst dengan ID AWS Pembangun mereka, dan pengguna di ruang perusahaan masuk CodeCatalyst menggunakan portal SSO untuk perusahaan yang terkait dengan ruang tersebut.

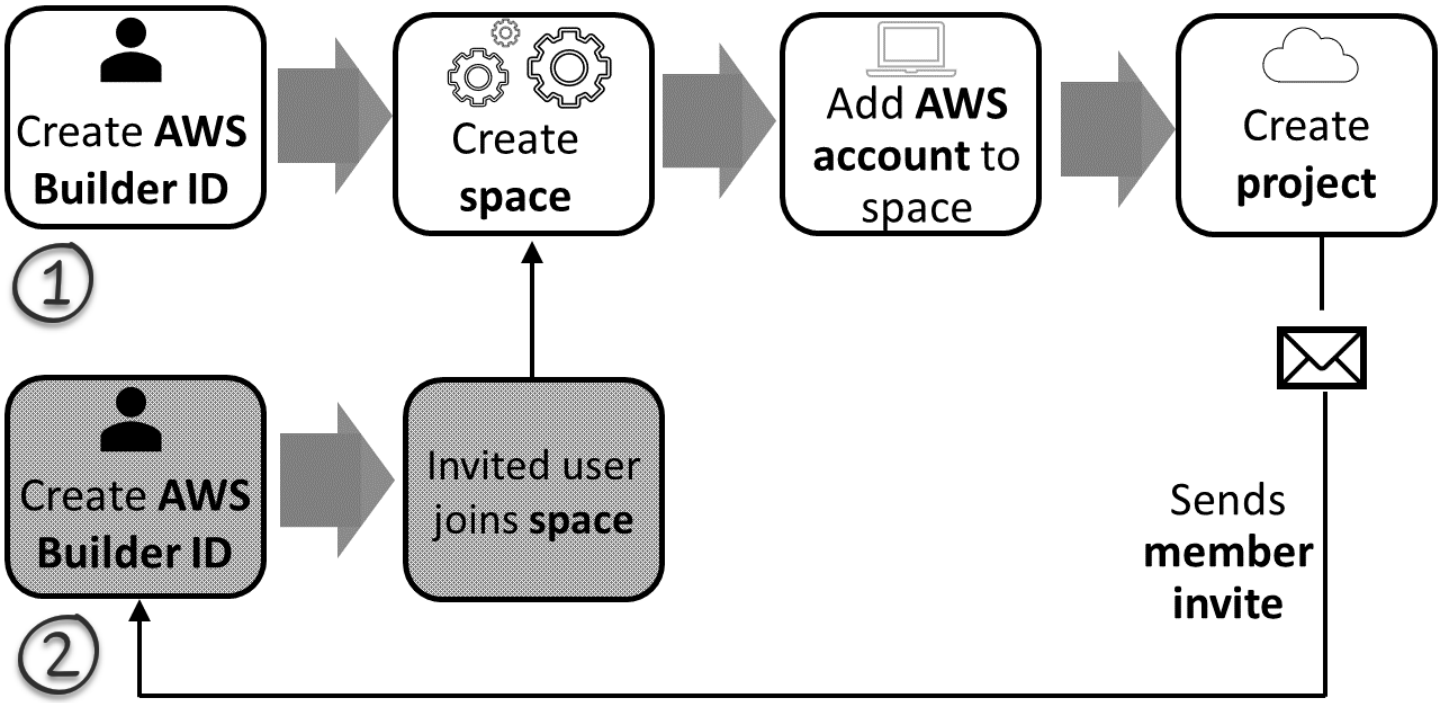
Langkah-langkah untuk mengatur dan mengelola ruang AWS Builder ID disediakan dalam panduan ini. Untuk bekerja dengan ruang CodeCatalyst AWS Builder ID, Anda akan mengatur CodeCatalyst menggunakan pengaturan pengguna dan AWS Builder ID yang Anda gunakan untuk masuk CodeCatalyst.

Langkah-langkah untuk mengatur dan mengelola ruang yang mendukung federasi identitas disediakan dalam Panduan CodeCatalyst Administrator. Untuk bekerja dengan spasi yang disiapkan untuk federasi identitas, lihat Pengaturan dan administrasi untuk CodeCatalyst spasi di Panduan CodeCatalyst Administrator Amazon.

Bagian ini menyediakan dua jalur umum untuk pengaturan agar bekerja di Amazon CodeCatalyst dengan ruang AWS Builder ID: membuat spasi dan proyek sebagai pengguna pertama, dan menerima undangan ke ruang atau proyek yang ada. Alur kerja pengaturan ini tentu sangat berbeda. Diagram berikut menunjukkan kedua proses pendaftaran sebagai berikut:

Ada dua jalur umum untuk menyiapkan untuk bekerja di Amazon CodeCatalyst: membuat ruang dan proyek sebagai pengguna pertama, dan menerima undangan ke ruang atau proyek yang ada. Alur kerja pengaturan ini tentu sangat berbeda. Diagram berikut menunjukkan kedua proses pendaftaran sebagai berikut:

1. Dalam kasus pertama, Anda membuat dan menyiapkan ruang untuk perusahaan, tim, atau grup Anda, dan membuat proyek sebelum mengundang orang lain ke sumber daya ini. An Akun AWS harus disediakan untuk tujuan penagihan, di mana Anda masih dapat default ke tingkat Gratis.
2. Dalam kasus kedua, jika Anda bergabung CodeCatalyst dengan menerima undangan ke suatu proyek, orang lain telah membuat ruang dan proyek untuk Anda. Namun, Anda masih ingin mengonfigurasi profil Anda sehingga Anda siap untuk mulai bekerja dengan orang lain.



> **Tip**
>
> CodeCatalyst menggunakan spasi untuk mengelompokkan proyek dan sumber daya. Saat pertama kali mendaftar CodeCatalyst, Anda akan diminta untuk membuat ruang serta proyek.

Apakah Anda mendaftar untuk membuat spasi dan proyek atau Anda mendaftar sebagai bagian dari menerima undangan, Anda membuat ID AWS Pembangun yang akan Anda gunakan untuk CodeCatalyst masuk. Untuk membuat AWS Builder ID, Anda memberikan nama lengkap, kata sandi, dan alamat email yang Anda gunakan untuk masuk ke AWS aplikasi. Anda menggunakan email dan kata sandi untuk masuk CodeCatalyst setelah titik ini. Anda juga dapat menggunakan AWS Builder ID ini untuk masuk ke aplikasi lain yang menggunakan kredensi AWS Builder ID.

Di dalam CodeCatalyst dan di AWS Builder ID, profil dibuat berdasarkan informasi login Anda. Profil Anda berisi CodeCatalyst preferensi Anda untuk pengaturan bahasa dan pemberitahuan di CodeCatalyst proyek Anda.

> **Tip**
>
> Jika Anda mengalami masalah saat mendaftar ke CodeCatalyst profil Amazon Anda, ikuti langkah-langkah yang disediakan di halaman itu. Jika Anda membutuhkan bantuan tambahan, lihatMasalah mendaftar.

Topik

Membuat peran ruang dan pengembangan pertama Anda (dimulai tanpa undangan)

Anda dapat mendaftar ke Amazon CodeCatalyst tanpa undangan ke ruang atau proyek yang ada. Ketika Anda melakukannya, Anda akan membuat spasi dan proyek setelah membuat AWS Builder ID Anda. Sebagai bagian dari menciptakan ruang, Anda perlu menambahkan untuk tujuan penagihan. Akun AWS

> **Tip**
>
> Jika Anda mengalami masalah saat mendaftar ke CodeCatalyst profil Amazon Anda, ikuti langkah-langkah yang disediakan di halaman itu. Jika Anda membutuhkan bantuan tambahan, lihatMasalah mendaftar.

Berikut adalah salah satu aliran yang mungkin untuk pengguna yang memulai CodeCatalyst tanpa undangan ke proyek atau spasi.

Mary Major adalah pengembang yang tertarik CodeCatalyst dan memutuskan untuk mencobanya. Dia menavigasi ke CodeCatalyst konsol dan memilih opsi untuk mendaftar dan membuat ID AWS Pembangun. Mary memberikan alamat email dan kata sandi untuk membuat ID AWS Pembangunnya. Dia akan dapat menggunakan AWS Builder ID untuk masuk CodeCatalyst dan aplikasi lainnya. Ketika diminta untuk memilih alias, dia menentukan `MaryMajor` sebagai nama CodeCatalyst pengguna yang akan ditampilkan CodeCatalyst dan anggota proyek lainnya akan digunakan untuk @mention Mary.

Selanjutnya, Mary secara otomatis diarahkan untuk membuat ruang. Sebagai bagian dari aliran ini, Mary diminta untuk mengasosiasikan Akun AWS dengan ruang yang dia buat sehingga dia dapat melihat kode sampel dalam pembuatan dan penerapan proyek pertamanya. Dia menambahkan informasi itu dan menciptakan ruangnya, di mana dia memilih opsi untuk membuat peran pengembangan pratinjau yang dapat digunakan untuk proyek di ruang barunya. Mary memilih untuk membuat proyek, dan kemudian dia melihat daftar cetak biru untuk proyek. Setelah meninjau informasi untuk cetak biru yang tersedia, ia memutuskan untuk mencoba cetak biru aplikasi web tiga tingkat Modern untuk proyek pertamanya. Dia mengisi bidang yang diperlukan dan membuat proyek. Segera setelah proyek siap, dia dibawa ke halaman ringkasan proyek yang mencakup aktivitas terbaru serta tautan ke kode proyek dan alur kerja yang secara otomatis membangun dan menyebarkan kode itu. Dia mengeksplorasi kode dan alur kerja, termasuk melihat contoh aplikasi web yang digunakan. Menyukai apa yang dilihatnya, dia memutuskan untuk mengundang beberapa rekan kerjanya ke proyek untuk mulai menjelajah. CodeCatalyst

Ketika dia memiliki momen, Mary mengonfigurasi ID AWS Pembangunnya untuk masuk CodeCatalyst dengan otentikasi multi-faktor (MFA). Dengan MFA yang dikonfigurasi, Mary dapat masuk CodeCatalyst menggunakan kombinasi CodeCatalyst kata sandinya dan kode sandi atau token dari aplikasi otentikasi pihak ketiga yang disetujui.

Membuat ruang pertama Anda dan peran IAM

Ikuti langkah-langkah berikut untuk mendaftar ke CodeCatalyst profil Amazon Anda, membuat spasi, dan menambahkan akun, peran dukungan, dan peran pengembang untuk ruang Anda.

Prosedur akhir membuat dan menambahkan peran pengembang. Peran pengembang adalah peran AWS IAM yang memungkinkan CodeCatalyst alur kerja Anda mengakses AWS sumber daya. Peran pengembang adalah peran layanan yang digunakan untuk mengelola Layanan AWS dan akan dibuat di akun yang masuk. Peran layanan adalah peran IAM yang diambil oleh sebuah layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, mengubah, dan menghapus peran layanan dari dalam

IAM. Peran akan memiliki nama`CodeCatalystWorkflowDevelopmentRole-`*`spaceName`*. Untuk informasi selengkapnya tentang kebijakan peran dan peran, lihatMemahami peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan.

> **Note**
>
> Sebagai praktik terbaik keamanan, hanya menetapkan akses administratif ke pengguna administratif dan pengembang yang perlu mengelola akses ke AWS sumber daya di ruang tersebut.

Sebelum Anda mulai, Anda harus siap memberikan Akun AWS ID untuk akun di mana Anda memiliki hak administratif. Siapkan Akun AWS ID 12 digit Anda. Untuk informasi tentang menemukan Akun AWS ID Anda, lihat Akun AWS ID Anda dan aliasnya.

Untuk mendaftar sebagai pengguna baru

1. Sebelum Anda memulai di CodeCatalyst konsol, buka AWS Management Console, dan kemudian pastikan Anda masuk dengan yang sama Akun AWS yang ingin Anda gunakan untuk membuat ruang Anda.
2. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
3. Pada halaman selamat datang, pilih Daftar. Halaman Create your AWS Builder ID ditampilkan. AWS Builder ID Anda adalah identitas yang Anda buat untuk masuk. Ini tidak sama dengan Akun AWS.
4. Di alamat email Anda, masukkan alamat email yang ingin Anda kaitkan CodeCatalyst. Lalu pilih Selanjutnya.
5. Dalam nama Anda, berikan nama depan dan belakang yang ingin ditampilkan dalam aplikasi tempat Anda menggunakan AWS Builder ID. Spasi diperbolehkan. Ini akan menjadi nama profil AWS Builder ID Anda, seperti Mary Major. Anda dapat mengubah nama nanti.

   Pilih Selanjutnya. Halaman verifikasi Email ditampilkan.
6. Kode verifikasi akan dikirim ke email yang Anda tentukan. Masukkan kode ini dalam kode Verifikasi, lalu pilih Verifikasi. Jika Anda tidak menerima kode setelah 5 menit dan tidak dapat menemukannya di folder spam atau sampah, pilih Kirim ulang kode.
7. Setelah kami memverifikasi kode Anda, masukkan kata sandi yang memenuhi persyaratan di Kata Sandi dan Konfirmasi kata sandi.

Pilih kotak centang yang mengonfirmasi perjanjian Anda dengan Perjanjian AWS Pelanggan dan Ketentuan AWS Layanan, lalu pilih Buat ID AWS Pembangun.

8. Pada halaman Buat CodeCatalyst alias Anda, masukkan alias yang ingin Anda gunakan untuk pengenal pengguna unik Anda. CodeCatalyst Pilih versi singkat dari nama Anda tanpa spasi, seperti MaryMajor. CodeCatalyst Pengguna lain akan menggunakan ini untuk @mention Anda di komentar dan permintaan tarik. CodeCatalyst Profil Anda akan berisi nama lengkap Anda dari AWS Builder ID dan CodeCatalyst alias Anda. Anda tidak dapat mengubah CodeCatalyst alias Anda nanti.

   Nama lengkap Anda dan alias Anda akan ditampilkan di berbagai area di CodeCatalyst. Misalnya, nama profil Anda ditampilkan untuk aktivitas yang terdaftar di feed aktivitas, tetapi anggota proyek akan menggunakan alias Anda ke @mention Anda.

   Pilih Selanjutnya. Halaman diperbarui untuk menampilkan bagian Buat CodeCatalyst ruang Anda.

9. Dalam Nama ruang Anda, masukkan nama ruang Anda. Anda tidak dapat mengubah ini nanti.

   > **Note**
   >
   > Nama ruang harus unik di seberang CodeCatalyst. Anda tidak dapat menggunakan kembali nama spasi yang dihapus.

10. Di menu Wilayah AWStarik-turun, pilih wilayah tempat Anda ingin menyimpan ruang dan data proyek Anda. Anda tidak dapat mengubah ini nanti.
11. Pilih Selanjutnya. Halaman diperbarui untuk menampilkan halaman untuk menambahkan file Akun AWS. Akun ini akan digunakan sebagai akun penagihan untuk ruang tersebut.
12. Di Akun AWS ID, masukkan ID dua belas digit untuk akun yang ingin Anda sambungkan ke ruang Anda.

    Dalam token verifikasi AWS akun, salin ID token yang dihasilkan. Token secara otomatis disalin untuk Anda, tetapi Anda mungkin ingin menyimpannya saat Anda menyetujui permintaan AWS koneksi.

13. Pilih Buka AWS konsol untuk memverifikasi.
14. Halaman CodeCatalyst spasi Verifikasi Amazon terbuka di AWS Management Console. Ini adalah halaman CodeCatalyst spasi Amazon. Anda mungkin perlu masuk untuk mengakses halaman.

Di dalam AWS Management Console, pastikan untuk memilih yang sama Wilayah AWS di mana Anda ingin membuat ruang Anda.

Untuk mengakses halaman secara langsung, masuk ke Amazon CodeCatalyst Spaces AWS Management Console di https://console.aws.amazon.com/codecatalyst/home/.

Bidang token verifikasi di dalam AWS Management Console secara otomatis diisi dengan token yang dihasilkan di CodeCatalyst.

15. (Opsional) Di bawah tingkatan berbayar resmi, pilih Otorisasi tingkatan berbayar (Standar, Perusahaan) untuk mengaktifkan tingkatan berbayar untuk akun penagihan Anda.

> **Note**
>
> Ini tidak meningkatkan tingkat penagihan ke tingkat berbayar. Namun, ini mengonfigurasi Akun AWS sehingga Anda dapat mengubah tingkat penagihan untuk ruang Anda kapan saja. CodeCatalyst Anda dapat mengaktifkan tingkatan berbayar kapan saja. Tanpa membuat perubahan ini, ruang hanya dapat menggunakan tingkat Gratis.

16. Pilih Verifikasi ruang.

    Pesan sukses terverifikasi Akun ditampilkan untuk menunjukkan bahwa akun telah ditambahkan ke ruang.

17. Tetap di halaman Verifikasi Amazon CodeCatalyst spasi. Pilih tautan berikut: Untuk menambahkan peran IAM untuk ruang ini, lihat detail spasi.

    Halaman koneksi dengan detail CodeCatalyst spasi terbuka di AWS Management Console. Ini adalah halaman CodeCatalyst spasi Amazon. Anda mungkin perlu masuk untuk mengakses halaman.

18. Kembali ke CodeCatalyst halaman, lalu pilih Berikutnya.

19. Pesan status ditampilkan saat ruang Anda sedang dibuat. Saat ruang dibuat, pesan CodeCatalyst berikut ditampilkan: Ruang Anda sudah siap. Langkah terakhir Anda adalah membuat proyek. . Anda dapat melakukan salah satu hal berikut:

    - Pilih Lewati untuk saat ini.
    - Pilih Buat proyek pertama Anda untuk ruang Anda. Untuk tutorial yang menunjukkan cara membuat proyek dengan cetak biru, lihat Tutorial: Membuat proyek dengan cetak biru aplikasi web tiga tingkat Modern

> **Note**
>
> Jika kesalahan izin atau spanduk ditampilkan, maka segarkan halaman dan coba lihat halaman lagi.

Untuk membuat dan menambahkan CodeCatalyst CodeCatalystWorkflowDevelopmentRole-***spaceName***

1. Sebelum Anda mulai di CodeCatalyst konsol, buka AWS Management Console, dan kemudian pastikan Anda masuk dengan yang sama Akun AWS untuk ruang Anda.
2. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
3. Arahkan ke CodeCatalyst ruang Anda. Pilih Pengaturan, lalu pilih Akun AWS.
4. Pilih tautan untuk Akun AWS tempat Anda ingin membuat peran. Halaman Akun AWS detail ditampilkan.
5. Pilih Kelola peran dari AWS Management Console.

   Peran Tambahkan IAM ke halaman CodeCatalyst ruang Amazon terbuka di. AWS Management Console Ini adalah halaman CodeCatalyst spasi Amazon. Anda mungkin perlu masuk untuk mengakses halaman.
6. Pilih Buat peran administrator CodeCatalyst pengembangan di IAM. Opsi ini membuat peran layanan yang berisi kebijakan izin dan kebijakan kepercayaan untuk peran pengembangan. Peran akan memiliki nama`CodeCatalystWorkflowDevelopmentRole-`*`spaceName`*. Untuk informasi selengkapnya tentang kebijakan peran dan peran, lihatMemahami peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan.

   > **Note**
   >
   > Peran ini hanya disarankan untuk digunakan dengan akun pengembang dan menggunakan kebijakan `AdministratorAccess` AWS terkelola, memberikan akses penuh untuk membuat kebijakan dan sumber daya baru dalam hal ini Akun AWS.
7. Pilih Buat peran pengembangan.

8. Pada halaman koneksi, di bawah peran IAM yang tersedia untuk CodeCatalyst, lihat `CodeCatalystWorkflowDevelopmentRole-spaceName` peran dalam daftar peran IAM yang ditambahkan ke akun Anda.
9. Untuk kembali ke ruang Anda, pilih Buka Amazon CodeCatalyst.

Untuk membuat dan menambahkan CodeCatalyst AWSRoleForCodeCatalystSupport

1. Sebelum Anda mulai di CodeCatalyst konsol, buka AWS Management Console, dan kemudian pastikan Anda masuk dengan yang sama Akun AWS untuk ruang Anda.
2. Arahkan ke CodeCatalyst ruang Anda. Pilih Pengaturan, lalu pilih Akun AWS.
3. Pilih tautan untuk Akun AWS tempat Anda ingin membuat peran. Halaman Akun AWS detail ditampilkan.
4. Pilih Kelola peran dari AWS Management Console.

   Peran Tambahkan IAM ke halaman CodeCatalyst ruang Amazon terbuka di. AWS Management Console Ini adalah halaman Amazon CodeCatalyst Spaces. Anda mungkin perlu masuk untuk mengakses halaman.
5. Di bawah detail CodeCatalyst spasi, pilih peran Tambah CodeCatalyst Dukungan. Opsi ini membuat peran layanan yang berisi kebijakan izin dan kebijakan kepercayaan untuk peran pengembangan pratinjau. Peran akan memiliki nama AWSRoleForCodeCatalystSupportdengan pengenal unik ditambahkan. Untuk informasi selengkapnya tentang kebijakan peran dan peran, lihatMemahami peran AWSRoleForCodeCatalystSupportlayanan.
6. Pada halaman Add role for CodeCatalyst Support, biarkan default dipilih, lalu pilih Create role.
7. Di bawah peran IAM yang tersedia CodeCatalyst, lihat `CodeCatalystWorkflowDevelopmentRole-spaceName` peran dalam daftar peran IAM yang ditambahkan ke akun Anda.
8. Untuk kembali ke ruang Anda, pilih Buka Amazon CodeCatalyst.

Setelah Anda membuat AWS Builder ID, membuat spasi pertama Anda, dan menambahkan akun, Anda kemudian dapat membuat proyek. Untuk informasi selengkapnya, lihat Membuat proyek. Jika ini adalah pertama kalinya Anda menggunakan CodeCatalyst, kami sarankan untuk memulai denganTutorial: Membuat proyek dengan cetak biru aplikasi web tiga tingkat Modern.

Menerima undangan dan membuat AWS Builder ID

Anda dapat mendaftar ke Amazon CodeCatalyst sebagai bagian dari menerima undangan ke proyek atau ruang. Sebagai bagian dari menerima undangan, Anda akan diminta untuk membuat AWS Builder ID. Anda akan menggunakan AWS Builder ID untuk mengakses sumber daya di CodeCatalyst.

> **Tip**
>
> Jika Anda membutuhkan bantuan tambahan, lihatMasalah mendaftar.

Berikut adalah salah satu aliran yang mungkin bagi pengguna yang memulai CodeCatalyst dengan undangan ke proyek atau ruang.

Saanvi Sarkar adalah pengembang yang telah menerima undangan untuk bergabung dengan CodeCatalyst proyek sebagai administrator proyek. Saanvi menerima undangan, yang membuka halaman masuk untuk. CodeCatalyst Dia memilih untuk mendaftar dan memberikan alamat email dan kata sandi untuk membuat ID AWS Pembangunnya. Saanvi akan dapat menggunakan AWS Builder ID untuk masuk CodeCatalyst dan aplikasi lainnya. Kemudian, dia dapat mengedit profilnya untuk mengubah alamat email atau kata sandi loginnya. Ketika diminta untuk memilih alias, Saanvi menentukan `SaanviSarkar` sebagai CodeCatalyst alias yang akan ditampilkan CodeCatalyst dan anggota proyek lainnya akan digunakan untuk @mention Saanvi. Setelah dia mendaftar, Saanvi juga akan dapat menggunakan kredensi masuknya untuk aplikasi lain yang menggunakan kredensi Builder ID. AWS

Setelah menyelesaikan pendaftaran, Saanvi secara otomatis bergabung dengan CodeCatalyst proyek dan ruang yang ditentukan dalam undangan. Undangan juga memberikan izin yang telah ditentukan untuk perannya dalam proyek dan ruang. Dalam pengaturan proyek, alias Saanvi ditampilkan dalam daftar anggota dengan peran proyek yang ditugaskan. Untuk bekerja dengan repositori sumber di CodeCatalyst, Saanvi membutuhkan waktu sejenak untuk membuat token akses pribadi (PAT). PAT akan digunakan CodeCatalyst untuk otentikasi saat membuat perubahan sumber atau tindakan yang memerlukan token otentikasi.

Ketika Saanvi mengerjakan sebuah proyek, aliasnya akan terdaftar di log aktivitas kerja untuk proyek tersebut. Masalah dan komentar oleh Saanvi akan menunjukkan aliasnya, di mana anggota proyek lain dapat @mention dia sebagai balasan. Untuk @mention anggota proyek lain, Saanvi mencari alias mereka di profil mereka. CodeCatalyst

Ketika dia memiliki momen, Saanvi mengonfigurasi ID AWS Pembangunnya untuk masuk CodeCatalyst dengan otentikasi multi-faktor (MFA). Dengan MFA yang dikonfigurasi, Saanvi dapat masuk CodeCatalyst menggunakan kombinasi CodeCatalyst kata sandinya dan kode sandi atau token dari aplikasi otentikasi pihak ketiga yang disetujui.

Menerima undangan dan membuat AWS Builder ID

Saat Anda diundang ke proyek atau ruang di Amazon CodeCatalyst, Anda akan menerima email dari notify@codecatalyst.aws yang meminta Anda untuk menerima undangan tersebut. Jika Anda sudah memiliki AWS Builder ID dan masuk ke CodeCatalyst, memilih Terima undangan akan secara otomatis membuka proyek atau spasi di tab browser. Jika Anda tidak masuk ke konsol tetapi memiliki AWS Builder ID, Anda akan dibawa ke halaman login. Untuk informasi selengkapnya, lihat Masuk dengan AWS Builder ID.

Jika Anda tidak memiliki AWS Builder ID, memilih Terima undangan akan membawa Anda ke halaman login, di mana Anda harus memilih opsi untuk membuat AWS Builder ID Anda.

Untuk menerima undangan dan membuat AWS Builder ID

1. Di email undangan, pilih Terima undangan.
2. Pada halaman masuk, pilih Tidak mendaftar? Buat ID AWS Builder Anda.

   > **Tip**
   >
   > AWS Builder ID Anda adalah identitas yang Anda buat untuk masuk. Ini tidak sama dengan Akun AWS.

3. Pada halaman Buat ID AWS Pembangun Anda, di alamat Email, masukkan alamat email yang ingin Anda gunakan untuk ID AWS Pembangun Anda.

   Dalam nama Anda, berikan nama depan dan belakang yang ingin ditampilkan dalam aplikasi tempat Anda menggunakan AWS Builder ID. Spasi diperbolehkan. Ini akan menjadi nama profil AWS Builder ID Anda, seperti Mary Major. Anda dapat mengubah nama nanti.

   Pilih Selanjutnya.

   Kode verifikasi akan dikirim ke email yang Anda tentukan. Masukkan kode ini dalam kode Verifikasi, lalu pilih Verifikasi. Jika Anda tidak menerima kode setelah 5 menit dan tidak dapat menemukannya di folder spam atau sampah, pilih Kirim ulang kode.

4. Setelah kode Anda diverifikasi, masukkan kata sandi yang memenuhi persyaratan di Kata Sandi dan Konfirmasi kata sandi.
5. Pilih Buat ID AWS Pembangun.
6. Pada halaman Buat alias Anda, masukkan alias yang ingin Anda gunakan untuk pengenal pengguna unik Anda. CodeCatalyst Pilih versi singkat dari nama Anda tanpa spasi, seperti MaryMajor. CodeCatalyst Pengguna lain akan menggunakan ini untuk @mention Anda di komentar dan permintaan tarik. CodeCatalyst Profil Anda akan berisi nama lengkap Anda dari AWS Builder ID dan CodeCatalyst alias Anda. Anda tidak dapat mengubah CodeCatalyst alias Anda.

   Nama lengkap Anda dan alias Anda akan ditampilkan di berbagai area di CodeCatalyst. Misalnya, nama profil Anda ditampilkan untuk aktivitas yang terdaftar di feed aktivitas, tetapi anggota proyek akan menggunakan alias Anda ke @mention Anda.

   Pilih Buat alias. Anda akan dibawa ke proyek atau ruang yang Anda undang.

Masuk dengan AWS Builder ID

Ikuti langkah-langkah ini untuk masuk ke CodeCatalyst profil Amazon Anda.

> **Note**
>
> Sudahkah Anda mendaftarkan perangkat untuk otentikasi multi-faktor (MFA)? Kami sangat menyarankan Anda mengonfigurasi MFA di Amazon CodeCatalyst untuk meningkatkan keamanan Anda. Untuk informasi selengkapnya, lihat Cara mendaftarkan perangkat untuk digunakan dengan otentikasi multi-faktor.

Untuk masuk dengan AWS Builder ID

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Masukkan alamat Email Anda. Secara opsional, pilih Simpan alamat email saya jika Anda ingin menyimpan alamat email Anda untuk login di masa mendatang. Pilih Continue (Lanjutkan).
3. Masukkan Kata Sandi Anda. Pilih Masuk. Jika Anda tidak ingat kata sandi Anda, ikuti langkah-langkahnyaSaya lupa kata sandi.

Perangkat tepercaya

Setelah Anda memilih opsi Ini adalah perangkat tepercaya dari halaman masuk, Amazon CodeCatalyst menganggap semua login di masa mendatang dari perangkat tersebut sebagai diizinkan. Amazon tidak CodeCatalyst akan menyajikan opsi untuk memasukkan kode MFA selama Anda menggunakan perangkat tepercaya itu. Beberapa pengecualian termasuk masuk dari browser baru atau ketika perangkat Anda telah mengeluarkan alamat IP yang tidak dikenal.

Masuk dengan SSO

Ikuti langkah-langkah berikut untuk menggunakan SSO untuk masuk ke Amazon CodeCatalyst.

Untuk masuk dengan AWS Builder ID Anda, lihatMasuk dengan AWS Builder ID.

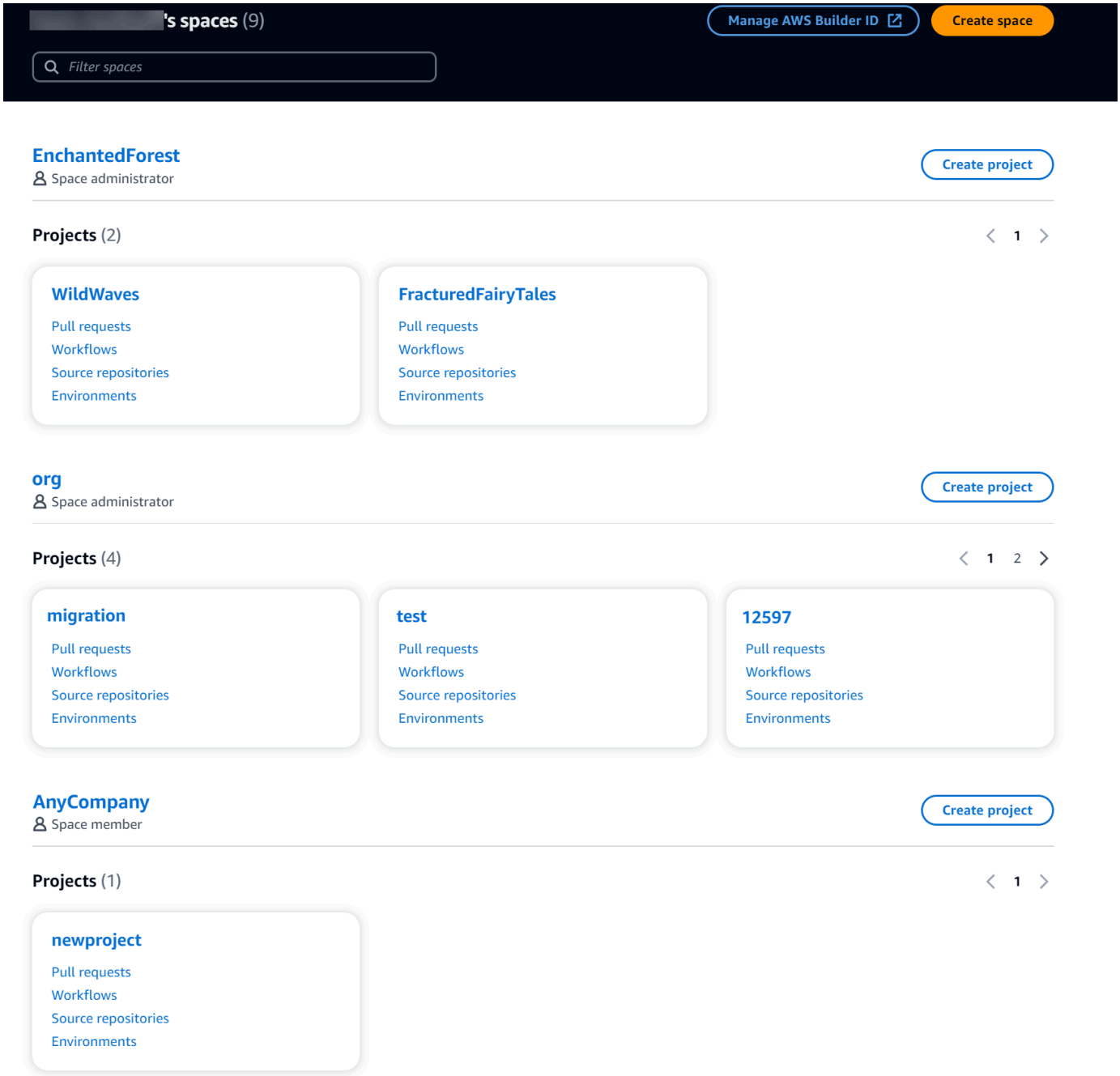
Untuk masuk dengan SSO

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di bawah Pilih opsi masuk, pilih Gunakan Single Sign-On (SSO).
3. Dalam nama aplikasi Pusat AWS Identitas, masukkan nama aplikasi yang disediakan oleh administrator federasi identitas Anda.
4. Pilih Lanjutkan ke Pusat Identitas IAM.

Lihat semua spasi dan proyek untuk pengguna

Anda dapat melihat daftar ruang dan proyek Anda di halaman beranda pengguna. Halaman beranda pengguna menunjukkan daftar setiap ruang yang dimiliki pengguna, peran pengguna di ruang itu, seperti administrator Space, dan proyek di setiap ruang tempat pengguna memiliki keanggotaan.

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di browser, masukkan alamat berikut: https://codecatalyst.aws/home



3. Pilih ruang atau proyek yang ingin Anda buka. Jika Anda tidak melihat spasi atau proyek yang diharapkan untuk dilihat, Anda mungkin perlu masuk sebagai pengguna lain.

Melihat dan mengelola CodeCatalyst profil

Anda dapat melihat profil pengguna di Amazon CodeCatalyst untuk mendapatkan informasi seperti alamat email dan CodeCatalyst alias. Anda juga dapat memperbarui profil dan AWS Builder ID Anda. Jika Anda lupa kata sandi, Anda dapat meminta pengaturan ulang kata sandi.

Melihat CodeCatalyst profil Anda

Anda memberikan informasi saat mendaftar yang akan digunakan sebagai kredensi Anda untuk masuk ke Amazon CodeCatalyst dan yang akan dikelola di profil Anda. Ini termasuk Nama, Nama Panggilan, dan alamat Email yang Anda gunakan untuk CodeCatalyst masuk.

> **Note**
>
> Nama Panggilan AWS Builder ID bukan CodeCatalyst alias Anda. Anda memilih CodeCatalyst alias Anda saat mendaftar.

Untuk melihat CodeCatalyst profil Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di kanan atas, pilih panah di sebelah ikon dengan inisiasi pertama Anda, lalu pilih Pengaturan saya. Halaman Pengaturan CodeCatalyst saya terbuka.
3. Untuk memperbarui alamat email atau kata sandi AWS Builder ID Anda, atau untuk menyiapkan MFA, pilih Kelola ID AWS Pembuat. Halaman AWS Builder ID terbuka.

Melihat CodeCatalyst profil pengguna lain

Untuk melihat CodeCatalyst profil pengguna lain

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pada navigasi samping, pilih Pengaturan proyek. Pilih tab Anggota. Lihat daftar anggota untuk CodeCatalyst proyek Anda.
3. Pilih nama anggota yang ingin Anda cari atau @mention. Halaman Pengaturan saya menunjukkan alias pengguna, alamat email, dan nama lengkap. Gunakan CodeCatalyst alias untuk anggota proyek @mention.

   > **Note**
   >
   > Nama Panggilan AWS Builder ID pengguna bukan CodeCatalyst alias mereka. Mereka memilih CodeCatalyst alias mereka saat mendaftar.

Untuk melihat profil pengguna lain di proyek Anda, pilih nama mereka dalam daftar.

Memperbarui profil Anda

Di CodeCatalyst, profil Anda terdiri dari informasi pribadi yang dikelola oleh AWSBuilder ID dan pengaturan yang dikelola di CodeCatalyst.

- Nama lengkap, alamat email, dan kata sandi profil Anda dikelola oleh AWSBuilder ID. Anda memasukkan informasi ini saat mendaftar. Saat Anda mengatur MFA untuk menggunakan aplikasi autentikator untuk login aplikasi, akan CodeCatalyst membawa Anda ke halaman Builder ID. AWS
- CodeCatalyst pengaturan untuk token akses pribadi (PAT), CodeCatalyst notifikasi, dan preferensi bahasa Anda dikelola di halaman Pengaturan saya di CodeCatalyst. Untuk informasi selengkapnya, lihat Berikan akses repositori pengguna dengan token akses pribadi.

> **Note**
>
> Anda dapat memperbarui nama lengkap AWS Builder ID Anda (nama CodeCatalyst tampilan) dan nama depan. Namun, Anda tidak dapat mengubah CodeCatalyst alias Anda.

Memperbarui AWS Builder ID atau alamat email

Untuk memperbarui alamat email ID AWS Builder atau Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di kanan atas, pilih panah di sebelah ikon dengan inisiasi pertama Anda, lalu pilih Pengaturan saya. Halaman Pengaturan CodeCatalyst saya terbuka.
3. Pada halaman profil, pilih Kelola ID AWS Pembangun. Halaman AWSBuilder ID terbuka.
4. Di sisi kiri halaman, pilih Detail saya.
5. Di bawah Informasi profil, pilih Edit untuk memperbarui Nama atau Nama Panggilan Anda. Jika Anda tidak menentukan nama panggilan, bidang Nama Panggilan mencerminkan nama depan dalam nama lengkap. Ini bukan CodeCatalyst alias Anda.

> **Note**
>
> Ini memperbarui nama lengkap dan nama depan AWS Builder ID. Ini tidak memperbarui CodeCatalyst alias Anda.

Di bawah Informasi kontak, pilih Edit untuk memperbarui alamat Email Anda.

> **Note**
>
> Ini memperbarui alamat email yang akan Anda gunakan untuk masuk CodeCatalyst.

Mengubah CodeCatalyst kata sandi Anda

Untuk mengubah CodeCatalyst kata sandi Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di kanan atas, pilih panah di sebelah ikon dengan inisiasi pertama Anda, lalu pilih Profil pengguna. Halaman Pengaturan CodeCatalyst saya terbuka.
3. Pada halaman profil, pilih Kelola ID AWS Pembangun. Halaman AWS Builder ID terbuka.
4. Di sisi kiri halaman, pilih Keamanan.
5. Pilih Ubah kata sandi dan ikuti petunjuknya.

Menyiapkan untuk menggunakan AWS CLI dengan CodeCatalyst

CodeCatalyst Konsol Amazon adalah tempat Anda akan mengerjakan sebagian besar tugas harian Anda. Namun, Anda mungkin ingin mengatur dan mengonfigurasi AWS CLI saat Anda bekerja dengan Lingkungan Pengembang, token akses pribadi, atau log peristiwa di CodeCatalyst. Anda harus menginstal AWS CLI dan mengkonfigurasi profil sebelum Anda dapat menggunakannya dengan CodeCatalyst.

Untuk mengatur AWS CLI untuk CodeCatalyst

1. Instal versi terbaru dari fileAWS CLI. Jika Anda sudah memiliki versi yang AWS CLI diinstal, pastikan itu terbaru dan termasuk perintah untuk CodeCatalyst, dan perbarui jika diperlukan.

Untuk memverifikasi bahwa Anda memiliki versi yang diinstal yang menyertakan CodeCatalyst perintah, buka prompt perintah dan jalankan perintah berikut:

```
aws codecatalyst help
```

Jika Anda melihat daftar CodeCatalyst perintah, Anda memiliki versi yang mendukung CodeCatalyst. Jika perintah tidak dikenali, perbarui versi Anda AWS CLI ke versi terbaru. Untuk informasi selengkapnya, lihat Menginstal atau memperbarui versi terbaru dari Panduan AWS Command Line Interface Pengguna. AWS CLI

2. Jalankan aws configure perintah untuk membuat profil jika Anda tidak memilikinya atau jika Anda ingin menggunakan profil bernama khusus untuk CodeCatalyst. Kami merekomendasikan membuat profil bernama untuk digunakan secara khusus CodeCatalyst, tetapi Anda juga dapat menggunakan profil default. Untuk informasi selengkapnya, lihat Dasar-dasar konfigurasi.
3. Edit `config` file untuk profil untuk menambahkan bagian untuk menghubungkan ke CodeCatalyst sebagai berikut. `config`File ini terletak `~/.aws/config` di Linux atau macOS, atau `C:\Users\`*`USERNAME`*`\.aws\config` di Windows.

```
[profile codecatalyst]
region = us-west-2
sso_session = codecatalyst

[sso-session codecatalyst]
sso_region = us-east-1
sso_start_url = https://view.awsapps.com/start
sso_registration_scopes = codecatalyst:read_write
```

4. Simpan file tersebut.
5. Sebelum mencoba menjalankan CodeCatalyst perintah apa pun, buka terminal atau command prompt baru dan jalankan perintah berikut untuk meminta dan mengambil kredensi untuk menjalankan perintah. `aws codecatalyst` Ganti `codecatalyst` dengan nama profil Anda jika diperlukan.

```
aws sso login --profile codecatalyst
```

Untuk melihat contoh codecatalyst perintah, lihat topik berikut:

- Berikan akses repositori pengguna dengan token akses pribadi

- Mengakses peristiwa yang dicatat menggunakan pencatatan peristiwa

Memulai tutorial

Amazon CodeCatalyst menyediakan sejumlah templat berbeda untuk membantu Anda memulai proyek Anda. Anda juga dapat memilih untuk memulai dengan proyek kosong dan menambahkan sumber daya ke dalamnya. Ikuti langkah-langkah dalam tutorial ini untuk mempelajari beberapa cara Anda dapat bekerja CodeCatalyst.

Jika ini adalah pertama kalinya Anda menggunakan CodeCatalyst, kami sarankan memulai dengan Tutorial: Membuat proyek dengan cetak biru aplikasi web tiga tingkat Modern.

> **Note**
>
> Untuk mengikuti tutorial ini, Anda harus terlebih dahulu menyelesaikan pengaturan. Untuk informasi selengkapnya, lihat Siapkan dan masuk ke CodeCatalyst.

Topik

- Tutorial: Membuat proyek dengan cetak biru aplikasi web tiga tingkat Modern
- Tutorial: Dimulai dengan proyek kosong dan menambahkan sumber daya secara manual
- Tutorial: Menggunakan fitur AI CodeCatalyst generatif untuk mempercepat pekerjaan pengembangan Anda
- Tutorial: Membuat aplikasi full-stack dengan cetak biru PDK yang dapat dikomposisi

Untuk tutorial tambahan yang berfokus pada area fungsional tertentu CodeCatalyst, lihat:

- Memulai dengan notifikasi Slack
- Memulai dengan repositori CodeCatalyst sumber dan cetak biru aplikasi Single-page
- Memulai dengan alur kerja
- Memulai dengan cetak biru khusus
- Memulai dengan panduan pengembang CodeCatalyst aksi Amazon

Untuk tutorial mendalam, lihat:

- Tutorial: Unggah artefak ke Amazon S3

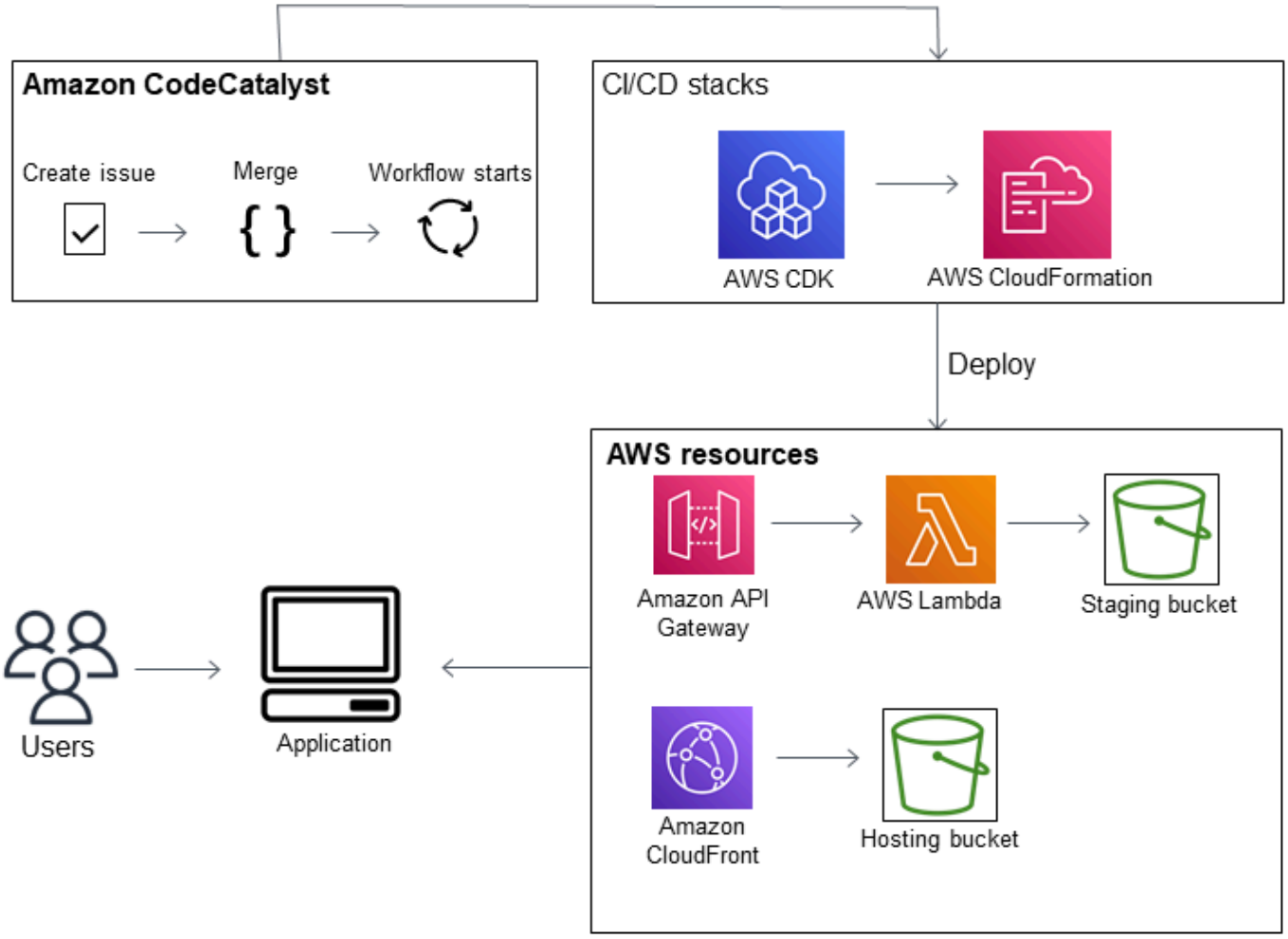
- Tutorial: Menyebarkan aplikasi tanpa server menggunakan AWS CloudFormation
- Tutorial: Menyebarkan aplikasi ke Amazon ECS
- Tutorial: Menyebarkan aplikasi ke Amazon EKS
- Tutorial: Kode lint menggunakan GitHub Action dalam alur kerja
- Tutorial: Membuat dan memperbarui aplikasi React

Tutorial: Membuat proyek dengan cetak biru aplikasi web tiga tingkat Modern

Anda dapat memulai lebih cepat dengan mengembangkan perangkat lunak dengan membuat proyek dengan cetak biru. Proyek yang dibuat dengan cetak biru mencakup sumber daya yang Anda butuhkan, termasuk repositori sumber untuk mengelola kode Anda, dan alur kerja untuk membangun dan menyebarkan aplikasi. Dalam tutorial ini, kami akan memandu Anda menggunakan cetak biru aplikasi web tiga tingkat Modern untuk membuat proyek di Amazon. CodeCatalyst Tutorial ini juga mencakup melihat sampel yang diterapkan, mengundang pengguna lain untuk mengerjakannya, dan membuat perubahan pada kode dengan permintaan tarik yang secara otomatis dibangun dan digunakan ke sumber daya yang terhubung Akun AWS saat permintaan tarik digabungkan. Saat CodeCatalyst membuat proyek Anda dengan laporan, umpan aktivitas, dan alat lainnya, cetak biru Anda membuat AWS sumber daya yang Akun AWS terkait dengan proyek Anda. File cetak biru Anda memungkinkan Anda untuk membangun dan menguji contoh aplikasi modern dan menyebarkannya ke infrastruktur di. AWS Cloud

Ilustrasi berikut menunjukkan bagaimana alat CodeCatalyst digunakan untuk membuat masalah untuk melacak, menggabungkan, dan membuat perubahan secara otomatis, lalu memulai alur kerja dalam CodeCatalyst proyek yang menjalankan tindakan untuk mengizinkan AWS CDK dan menyediakan infrastruktur AWS CloudFormation Anda.

Tindakan menghasilkan sumber daya yang terkait Akun AWS dan menerapkan aplikasi Anda ke AWS Lambda fungsi tanpa server dengan titik akhir API Gateway. AWS Cloud Development Kit (AWS CDK) Tindakan ini mengonversi satu atau beberapa AWS CDK tumpukan ke AWS CloudFormation templat dan menyebarkan tumpukan ke Anda. Akun AWS Sumber daya di tumpukan Anda mencakup CloudFront sumber daya Amazon untuk mendistribusikan konten web dinamis, instans Amazon DynamoDB untuk data aplikasi Anda, serta peran serta kebijakan yang mendukung aplikasi yang diterapkan.



Saat Anda membuat proyek dengan cetak biru aplikasi web tiga tingkat Modern, proyek Anda dibuat dengan sumber daya berikut:

Dalam CodeCatalyst proyek:

- Sebuah repositori sumber dengan kode sampel dan alur kerja YAMB
- Alur kerja yang membangun dan menerapkan kode sampel setiap kali perubahan dilakukan ke cabang default
- Papan masalah dan backlog yang dapat Anda gunakan untuk merencanakan dan melacak pekerjaan
- Rangkaian laporan pengujian dengan laporan otomatis yang disertakan dalam kode sampel

Dalam yang terkait Akun AWS:

- Tiga AWS CloudFormation tumpukan yang menciptakan sumber daya yang dibutuhkan untuk aplikasi.

Untuk detail yang diperluas tentang sumber daya yang akan dibuat di AWS dan CodeCatalyst sebagai bagian dari tutorial ini, lihatReferensi.

> **Note**
>
> Sumber daya dan sampel yang disertakan dalam proyek bergantung pada cetak biru yang Anda pilih. Amazon CodeCatalyst menawarkan beberapa cetak biru proyek yang mendefinisikan sumber daya yang terkait dengan bahasa atau kerangka kerja yang ditentukan. Untuk mempelajari lebih lanjut tentang cetak biru, lihat. Membuat proyek yang komprehensif dengan CodeCatalyst cetak biru

Topik

Prasyarat

Untuk membuat proyek aplikasi modern dalam tutorial ini, Anda harus telah menyelesaikan tugas-tugas Siapkan dan masuk ke CodeCatalyst sebagai berikut:

- Memiliki ID AWS Pembangun untuk masuk CodeCatalyst.

- Milik ruang dan memiliki administrator Space atau peran pengguna Power yang ditetapkan untuk Anda di ruang itu. Lihat informasi selengkapnya di Menciptakan ruang, Memberikan izin ruang kepada pengguna, dan Peran administrator ruang.
- Miliki Akun AWS hubungan dengan ruang Anda dan miliki peran IAM yang Anda buat saat mendaftar. Misalnya, saat mendaftar, Anda memiliki opsi untuk memilih membuat peran layanan dengan kebijakan peran yang disebut kebijakan CodeCatalystWorkflowDevelopmentRole-***spaceName***peran. Peran akan memiliki nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* dengan pengenal unik ditambahkan. Untuk informasi selengkapnya tentang kebijakan peran dan peran, lihatMemahami peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan. Untuk langkah-langkah untuk membuat peran, lihatMembuat CodeCatalystWorkflowDevelopmentRole-***spaceName***peran untuk akun dan ruang Anda.

Langkah 1: Buat proyek aplikasi web tiga tingkat modern

Setelah Anda membuatnya, proyek Anda adalah tempat Anda akan mengembangkan dan menguji kode, mengoordinasikan tugas pengembangan, dan melihat metrik proyek. Proyek Anda juga berisi alat dan sumber daya pengembangan Anda.

Dalam tutorial ini, Anda akan menggunakan cetak biru aplikasi web tiga tingkat Modern untuk membuat aplikasi interaktif. Alur kerja yang dibuat dan dijalankan secara otomatis sebagai bagian dari proyek Anda akan membangun dan menyebarkan aplikasi. Alur kerja hanya berjalan dengan sukses setelah semua peran dan informasi akun dikonfigurasi untuk ruang Anda. Setelah alur kerja berjalan dengan sukses, Anda dapat mengunjungi URL endpoint untuk melihat aplikasi.

Untuk membuat proyek dengan cetak biru

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, arahkan ke ruang tempat Anda ingin membuat proyek.
3. Pilih Buat proyek.
4. Pilih Mulai dengan cetak biru.
5. Di bilah pencarian, masukkan**`modern`**.
6. Pilih cetak biru aplikasi web tiga tingkat modern, lalu pilih Berikutnya.
7. Dalam Nama proyek Anda, masukkan nama proyek. Sebagai contoh:

   `MyExampleProject`.

> **Note**
>
> Nama harus unik di ruang Anda.

8. Di Akun, pilih yang Akun AWS Anda tambahkan saat mendaftar. Cetak biru akan menginstal sumber daya ke akun ini.
9. Di Peran Penerapan, pilih peran yang Anda tambahkan saat mendaftar. Misalnya, pilih `CodeCatalystWorkflowDevelopmentRole-spaceName`.

   Jika tidak ada peran yang terdaftar, tambahkan satu. Untuk menambahkan peran, pilih Tambahkan peran IAM dan tambahkan peran ke peran Anda Akun AWS. Untuk informasi selengkapnya, lihat Memungkinkan akses ke AWS sumber daya yang terhubung Akun AWS.
10. Di platform Compute, pilih Lambda.
11. Di Opsi Hosting Frontend, pilih Amplify Hosting. Untuk selengkapnya AWS Amplify, lihat Apa itu AWS Amplify Hosting? dalam AWS Amplify User Guide.
12. Di Wilayah Deployment, masukkan kode Region di Wilayah AWS mana Anda ingin cetak biru untuk menyebarkan aplikasi Mysfits dan sumber daya pendukung. Untuk daftar kode Wilayah, lihat Titik akhir Regional di. Referensi Umum AWS
13. Dalam nama Aplikasi, biarkan default dari`mysfits`*`string`*.
14. (Opsional) Di bawah Hasilkan pratinjau proyek, pilih Lihat kode untuk melihat pratinjau file sumber yang akan diinstal cetak biru. Pilih Lihat alur kerja untuk melihat pratinjau file definisi alur kerja CI/CD yang akan diinstal cetak biru. Pratinjau diperbarui secara dinamis berdasarkan pilihan Anda.
15. Pilih Buat proyek.

Alur kerja proyek dimulai segera setelah Anda membuat proyek. Butuh sedikit waktu untuk menyelesaikan pembangunan dan penerapan kode. Sementara itu, silakan dan undang orang lain ke proyek Anda.

Langkah 2: Undang seseorang ke proyek Anda

Sekarang setelah Anda menyiapkan proyek Anda, undang orang lain untuk bekerja dengan Anda.

Untuk mengundang seseorang ke proyek Anda

1. Arahkan ke proyek yang ingin Anda undang pengguna.

2. Di panel navigasi, pilih Pengaturan proyek.
3. Pada tab Anggota, pilih Undang.
4. Ketik alamat email orang yang ingin Anda undang sebagai pengguna untuk proyek Anda. Anda dapat mengetik beberapa alamat email yang dipisahkan oleh spasi atau koma. Anda juga dapat memilih dari anggota ruang Anda yang bukan anggota proyek.
5. Pilih peran untuk pengguna.

   Setelah selesai menambahkan pengguna, pilih Undang.

Langkah 3: Buat masalah untuk berkolaborasi dan melacak pekerjaan

CodeCatalyst membantu Anda melacak fitur, tugas, bug, dan pekerjaan lain yang terlibat dalam proyek Anda dengan masalah. Anda dapat membuat masalah untuk melacak pekerjaan dan ide yang dibutuhkan. Secara default, ketika Anda membuat masalah itu ditambahkan ke backlog Anda. Anda dapat memindahkan masalah ke papan tempat Anda melacak pekerjaan yang sedang berlangsung. Anda juga dapat menetapkan masalah ke anggota proyek tertentu.

Untuk membuat masalah untuk sebuah proyek

1. Di panel navigasi, pilih Masalah.
2. Pilih Buat masalah.
3. Di judul Masalah, berikan nama untuk masalah tersebut. Secara opsional, berikan deskripsi masalah. Dalam contoh ini, gunakan **`make a change in the src/mysfit_data.json file.`**
4. Pilih prioritas, estimasi, status, dan label. Di bawah penerima tugas, pilih +Tambahkan saya untuk menetapkan masalah untuk diri Anda sendiri.
5. Pilih Buat masalah. Masalahnya sekarang terlihat di papan tulis. Pilih kartu untuk memindahkan masalah ke kolom Sedang berlangsung.

Untuk informasi selengkapnya, lihat Lacak dan atur pekerjaan dengan masalah di CodeCatalyst.

Langkah 4: Lihat repositori sumber Anda

Cetak biru Anda menginstal repositori sumber yang berisi file untuk menentukan dan mendukung aplikasi atau layanan Anda. Beberapa direktori dan file penting dalam repositori sumber adalah:

- direktori.cloud9 - Berisi file pendukung untuk Lingkungan Dev. AWS Cloud9
- .codecatalyst directory - Berisi file definisi alur kerja untuk setiap `YAML` alur kerja yang disertakan dalam cetak biru.
- direktori.idea - Berisi file pendukung untuk Lingkungan JetBrains Dev.
- direktori.vscode - Berisi file pendukung untuk Lingkungan Pengembang Kode Visual Studio.
- Direktori CdkStacks - Berisi file AWS CDK tumpukan yang menentukan infrastruktur di file. AWS Cloud
- direktori src - Berisi kode sumber aplikasi.
- direktori tests - Berisi file untuk integ dan pengujian unit yang dijalankan sebagai bagian dari alur kerja CI/CD otomatis yang berjalan saat Anda membangun dan menguji aplikasi Anda.
- direktori web - Berisi kode sumber frontend. File lain termasuk file proyek seperti `package.json` file yang berisi metadata penting tentang proyek Anda, `index.html` halaman untuk situs web, `.eslintrc.cjs` file untuk kode linting, dan file untuk menentukan `tsconfig.json` file root dan opsi kompiler.
- `Dockerfile`file - Menjelaskan wadah aplikasi.
- `README.md`file - Berisi informasi konfigurasi untuk proyek.

Untuk menavigasi ke repositori sumber untuk sebuah proyek

1. Arahkan ke proyek Anda, dan lakukan salah satu hal berikut:
   - Pada halaman ringkasan untuk proyek Anda, pilih repositori yang Anda inginkan dari daftar, lalu pilih Lihat repositori.
   - Di panel navigasi, pilih Kode, lalu pilih Repositori sumber. Di repositori Sumber, pilih nama repositori dari daftar. Anda dapat memfilter daftar repositori dengan mengetikkan bagian dari nama repositori di bilah filter.
2. Pada halaman beranda untuk repositori, lihat isi repositori dan informasi tentang sumber daya terkait seperti jumlah permintaan tarik dan alur kerja. Secara default, konten untuk cabang default ditampilkan. Anda dapat mengubah tampilan dengan memilih cabang yang berbeda dari daftar drop-down.

Langkah 5: Buat Lingkungan Pengembang dengan cabang pengujian dan buat perubahan kode cepat

Anda dapat dengan cepat mengerjakan kode di repositori sumber Anda dengan membuat Lingkungan Dev. Untuk tutorial ini, kami berasumsi Anda akan:

- Buat Lingkungan AWS Cloud9 Pengembang.
- Pilih opsi untuk bekerja di cabang baru dari cabang utama saat membuat Lingkungan Dev.
- Gunakan nama `test` untuk cabang baru ini.

Pada langkah selanjutnya, Anda akan menggunakan Lingkungan Dev untuk membuat perubahan kode dan membuat permintaan tarik.

Untuk membuat Lingkungan Dev dengan cabang baru

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek tempat Anda ingin membuat Lingkungan Dev.
3. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, pilih repositori Sumber, dan pilih repositori yang ingin Anda buat Lingkungan Dev.
4. Pada halaman rumah repositori, pilih Create Dev Environment.
5. Pilih IDE yang didukung dari menu tarik-turun. Untuk informasi selengkapnya, lihat Lingkungan pengembangan terintegrasi yang didukung untuk Lingkungan Dev.
6. Pilih repositori untuk dikloning, pilih Bekerja di cabang baru, masukkan nama cabang ke bidang Nama cabang, dan pilih cabang untuk membuat cabang baru dari Buat cabang dari menu drop-down.
7. Secara opsional, tambahkan alias untuk Lingkungan Dev.
8. Secara opsional, pilih tombol edit konfigurasi Lingkungan Dev untuk mengedit konfigurasi komputasi, penyimpanan, atau batas waktu Dev Environment.
9. Pilih Buat. Saat Dev Environment Anda sedang dibuat, kolom status Dev Environment akan menampilkan Mulai, dan kolom status akan ditampilkan Running setelah Dev Environment telah dibuat. Tab baru akan terbuka dengan Lingkungan Dev Anda di IDE pilihan Anda. Anda dapat mengedit kode dan melakukan dan mendorong perubahan Anda.

Di bagian ini, Anda akan bekerja dengan aplikasi sampel yang Anda hasilkan CodeCatalyst dengan membuat perubahan pada kode dengan permintaan tarik yang secara otomatis dibangun dan digunakan ke sumber daya yang terhubung Akun AWS saat permintaan tarik digabungkan.

Untuk membuat perubahan pada **`src/mysfit_data.json`** file Anda

1. Arahkan ke Lingkungan Pengembang proyek Anda. Di AWS Cloud9, perluas menu navigasi samping untuk menelusuri file. Perluas`mysfits`,`src`, dan buka`src/mysfit_data.json`.
2. Dalam file, ubah nilai untuk `"Age":` bidang dari 6 menjadi 12. Baris Anda akan terlihat seperti berikut:

```
    {
        "Age": 12,
        "Description": "Twilight's personality sparkles like the night sky and is
 looking for a forever home with a Greek hero or God. While on the smaller side
 at 14 hands, he is quite adept at accepting riders and can fly to 15,000 feet.
 Twilight needs a large area to run around in and will need to be registered with
 the FAA if you plan to fly him above 500 feet. His favorite activities include
 playing with chimeras, going on epic adventures into battle, and playing with a
 large inflatable ball around the paddock. If you bring him home, he'll quickly
 become your favorite little Pegasus.",
        "GoodEvil": "Good",
        "LawChaos": "Lawful",
        "Name": "Twilight Glitter",
        "ProfileImageUri": "https://www.mythicalmysfits.com/images/
pegasus_hover.png",
        "Species": "Pegasus",
        "ThumbImageUri": "https://www.mythicalmysfits.com/images/pegasus_thumb.png"
    },
```

3. Simpan file tersebut.
4. Ubah ke repositori mysfits dengan perintah. **`cd /projects/mysfits`**
5. Tambahkan, komit, dan dorong perubahan Anda dengan perintah git add, git commit, dan git push.

```
git add .
git commit -m "make an example change"
git push
```

Langkah 6: Lihat alur kerja yang membangun aplikasi modern

Setelah membuat proyek aplikasi modern, CodeCatalyst hasilkan beberapa sumber daya atas nama Anda, termasuk alur kerja. Alur kerja adalah prosedur otomatis yang ditentukan dalam file.yaml yang menjelaskan cara membuat, menguji, dan menyebarkan kode Anda.

Dalam tutorial ini, CodeCatalyst buat alur kerja dan mulai secara otomatis ketika Anda membuat proyek Anda. (Alur kerja mungkin masih berjalan tergantung pada berapa lama Anda membuat proyek Anda.) Gunakan prosedur berikut untuk memeriksa kemajuan alur kerja, meninjau log yang dihasilkan dan laporan pengujian, dan terakhir, arahkan ke URL aplikasi yang digunakan.

Untuk memeriksa kemajuan alur kerja

1. Di CodeCatalyst konsol, di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.

   Daftar alur kerja muncul. Ini adalah alur kerja yang dihasilkan dan dimulai oleh CodeCatalyst cetak biru saat Anda membuat proyek.
2. Amati daftar alur kerja. Anda harus melihat empat:

   - Dua alur kerja di bagian atas sesuai dengan `test` cabang yang Anda buat sebelumnya. Langkah 5: Buat Lingkungan Pengembang dengan cabang pengujian dan buat perubahan kode cepat Alur kerja ini adalah klon alur kerja di cabang. `main` ApplicationDeploymentPipelineIni tidak aktif karena dikonfigurasi untuk digunakan dengan `main` cabang. OnPullRequestAlur kerja tidak berjalan karena tidak ada permintaan tarik yang dibuat.
   - Dua alur kerja di bagian bawah sesuai dengan `main` cabang yang dibuat saat Anda menjalankan cetak biru sebelumnya. ApplicationDeploymentPipelineAlur kerja aktif dan berjalan dalam proses (atau selesai).

     > **Note**
     >
     > Jika proses ApplicationDeploymentPipelinegagal dengan Build @cdk_bootstrap atau DeployBackendkesalahan, itu mungkin karena Anda menjalankan aplikasi web tiga tingkat Modern sebelumnya, dan itu meninggalkan sumber daya lama di belakang konflik itu dengan cetak biru saat ini. Anda harus menghapus sumber daya lama ini dan kemudian menjalankan kembali alur kerja. Untuk informasi selengkapnya, lihat Pembersihan sumber daya.

3. Pilih ApplicationDeploymentPipelinealur kerja yang terkait dengan `main` cabang, di bagian bawah. Alur kerja ini dijalankan menggunakan kode sumber di `main` cabang.

   Diagram alur kerja muncul. Diagram menunjukkan beberapa blok, masing-masing mewakili tugas atau tindakan. Sebagian besar tindakan diatur secara vertikal, dengan tindakan di bagian atas berjalan sebelum yang di bawah ini. Tindakan yang diatur berdampingan berjalan secara paralel. Tindakan yang dikelompokkan bersama harus berjalan dengan sukses sebelum tindakan di bawahnya dapat dimulai.

   Blok utama adalah:

   - WorkflowSource— Blok ini mewakili repositori sumber Anda. Ini menunjukkan, di antara informasi lainnya, nama repositori sumber (mysfits) dan komit yang secara otomatis memulai alur kerja berjalan. CodeCatalyst menghasilkan komit ini saat Anda membuat proyek Anda.
   - Build — Blok ini mewakili pengelompokan dua tindakan yang keduanya harus berhasil diselesaikan agar tindakan selanjutnya dimulai.
   - DeployBackend— Blok ini merupakan tindakan yang menyebarkan komponen backend aplikasi ke cloud. AWS
   - Pengujian — Blok ini mewakili pengelompokan dua tindakan pengujian yang harus diselesaikan dengan sukses agar tindakan selanjutnya dimulai.
   - DeployFrontend— Blok ini merupakan tindakan yang menyebarkan komponen frontend aplikasi ke cloud. AWS
4. Pilih tab Definisi (dekat bagian atas). File definisi alur kerja muncul di sebelah kanan. File ini memiliki bagian penting berikut:

   - `Triggers`Bagian, di bagian atas. Bagian ini menunjukkan bahwa alur kerja harus dimulai setiap kali kode didorong ke cabang repositori sumber. `main` Mendorong ke cabang lain (seperti`test`) tidak akan memulai alur kerja ini. Alur kerja berjalan menggunakan file di `main` cabang.
   - `Actions`Bagian, di bawah`Triggers`. Bagian ini mendefinisikan tindakan yang Anda lihat dalam diagram alur kerja.
5. Pilih tab Status terbaru (dekat bagian atas), dan pilih tindakan apa pun dalam diagram alur kerja.
6. Di sebelah kanan, pilih tab Konfigurasi untuk melihat pengaturan konfigurasi yang digunakan oleh tindakan selama proses terbaru. Setiap pengaturan konfigurasi memiliki properti yang cocok dalam file definisi alur kerja.
7. Biarkan konsol terbuka dan pergi ke prosedur berikutnya.

Untuk meninjau log build dan laporan pengujian

1. Pilih tab Status terbaru.
2. Dalam diagram alur kerja, pilih DeployFrontendtindakan.
3. Tunggu sampai aksi selesai. Perhatikan ikon “dalam proses” () untuk berubah menjadi ikon “sukses” ()
4. Pilih tindakan build_backend.
5. Pilih tab Log, dan perluas beberapa bagian untuk melihat pesan log untuk langkah-langkah ini. Anda dapat melihat pesan yang terkait dengan pengaturan backend.
6. Pilih tab Laporan, lalu pilih `backend-coverage.xml` laporan. CodeCatalyst menampilkan laporan terkait. Laporan menunjukkan tes cakupan kode yang dijalankan, dan menunjukkan proporsi baris kode yang berhasil divalidasi dengan pengujian, seperti 80%.

   Untuk informasi selengkapnya tentang laporan pengujian, lihatPengujian dengan alur kerja.

   > **Tip**
   >
   > Anda juga dapat melihat laporan pengujian dengan memilih Laporan di panel navigasi.

7. Biarkan CodeCatalyst konsol terbuka, dan pergi ke prosedur berikutnya.

Untuk mengonfirmasi bahwa aplikasi modern berhasil digunakan

1. Kembali ke ApplicationDeploymentPipelinealur kerja, dan pilih tautan Run- ***`string`*** dari proses terbaru.
2. Dalam diagram alur kerja, temukan DeployFrontendtindakan dan pilih tautan Lihat aplikasi. Situs web Mysfit muncul.

   > **Note**
   >
   > Jika Anda tidak melihat tautan Lihat aplikasi di dalam DeployFrontendtindakan, pastikan Anda memilih tautan run ID.

3. Cari pegasus Mysfit bernama Twilight Glitter. Perhatikan nilai untuk usia. Ini adalah6. Anda akan membuat perubahan kode untuk memperbarui usia.

Langkah 7: Minta orang lain untuk meninjau perubahan Anda

Sekarang setelah Anda memiliki perubahan di cabang bernama`test`, Anda dapat meminta orang lain untuk meninjaunya dengan membuat permintaan tarik. Lakukan langkah-langkah berikut untuk membuat permintaan tarik untuk menggabungkan perubahan dari `test` cabang ke `main` cabang.

Untuk membuat permintaan tarik

1. Arahkan ke proyek Anda.
2. Lakukan salah satu hal berikut:
   - Di panel navigasi, pilih Kode, pilih Tarik permintaan, lalu pilih Buat permintaan tarik.
   - Pada halaman beranda repositori, pilih Lainnya, lalu pilih Buat permintaan tarik.
   - Pada halaman proyek, pilih Buat permintaan tarik.
3. Di repositori Sumber, pastikan bahwa repositori sumber yang ditentukan adalah yang berisi kode yang dikomit. Opsi ini hanya muncul jika Anda tidak membuat permintaan tarik dari halaman utama repositori.
4. Di cabang Tujuan, pilih cabang untuk menggabungkan kode setelah ditinjau.
5. Di cabang Sumber, pilih cabang yang berisi kode komit.
6. Dalam judul permintaan tarik, masukkan judul yang membantu pengguna lain memahami apa yang perlu ditinjau dan alasannya.
7. (Opsional) Dalam deskripsi permintaan Tarik, berikan informasi seperti tautan ke masalah atau deskripsi perubahan Anda.

> **Tip**
>
> Anda dapat memilih Tulis deskripsi agar saya CodeCatalyst secara otomatis menghasilkan deskripsi tentang perubahan yang terkandung dalam permintaan tarik. Anda dapat membuat perubahan pada deskripsi yang dibuat secara otomatis setelah Anda menambahkannya ke permintaan tarik.
> Fungsionalitas ini mengharuskan fitur AI generatif diaktifkan untuk ruang tersebut. Untuk informasi selengkapnya, lihat Mengelola fitur AI generatif.

8. (Opsional) Dalam Masalah, pilih Masalah tautan, lalu pilih masalah dari daftar atau masukkan ID-nya. Untuk memutuskan tautan masalah, pilih ikon batalkan tautan.
9. (Opsional) Di Reviewer wajib, pilih Tambahkan pengulas yang diperlukan. Pilih dari daftar anggota proyek untuk menambahkannya. Peninjau yang diperlukan harus menyetujui perubahan sebelum permintaan tarik dapat digabungkan ke cabang tujuan.

   > **Note**
   >
   > Anda tidak dapat menambahkan pengulas sebagai pengulas yang diperlukan dan pengulas opsional. Anda tidak dapat menambahkan diri Anda sebagai reviewer.

10. (Opsional) Di pengulas opsional, pilih Tambahkan pengulas opsional. Pilih dari daftar anggota proyek untuk menambahkannya. Peninjau opsional tidak harus menyetujui perubahan sebagai persyaratan sebelum permintaan tarik dapat digabungkan ke cabang tujuan.
11. Tinjau perbedaan antara cabang. Perbedaan yang ditampilkan dalam permintaan tarik adalah perubahan antara revisi di cabang sumber dan basis gabungan, yang merupakan komit kepala cabang tujuan pada saat permintaan tarik dibuat. Jika tidak ada perubahan yang ditampilkan, cabang mungkin identik, atau Anda mungkin telah memilih cabang yang sama untuk sumber dan tujuan.
12. Ketika Anda puas bahwa permintaan tarik berisi kode dan perubahan yang ingin Anda tinjau, pilih Buat.

    > **Note**
    >
    > Setelah Anda membuat permintaan tarik, Anda dapat menambahkan komentar. Komentar dapat ditambahkan ke permintaan tarik atau ke baris individual dalam file serta permintaan tarik keseluruhan. Anda dapat menambahkan tautan ke sumber daya, seperti file, dengan menggunakan tanda @ diikuti dengan nama file.

Saat Anda membuat permintaan tarik, OnPullRequestalur kerja mulai menggunakan file sumber di `test` cabang. Saat pengulas menyetujui perubahan kode, Anda dapat mengamati hasilnya dengan memilih alur kerja dan melihat hasil pengujian.

Setelah perubahan ditinjau, Anda dapat menggabungkan kode. Menggabungkan kode ke cabang default akan secara otomatis memulai alur kerja yang akan membangun dan menyebarkan perubahan Anda.

Untuk menggabungkan permintaan tarik dari konsol CodeCatalyst

1. Arahkan ke proyek aplikasi modern Anda.
2. Pada halaman proyek, di bawah Buka permintaan tarik, pilih permintaan tarik yang ingin Anda gabungkan. Jika Anda tidak melihat permintaan tarik, pilih Lihat semua dan kemudian pilih dari daftar. Pilih Gabungkan.
3. Pilih dari strategi penggabungan yang tersedia untuk permintaan tarik. Secara opsional pilih atau batalkan pilihan untuk menghapus cabang sumber setelah menggabungkan permintaan tarik, dan kemudian pilih Gabung.

> **Note**
>
> Jika tombol Gabung tidak aktif, atau Anda melihat label Tidak dapat digabungkan, salah satu atau beberapa pengulas yang diperlukan belum menyetujui permintaan tarik, atau permintaan tarik tidak dapat digabungkan di konsol. CodeCatalyst Peninjau yang belum menyetujui permintaan tarik ditunjukkan oleh ikon jam di Ikhtisar di area detail permintaan Tarik. Jika semua pengulas yang diperlukan telah menyetujui permintaan tarik tetapi tombol Gabung masih belum aktif, Anda mungkin memiliki konflik gabungan. Anda dapat menyelesaikan konflik gabungan untuk cabang tujuan di CodeCatalyst konsol dan kemudian menggabungkan permintaan tarik, atau Anda dapat menyelesaikan konflik dan menggabungkan secara lokal, lalu mendorong komit yang berisi penggabungan ke. CodeCatalyst Untuk informasi selengkapnya, lihat Menggabungkan permintaan tarik (Git) dan dokumentasi Git Anda.

Setelah Anda menggabungkan perubahan dari `test` cabang ke **`main`** cabang, perubahan secara otomatis memulai ApplicationDeploymentPipelinealur kerja yang membangun dan menyebarkan perubahan Anda.

Untuk melihat komit gabungan berjalan melalui alur kerja ApplicationDeploymentPipeline

1. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Di Alur Kerja, di ApplicationDeploymentPipeline, perluas proses Terbaru. Anda dapat melihat alur kerja yang dijalankan oleh komit gabungan. Secara opsional pilih untuk menonton kemajuan lari.
3. Saat proses selesai, muat ulang URL yang Anda kunjungi sebelumnya. Lihat pegasus untuk memverifikasi bahwa usia berubah.

Langkah 8: Tutup masalah

Ketika masalah teratasi, itu dapat ditutup di CodeCatalyst konsol.

Untuk menutup masalah untuk sebuah proyek

1. Arahkan ke proyek Anda.
2. Di panel navigasi, pilih Masalah.
3. D rag-and-drop masalah ke kolom Selesai.

Untuk informasi selengkapnya, lihat Lacak dan atur pekerjaan dengan masalah di CodeCatalyst.

Pembersihan sumber daya

Bersihkan CodeCatalyst dan AWS untuk menghapus jejak tutorial ini dari lingkungan Anda.

Anda dapat memilih untuk tetap menggunakan proyek yang Anda gunakan untuk tutorial ini, atau Anda dapat menghapus proyek dan sumber daya yang terkait.

> **Note**
>
> Menghapus proyek ini akan menghapus semua repositori, masalah, dan artefak dalam proyek untuk semua anggota.

Untuk menghapus proyek

1. Arahkan ke proyek Anda, lalu pilih Pengaturan proyek.
2. Pilih tab Umum.
3. Di bawah nama proyek, pilih Hapus proyek.

Untuk menghapus sumber daya di AWS CloudFormation dan Amazon S3

1. Masuk ke akun AWS Management Console dengan akun yang sama yang Anda tambahkan ke CodeCatalyst ruang Anda.
2. Pergi ke AWS CloudFormationlayanan.
3. Hapus tumpukan ***`string`*** mysfits.
4. *`Hapus tumpukan`* ***`string development-mysfits.`***
5. Pilih (tapi jangan hapus) tumpukan CDKToolkit. Pilih tab Sumber Daya. Pilih StagingBuckettautannya, lalu hapus isi bucket dan bucket di Amazon S3.

   > **Note**
   >
   > Jika Anda tidak menghapus bucket ini secara manual, Anda mungkin akan melihat kesalahan saat menjalankan ulang cetak biru aplikasi web tiga tingkat Modern.

6. (Opsional) Hapus tumpukan CDKToolkit.

Referensi

Cetak biru aplikasi web tiga tingkat Modern menyebarkan sumber daya ke CodeCatalyst ruang Anda dan akun Anda di cloud. AWS AWS Sumber daya ini adalah:

- Di CodeCatalyst ruang Anda:
  - CodeCatalyst Proyek yang mencakup sumber daya berikut:

    - Repositori sumber - Repositori ini berisi kode contoh untuk aplikasi web 'Mysfits'.
    - Alur kerja - Alur kerja ini membangun dan menyebarkan kode aplikasi Mysfits setiap kali perubahan dilakukan ke cabang default
    - Papan masalah dan backlog — Papan dan backlog ini dapat digunakan untuk merencanakan dan melacak pekerjaan.
    - Rangkaian laporan pengujian — Suite ini mencakup laporan otomatis yang disertakan dalam kode sampel.
- Dalam yang terkait Akun AWS:
    - Tumpukan CDKToolkit - Tumpukan ini menyebarkan sumber daya berikut:
        - Bucket pementasan Amazon S3, kebijakan bucket, dan AWS KMS kunci yang digunakan untuk mengenkripsi bucket.
        - Peran penyebaran IAM untuk tindakan penerapan.
        - AWS Peran dan kebijakan IAM dalam mendukung sumber daya dalam tumpukan.

        > **Note**
        >
        > CDKToolkit tidak diruntuhkan dan dibuat ulang untuk setiap penerapan. Ini adalah tumpukan yang dimulai di setiap akun untuk mendukung. AWS CDK
    - BackEndTumpukan ***`string`*** development-mysfits - Tumpukan ini menyebarkan sumber daya backend berikut:
        - Titik akhir Amazon API Gateway.
        - AWS Peran dan kebijakan IAM dalam mendukung sumber daya dalam tumpukan.
        - AWS Lambda Fungsi dan lapisan menyediakan platform komputasi tanpa server untuk aplikasi modern.
        - Kebijakan dan peran IAM untuk penerapan bucket dan fungsi Lambda.
    - Tumpukan ***`string`*** mysfits — Tumpukan ini menyebarkan aplikasi frontend. AWS Amplify

Lihat juga

Untuk informasi selengkapnya tentang AWS layanan tempat sumber daya dibuat sebagai bagian dari tutorial ini, lihat berikut ini:

- Amazon S3 — Layanan untuk menyimpan aset frontend Anda pada layanan penyimpanan objek yang menawarkan skalabilitas terdepan di industri, ketersediaan data tinggi, keamanan, dan kinerja. Untuk informasi selengkapnya, lihat Panduan Pengguna Amazon S3.
- Amazon API Gateway — Layanan untuk membuat, menerbitkan, memelihara, memantau, dan mengamankan REST, HTTP, dan WebSocket API pada skala apa pun Untuk informasi selengkapnya, lihat Panduan Pengembang API Gateway.
- Amplify — Layanan untuk hosting aplikasi frontend Anda. Untuk informasi selengkapnya, lihat Panduan Pengguna AWS Amplify Hosting.
- AWS Cloud Development Kit (AWS CDK)— Kerangka kerja untuk mendefinisikan infrastruktur cloud dalam kode dan menyediakannya. AWS CloudFormation AWS CDK Termasuk AWS CDK Toolkit, yang merupakan alat baris perintah untuk berinteraksi dengan AWS CDK aplikasi dan tumpukan. Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS Cloud Development Kit (AWS CDK).
- Amazon DynamoDB — Layanan database NoSQL yang dikelola sepenuhnya untuk menyimpan data. Untuk informasi selengkapnya, lihat Panduan Developer Amazon DynamoDB.
- AWS Lambda— Layanan untuk menjalankan kode Anda pada infrastruktur komputasi ketersediaan tinggi tanpa menyediakan atau mengelola server. Untuk informasi selengkapnya, silakan lihat Panduan Developer AWS Lambda.
- AWS IAM — Layanan untuk mengontrol akses AWS dan sumber dayanya dengan aman. Untuk informasi selengkapnya, lihat Panduan Pengguna IAM.

Tutorial: Dimulai dengan proyek kosong dan menambahkan sumber daya secara manual

Anda dapat membuat proyek kosong tanpa sumber daya yang telah ditentukan sebelumnya di dalamnya dengan memilih cetak biru proyek Kosong saat Anda membuat proyek. Setelah Anda membuat proyek kosong, Anda dapat membuat dan menambahkan sumber daya sesuai dengan kebutuhan proyek Anda. Karena proyek yang dibuat tanpa cetak biru kosong saat pembuatan, opsi ini membutuhkan lebih banyak pengetahuan tentang membuat dan mengonfigurasi CodeCatalyst sumber daya untuk memulai.

Topik

- Buat repositori sumber
- Buat alur kerja untuk membangun, menguji, dan menerapkan perubahan kode
- Undang seseorang ke proyek Anda
- Buat masalah untuk berkolaborasi dan melacak pekerjaan

Prasyarat

Untuk membuat proyek kosong, Anda harus memiliki administrator Space atau peran pengguna Power yang ditetapkan untuk Anda. Jika ini adalah pertama kalinya Anda masuk CodeCatalyst, lihatSiapkan dan masuk ke CodeCatalyst.

Buat proyek kosong

Membuat proyek adalah langkah pertama untuk dapat bekerja sama. Jika Anda ingin membuat sumber daya Anda sendiri, seperti repositori sumber dan alur kerja, Anda dapat memulai dengan proyek kosong.

Untuk membuat proyek kosong

1. Arahkan ke ruang tempat Anda ingin membuat proyek.
2. Di dasbor ruang, pilih Buat proyek.
3. Pilih Mulai dari awal.
4. Di bawah Berikan nama untuk proyek Anda, masukkan nama yang ingin Anda tetapkan ke proyek Anda. Nama harus unik di dalam ruang Anda.
5. Pilih Buat proyek.

Sekarang Anda memiliki proyek kosong, langkah selanjutnya adalah membuat repositori sumber.

Buat repositori sumber

Buat repositori sumber untuk menyimpan dan berkolaborasi pada kode proyek Anda. Anggota proyek dapat mengkloning repositori ini ke komputer lokal mereka untuk bekerja pada kode. Atau, Anda dapat memilih untuk menautkan repositori yang dihosting di layanan yang didukung, tetapi itu tidak tercakup dalam tutorial ini. Untuk informasi selengkapnya, lihat Menautkan repositori sumber.

Untuk membuat repositori sumber

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda.
3. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
4. Pilih Tambahkan repositori, lalu pilih Buat repositori.
5. Dalam nama Repositori, berikan nama untuk repositori. Dalam panduan ini, kami menggunakan *codecatalyst-source-repository*, tetapi Anda dapat memilih nama yang berbeda. Nama repositori harus unik dalam sebuah proyek. Untuk informasi selengkapnya tentang persyaratan untuk nama repositori, lihat. Kuota untuk repositori sumber di CodeCatalyst
6. (Opsional) Dalam Deskripsi, tambahkan deskripsi untuk repositori yang akan membantu pengguna lain dalam proyek memahami untuk apa repositori digunakan.
7. (Opsional) Tambahkan `.gitignore` file untuk jenis kode yang Anda rencanakan untuk dorong.
8. Pilih Buat.

> **Note**
>
> CodeCatalyst menambahkan `README.md` file ke repositori Anda saat Anda membuatnya. CodeCatalystjuga membuat komit awal untuk repositori di cabang default bernama main. Anda dapat mengedit atau menghapus file README.md, tetapi Anda tidak dapat mengubah atau menghapus cabang default.

Anda dapat dengan cepat menambahkan kode di repositori Anda dengan membuat Lingkungan Dev. Untuk tutorial ini, kami menyarankan Anda membuat Lingkungan Dev menggunakan AWS Cloud9, dan memilih opsi untuk membuat cabang dari cabang utama saat membuat Lingkungan Dev. Kami menggunakan nama **`test`** untuk cabang ini, tetapi Anda dapat memasukkan nama cabang yang berbeda jika Anda mau.

Untuk membuat Lingkungan Dev dengan cabang baru

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek tempat Anda ingin membuat Lingkungan Dev.
3. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, pilih repositori Sumber, dan pilih repositori yang ingin Anda buat Lingkungan Dev.
4. Pada halaman rumah repositori, pilih Create Dev Environment.

5. Pilih IDE yang didukung dari menu tarik-turun. Untuk informasi selengkapnya, lihat Lingkungan pengembangan terintegrasi yang didukung untuk Lingkungan Dev.
6. Pilih repositori untuk dikloning, pilih Bekerja di cabang baru, masukkan nama cabang ke bidang Nama cabang, dan pilih cabang untuk membuat cabang baru dari Buat cabang dari menu drop-down.
7. Secara opsional, tambahkan alias untuk Lingkungan Dev.
8. Secara opsional, pilih tombol edit konfigurasi Lingkungan Dev untuk mengedit konfigurasi komputasi, penyimpanan, atau batas waktu Dev Environment.
9. Pilih Buat. Saat Dev Environment Anda sedang dibuat, kolom status Dev Environment akan menampilkan Mulai, dan kolom status akan ditampilkan Running setelah Dev Environment telah dibuat. Tab baru akan terbuka dengan Lingkungan Dev Anda di IDE pilihan Anda. Anda dapat mengedit kode dan melakukan dan mendorong perubahan Anda.

Buat alur kerja untuk membangun, menguji, dan menerapkan perubahan kode

Di CodeCatalyst, Anda mengatur pembuatan, pengujian, dan penyebaran aplikasi atau layanan Anda dalam alur kerja. Alur kerja terdiri dari tindakan dan dapat dikonfigurasi untuk berjalan secara otomatis setelah peristiwa repositori sumber tertentu terjadi, seperti mendorong kode atau membuka atau memperbarui permintaan tarik. Untuk informasi lebih lanjut tentang alur kerja, lihat Bangun, uji, dan terapkan dengan alur kerja di CodeCatalyst.

Ikuti petunjuk Memulai dengan alur kerja untuk membuat alur kerja pertama Anda.

Undang seseorang ke proyek Anda

Sekarang setelah Anda menyiapkan proyek kustom Anda, undang orang lain untuk bekerja dengan Anda.

Untuk mengundang seseorang ke proyek Anda

1. Arahkan ke proyek yang ingin Anda undang pengguna.
2. Di panel navigasi, pilih Pengaturan proyek.
3. Pada tab Anggota, pilih Undang.

4. Ketik alamat email orang yang ingin Anda undang sebagai pengguna untuk proyek Anda. Anda dapat mengetik beberapa alamat email yang dipisahkan oleh spasi atau koma. Anda juga dapat memilih dari anggota ruang Anda yang bukan anggota proyek.
5. Pilih peran untuk pengguna.

   Setelah selesai menambahkan pengguna, pilih Undang.

Buat masalah untuk berkolaborasi dan melacak pekerjaan

CodeCatalyst membantu Anda melacak fitur, tugas, bug, dan pekerjaan lain yang terlibat dalam proyek Anda dengan masalah. Anda dapat membuat masalah untuk melacak pekerjaan dan ide yang dibutuhkan. Secara default, ketika Anda membuat masalah itu ditambahkan ke backlog Anda. Anda dapat memindahkan masalah ke papan tempat Anda melacak pekerjaan yang sedang berlangsung. Anda juga dapat menetapkan masalah ke anggota proyek tertentu.

Untuk membuat masalah untuk sebuah proyek

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.

   Pastikan Anda menavigasi dalam proyek tempat Anda ingin membuat masalah. Untuk melihat semua proyek, di panel navigasi, pilih Amazon CodeCatalyst, dan jika diperlukan, pilih Lihat semua proyek. Pilih proyek tempat Anda ingin membuat atau bekerja dengan masalah.
2. Di panel navigasi, pilih Lacak, lalu pilih Backlog.
3. Pilih Buat masalah.
4. Di judul Masalah, berikan nama untuk masalah tersebut. Secara opsional memberikan deskripsi masalah. Pilih status, prioritas, dan estimasi untuk masalah jika diinginkan. Anda juga dapat menetapkan masalah ke anggota proyek dari daftar anggota proyek.

> **Tip**
>
> Anda dapat memilih untuk menetapkan masalah ke Amazon Q agar Amazon Q mencoba menyelesaikan masalah tersebut. Jika upaya berhasil, permintaan tarik akan dibuat dan status masalah akan berubah menjadi Dalam peninjauan sehingga Anda dapat meninjau dan menguji kode. Untuk informasi selengkapnya, lihat Tutorial: Menggunakan fitur AI CodeCatalyst generatif untuk mempercepat pekerjaan pengembangan Anda.

Fungsionalitas ini mengharuskan fitur AI generatif diaktifkan untuk ruang tersebut. Untuk informasi selengkapnya, lihat Mengelola fitur AI generatif.

5. Pilih Simpan.

Setelah Anda membuat masalah, Anda dapat menetapkannya ke anggota proyek, memperkirakannya, dan melacaknya di papan Kanban. Untuk informasi selengkapnya, lihat Lacak dan atur pekerjaan dengan masalah di CodeCatalyst.

Tutorial: Menggunakan fitur AI CodeCatalyst generatif untuk mempercepat pekerjaan pengembangan Anda

Jika Anda memiliki proyek dan repositori sumber CodeCatalyst di Amazon di ruang di mana fitur AI generatif diaktifkan, Anda dapat menggunakan fitur ini untuk membantu mempercepat pengembangan perangkat lunak. Pengembang sering memiliki lebih banyak tugas yang harus dilakukan daripada waktu untuk menyelesaikannya. Mereka sering tidak meluangkan waktu untuk menjelaskan perubahan kode mereka kepada rekan tim mereka saat membuat permintaan tarik untuk meninjau perubahan tersebut, mengharapkan pengguna lain menemukan perubahan yang cukup jelas. Pembuat dan pengulas permintaan tarik juga tidak punya waktu untuk menemukan dan membaca semua komentar pada permintaan tarik secara menyeluruh, terutama jika permintaan tarik memiliki beberapa revisi. CodeCatalyst terintegrasi dengan Agen Pengembang Amazon Q untuk pengembangan perangkat lunak guna menyediakan fitur AI generatif yang dapat membantu anggota tim menyelesaikan tugas mereka dengan lebih cepat, dan meningkatkan waktu yang mereka miliki untuk fokus pada bagian terpenting dari pekerjaan mereka.

Amazon Q Developer adalah asisten percakapan bertenaga AI generatif yang dapat membantu Anda memahami, membangun, memperluas, dan mengoperasikan aplikasi. AWS Untuk mempercepat pembangunan Anda AWS, model yang mendukung Amazon Q ditambah dengan AWS konten berkualitas tinggi untuk menghasilkan jawaban yang lebih lengkap, dapat ditindaklanjuti, dan direferensikan. Untuk informasi selengkapnya, lihat Apa itu Pengembang Amazon Q? di Panduan Pengguna Pengembang Amazon Q.

> **Note**
>
> Didukung oleh Amazon Bedrock: AWS mengimplementasikan deteksi penyalahgunaan otomatis. Karena deskripsi Tulis untuk saya, Buat ringkasan konten, dan Tetapkan masalah ke Amazon Q fitur dengan Agen Pengembang Amazon Q untuk pengembangan perangkat

lunak dibangun di Amazon Bedrock, pengguna dapat memanfaatkan sepenuhnya kontrol yang diterapkan di Amazon Bedrock untuk menegakkan keselamatan, keamanan, dan penggunaan kecerdasan buatan (AI) yang bertanggung jawab.

Dalam tutorial ini, Anda akan mempelajari cara menggunakan fitur AI generatif CodeCatalyst untuk membantu Anda meringkas perubahan antar cabang saat membuat permintaan tarik dan meringkas komentar yang tersisa pada permintaan tarik. Anda juga akan belajar cara membuat masalah dengan ide Anda untuk perubahan atau peningkatan kode dan menetapkannya ke Amazon Q. Sebagai bagian dari menangani masalah yang ditetapkan ke Amazon Q, Anda akan belajar cara mengizinkan Amazon Q menyarankan tugas dan cara menetapkan dan mengerjakan tugas apa pun yang dibuatnya sebagai bagian dari mengerjakan suatu masalah.

Prasyarat

Untuk bekerja dengan CodeCatalyst fitur-fitur dalam tutorial ini, Anda harus terlebih dahulu menyelesaikan dan memiliki akses ke sumber daya berikut:

- Anda memiliki ID AWS Pembangun atau identitas masuk tunggal (SSO) untuk masuk. CodeCatalyst
- Proyek Anda berada di ruang yang mengaktifkan fitur AI generatif. Untuk informasi selengkapnya, lihat Mengelola fitur AI generatif.
- Anda memiliki peran administrator Kontributor atau Proyek dalam proyek di ruang tersebut.
- Proyek ini memiliki setidaknya satu repositori sumber yang dikonfigurasi untuk itu. Repositori tertaut tidak didukung.
- Saat menetapkan masalah untuk memiliki solusi awal yang dibuat oleh AI generatif, proyek tidak dapat dikonfigurasi dengan ekstensi Perangkat Lunak Jira. Ekstensi tidak didukung untuk fitur ini.

Untuk informasi lebih lanjut, lihat Menciptakan ruang, Lacak dan atur pekerjaan dengan masalah di CodeCatalyst, Tambahkan fungsionalitas ke proyek dengan ekstensi di CodeCatalyst, dan Memberikan akses dengan peran pengguna .

Tutorial ini didasarkan pada proyek yang dibuat menggunakan cetak biru aplikasi web tiga tingkat Modern dengan Python. Jika Anda menggunakan proyek yang dibuat dengan cetak biru yang berbeda, Anda masih dapat mengikuti langkah-langkahnya, tetapi beberapa spesifik akan bervariasi, seperti kode sampel dan bahasa.

Buat ringkasan perubahan kode antar cabang saat membuat permintaan tarik

Permintaan tarik adalah cara utama Anda dan anggota proyek lainnya dapat meninjau, mengomentari, dan menggabungkan perubahan kode dari satu cabang ke cabang lainnya. Anda dapat menggunakan permintaan tarik untuk meninjau perubahan kode secara kolaboratif untuk perubahan kecil atau perbaikan, penambahan fitur utama, atau versi baru perangkat lunak yang dirilis. Meringkas perubahan kode dan maksud di balik perubahan sebagai bagian dari deskripsi permintaan tarik sangat membantu orang lain yang akan meninjau kode, dan juga membantu dengan pemahaman historis tentang perubahan kode dari waktu ke waktu. Namun, pengembang sering mengandalkan kode mereka untuk menjelaskan dirinya sendiri atau memberikan detail yang ambigu daripada menggambarkan perubahan mereka dengan detail yang cukup bagi pengulas untuk memahami apa yang mereka ulas atau apa maksud di balik perubahan dalam kode.

Anda dapat menggunakan fitur Tulis deskripsi untuk saya saat membuat permintaan tarik agar Amazon Q membuat deskripsi perubahan yang terkandung dalam permintaan tarik. Saat Anda memilih opsi ini, Amazon Q menganalisis perbedaan antara cabang sumber yang berisi perubahan kode dan cabang tujuan tempat Anda ingin menggabungkan perubahan ini. Ini kemudian menciptakan ringkasan tentang apa perubahan itu, serta interpretasi terbaiknya tentang maksud dan efek dari perubahan tersebut.

> **Note**
>
> Fitur ini tidak bekerja dengan submodul Git. Ini tidak akan meringkas perubahan apa pun dalam submodul Git yang merupakan bagian dari permintaan tarik.

Anda dapat mencoba fitur ini dengan permintaan tarik apa pun yang Anda buat, tetapi dalam tutorial ini, kami akan mengujinya dengan membuat beberapa perubahan sederhana pada kode yang terkandung dalam proyek yang dibuat dalam cetak biru aplikasi web tiga tingkat Modern berbasis Python.

> **Tip**
>
> Jika Anda menggunakan proyek yang dibuat dengan cetak biru yang berbeda atau kode Anda sendiri, Anda masih dapat mengikuti tutorial ini, tetapi contoh dalam tutorial ini tidak akan cocok dengan kode dalam proyek Anda. Alih-alih contoh yang disarankan di bawah

ini, buat perubahan sederhana pada kode proyek Anda di cabang, lalu buat permintaan tarik untuk menguji fitur seperti yang ditunjukkan pada langkah-langkah berikut.

Pertama, Anda akan membuat cabang di repositori sumber. Anda kemudian akan membuat perubahan kode cepat ke file di cabang itu menggunakan editor teks di konsol. Anda kemudian akan membuat permintaan tarik, dan menggunakan fitur Write description for me untuk meringkas perubahan yang Anda buat.

Untuk membuat cabang (konsol)

1. Di CodeCatalyst konsol, arahkan ke proyek tempat repositori sumber Anda berada.
2. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
3. Pilih repositori tempat Anda ingin membuat cabang.
4. Pada halaman ikhtisar repositori, pilih Lainnya, lalu pilih Buat cabang.
5. Masukkan nama untuk cabang.
6. Pilih cabang untuk membuat cabang, lalu pilih Buat.

Setelah Anda memiliki cabang, edit file di cabang itu dengan perubahan sederhana. Dalam contoh ini, Anda akan mengedit `test_endpoint.py` file untuk mengubah jumlah percobaan ulang untuk pengujian dari **`3`** 5.

> **Tip**
>
> Anda juga dapat memilih untuk membuat atau menggunakan Dev Environment untuk membuat perubahan kode ini. Untuk informasi selengkapnya, lihat Membuat Lingkungan Dev.

Untuk mengedit **`test_endpoint.py`** file di konsol

1. Pada halaman ikhtisar untuk repositori **`mysfits`** sumber, pilih drop-down cabang dan pilih cabang yang Anda buat dalam prosedur sebelumnya.
2. Di File, navigasikan ke file yang ingin Anda edit. Misalnya, untuk mengedit `test_endpoint.py` file, memperluas tes, memperluas integ, dan kemudian memilih`test_endpoint.py`.

3. Pilih Edit.
4. Pada baris 7, ubah berapa kali semua tes akan dicoba ulang dari:

```
def test_list_all(retry=3):
```

ke:

```
def test_list_all(retry=5):
```

5. Pilih Komit dan komit perubahan Anda ke cabang Anda.

Sekarang setelah Anda memiliki cabang dengan perubahan, Anda dapat membuat permintaan tarik.

Buat permintaan tarik dengan ringkasan perubahan

1. Pada halaman ikhtisar repositori, pilih Lainnya, lalu pilih Buat permintaan tarik.
2. Di cabang Tujuan, pilih cabang untuk menggabungkan kode setelah ditinjau.

> **Tip**
>
> Pilih cabang tempat Anda membuat cabang Anda dari prosedur sebelumnya untuk demonstrasi paling sederhana dari fitur ini. Misalnya, jika Anda membuat cabang dari cabang default repositori, pilih cabang itu sebagai cabang tujuan untuk permintaan tarik Anda.

3. Di cabang Sumber, pilih cabang yang berisi perubahan yang baru saja Anda lakukan pada `test_endpoint.py` file.
4. Dalam judul permintaan tarik, masukkan judul yang membantu pengguna lain memahami apa yang perlu ditinjau dan alasannya.
5. Dalam deskripsi permintaan tarik, pilih Tulis deskripsi agar saya meminta Amazon Q membuat deskripsi perubahan yang terkandung dalam permintaan tarik.
6. Ringkasan perubahan muncul. Tinjau teks yang disarankan lalu pilih Terima dan tambahkan ke deskripsi.
7. Secara opsional memodifikasi ringkasan untuk lebih mencerminkan perubahan yang Anda buat pada kode. Anda juga dapat memilih untuk menambahkan pengulas atau masalah tautan ke permintaan tarik ini. Setelah selesai membuat perubahan tambahan yang Anda inginkan, pilih Buat.

Buat ringkasan komentar yang tersisa pada perubahan kode dalam permintaan tarik

Saat pengguna meninjau permintaan tarik, mereka sering meninggalkan beberapa komentar tentang perubahan permintaan tarik itu. Jika ada banyak komentar dari banyak pengulas, mungkin sulit untuk memilih tema umum dalam umpan balik, atau bahkan pastikan bahwa Anda telah meninjau semua komentar di semua revisi. Anda dapat menggunakan fitur Buat ringkasan komentar agar Amazon Q menganalisis semua komentar yang tersisa pada perubahan kode dalam permintaan tarik dan membuat ringkasan komentar tersebut.

> **Note**
>
> Ringkasan komentar bersifat sementara. Jika Anda menyegarkan permintaan tarik, ringkasan akan hilang. Ringkasan konten tidak menyertakan komentar pada permintaan tarik keseluruhan, cukup komentar yang tersisa tentang perbedaan kode dalam revisi permintaan tarik.
> Fitur ini tidak berfungsi dengan komentar apa pun yang tersisa pada perubahan kode di submodul Git.

Untuk membuat ringkasan komentar dalam permintaan tarik

1. Arahkan ke permintaan tarik yang Anda buat di prosedur sebelumnya.

   > **Tip**
   >
   > Jika mau, Anda dapat menggunakan permintaan tarik terbuka apa pun di proyek Anda. Di bilah navigasi, pilih Kode, pilih Permintaan tarik, dan pilih permintaan tarik terbuka.

2. Tambahkan beberapa komentar ke permintaan tarik di Perubahan jika permintaan tarik belum memiliki komentar.
3. Di Ikhtisar, pilih Buat ringkasan komentar. Setelah selesai, bagian Ringkasan komentar akan diperluas.
4. Tinjau ringkasan komentar yang tersisa pada perubahan kode dalam revisi permintaan tarik, dan bandingkan dengan komentar dalam permintaan tarik.

Buat masalah dan tetapkan ke Amazon Q

Tim pengembangan membuat masalah untuk melacak dan mengelola pekerjaan mereka, tetapi terkadang masalah tetap ada karena tidak jelas siapa yang harus mengerjakannya, atau masalah tersebut memerlukan penelitian ke bagian tertentu dari basis kode, atau pekerjaan mendesak lainnya harus dihadiri terlebih dahulu. CodeCatalyst termasuk integrasi dengan Amazon Q Developer Agent untuk pengembangan perangkat lunak. Anda dapat menetapkan masalah ke asisten AI generatif yang disebut Amazon Q yang dapat menganalisis masalah berdasarkan judul dan deskripsinya. Jika Anda menetapkan masalah ke Amazon Q, itu akan mencoba membuat solusi draf untuk Anda evaluasi. Ini dapat membantu Anda dan tim Anda untuk fokus dan mengoptimalkan pekerjaan Anda pada masalah yang memerlukan perhatian Anda, sementara Amazon Q bekerja pada solusi untuk masalah yang tidak memiliki sumber daya untuk segera ditangani.

> **Tip**
>
> Amazon Q berkinerja terbaik pada masalah sederhana dan masalah langsung. Untuk hasil terbaik, gunakan bahasa sederhana untuk menjelaskan dengan jelas apa yang ingin Anda lakukan.

Ketika Anda menetapkan masalah ke Amazon Q, CodeCatalyst akan menandai masalah sebagai diblokir sampai Anda mengonfirmasi bagaimana Anda ingin Amazon Q bekerja pada masalah tersebut. Ini mengharuskan Anda untuk menjawab tiga pertanyaan sebelum dapat melanjutkan:

- Apakah Anda ingin mengonfirmasi setiap langkah yang diperlukan atau apakah Anda ingin melanjutkan tanpa umpan balik. Jika Anda memilih untuk mengonfirmasi setiap langkah, Anda dapat membalas Amazon Q dengan umpan balik tentang pendekatan yang dibuatnya sehingga dapat mengulangi pendekatannya jika diperlukan. Amazon Q juga dapat meninjau umpan balik yang ditinggalkan pengguna pada permintaan tarik apa pun yang dibuatnya jika Anda memilih opsi ini. Jika Anda memilih untuk tidak mengonfirmasi setiap langkah, Amazon Q mungkin menyelesaikan pekerjaannya lebih cepat, tetapi Amazon Q tidak akan meninjau umpan balik apa pun yang Anda berikan dalam masalah atau permintaan tarik apa pun yang dibuatnya.
- Apakah Anda ingin mengizinkannya memperbarui file alur kerja sebagai bagian dari pekerjaannya. Proyek Anda mungkin memiliki alur kerja yang dikonfigurasi untuk mulai berjalan pada peristiwa permintaan tarik. Jika demikian, permintaan tarik apa pun yang dibuat Amazon Q yang menyertakan pembuatan atau pembaruan alur kerja YAMG dapat memulai menjalankan alur kerja yang disertakan dalam permintaan tarik. Sebagai praktik terbaik, jangan memilih untuk

mengizinkan Amazon Q bekerja pada file alur kerja kecuali Anda yakin tidak ada alur kerja dalam proyek Anda yang akan secara otomatis menjalankan alur kerja ini sebelum Anda meninjau dan menyetujui permintaan tarik yang dibuatnya.

- Apakah Anda ingin mengizinkannya menyarankan membuat tugas untuk memecah pekerjaan dalam masalah menjadi peningkatan yang lebih kecil yang dapat ditetapkan secara individual kepada pengguna, termasuk Amazon Q itu sendiri. Mengizinkan Amazon Q menyarankan dan membuat tugas dapat membantu mempercepat pengembangan pada masalah kompleks dengan memungkinkan banyak orang mengerjakan bagian terpisah dari masalah tersebut. Ini juga dapat membantu mengurangi kompleksitas pemahaman keseluruhan pekerjaan karena pekerjaan yang diperlukan untuk menyelesaikan setiap tugas idealnya lebih sederhana daripada masalah yang dimilikinya.
- Repositori sumber apa yang Anda inginkan untuk berfungsi. Bahkan jika proyek Anda memiliki beberapa repositori sumber, Amazon Q hanya dapat bekerja pada kode dalam satu repositori sumber. Repositori tertaut tidak didukung.

Setelah Anda membuat dan mengonfirmasi pilihan Anda, Amazon Q akan memindahkan masalah ke status Sedang berlangsung saat mencoba menentukan permintaan berdasarkan judul masalah dan deskripsinya, serta kode di repositori yang ditentukan. Ini akan membuat komentar yang disematkan di mana ia akan memberikan pembaruan tentang status pekerjaannya. Setelah meninjau data, Amazon Q akan merumuskan pendekatan potensial untuk solusi. Amazon Q mencatat tindakannya dengan memperbarui komentar yang disematkan dan mengomentari kemajuannya pada masalah ini di setiap tahap. Tidak seperti komentar dan balasan yang disematkan, itu tidak menyimpan catatan kronologis pekerjaannya secara ketat. Sebaliknya, ini menempatkan informasi yang paling relevan tentang pekerjaannya di tingkat atas komentar yang disematkan. Ini akan mencoba membuat kode berdasarkan pendekatannya dan analisisnya terhadap kode yang sudah ada di repositori. Jika berhasil menghasilkan solusi potensial, itu akan membuat cabang dan kode komit ke cabang itu. Kemudian membuat permintaan tarik yang akan menggabungkan cabang itu dengan cabang default. Ketika Amazon Q menyelesaikan pekerjaannya, ia memindahkan masalah ke Dalam tinjauan sehingga Anda dan tim Anda tahu ada kode yang siap untuk Anda evaluasi.

> **Note**
>
> Fitur ini hanya tersedia melalui Masalah di Wilayah Barat AS (Oregon). Ini tidak tersedia jika Anda telah mengonfigurasi proyek Anda untuk menggunakan Jira dengan ekstensi Perangkat Lunak Jira. Selain itu, jika Anda telah menyesuaikan tata letak papan Anda, masalah mungkin

tidak mengubah status. Untuk hasil terbaik, hanya gunakan fitur ini dengan proyek yang memiliki tata letak papan standar.
Fitur ini tidak bekerja dengan submodul Git. Itu tidak dapat membuat perubahan pada submodul Git apa pun yang termasuk dalam repositori.
Setelah Anda menetapkan masalah ke Amazon Q, Anda tidak dapat mengubah judul atau deskripsi masalah atau menetapkannya kepada orang lain. Jika Anda membatalkan penetapan Amazon Q dari masalah, itu akan menyelesaikan langkahnya saat ini dan kemudian berhenti bekerja. Itu tidak dapat melanjutkan pekerjaan atau dipindahkan ke masalah setelah tidak ditetapkan.
Masalah dapat secara otomatis dipindahkan ke kolom Dalam tinjauan jika ditetapkan ke Amazon Q jika pengguna memilih untuk mengizinkannya membuat tugas. Namun, masalah di Dalam tinjauan mungkin masih memiliki tugas yang berada dalam keadaan berbeda, seperti dalam status Sedang berlangsung.

Di bagian tutorial ini, Anda akan membuat tiga masalah berdasarkan fitur potensial untuk kode yang disertakan dalam proyek yang dibuat dengan cetak biru aplikasi web tiga tingkat Modern: satu untuk menambahkan untuk membuat makhluk mysfit baru, satu untuk menambahkan fitur pengurutan, dan satu untuk memperbarui alur kerja untuk menyertakan cabang bernama. **`test`**

> **Note**
>
> Jika Anda bekerja dalam proyek dengan kode yang berbeda, buat masalah dengan judul dan deskripsi yang berhubungan dengan basis kode tersebut.

Untuk membuat masalah dan memiliki solusi yang dihasilkan untuk Anda evaluasi

1. Di panel navigasi, pilih Masalah dan pastikan Anda berada di tampilan Papan.
2. Pilih Buat masalah.
3. Berikan judul yang menjelaskan apa yang ingin Anda lakukan dalam bahasa sederhana. Misalnya, untuk masalah ini, masukkan judul**`Create another mysfit named Quokkapus`**. Dalam Deskripsi, berikan detail berikut:

```
Expand the table of mysfits to 13, and give the new mysfit the following
 characteristics:

Name: Quokkapus
```

```
Species: Quokka-Octopus hybrid

Good/Evil: Good

Lawful/Chaotic: Chaotic

Age: 216

Description: Australia is full of amazing marsupials, but there's nothing there
 quite like the Quokkapus.
She's always got a friendly smile on her face, especially when she's using her
 eight limbs to wrap you up
in a great big hug. She exists on a diet of code bugs and caffeine. If you've got
 some gnarly code that needsa
assistance, adopt Quokkapus and put her to work - she'll love it! Just make sure
 you leave enough room for
her to grow, and keep that coffee coming.
```

4. (Opsional) Lampirkan gambar untuk digunakan sebagai thumbnail dan gambar profil untuk mysfit ke masalah tersebut. Jika Anda melakukan ini, perbarui deskripsi untuk menyertakan detail gambar apa yang ingin Anda gunakan dan mengapa. Misalnya, Anda dapat menambahkan yang berikut ke deskripsi: “Mysfit membutuhkan file gambar untuk disebarkan ke situs web. Tambahkan gambar-gambar ini yang dilampirkan pada masalah ini ke repositori sumber sebagai bagian dari pekerjaan, dan menyebarkan gambar ke situs web.

   > **Note**
   >
   > Gambar terlampir mungkin atau mungkin tidak disebarkan ke situs web selama interaksi dalam tutorial ini. Anda dapat menambahkan gambar ke situs web sendiri, dan kemudian meninggalkan komentar untuk Amazon Q untuk memperbarui kodenya untuk menunjuk ke gambar yang ingin Anda gunakan setelah membuat permintaan tarik.

   Tinjau deskripsi dan pastikan berisi semua detail yang mungkin diperlukan sebelum Anda melanjutkan ke langkah berikutnya.
5. Di Penerima Tugas, pilih Tetapkan ke Amazon Q.
6. Di repositori Sumber, pilih repositori sumber yang berisi kode proyek.
7. Geser Require Amazon Q untuk berhenti setelah setiap langkah dan tunggu peninjauan pemilih kerjanya ke status aktif jika perlu.

> **Note**
>
> Memilih opsi untuk menghentikan Amazon Q setelah setiap langkah memungkinkan Anda mengomentari masalah atau tugas apa pun yang dibuat untuk memiliki opsi agar Amazon Q mengubah pendekatannya hingga tiga kali berdasarkan komentar Anda. Jika Anda memilih opsi untuk tidak menghentikan Amazon Q setelah setiap langkah sehingga Anda dapat meninjau pekerjaannya, pekerjaan mungkin berjalan lebih cepat karena Amazon Q tidak menunggu umpan balik Anda, tetapi Anda tidak akan dapat memengaruhi arah Amazon Q mengambil dengan meninggalkan komentar. Amazon Q juga tidak akan menanggapi komentar yang tersisa dalam permintaan tarik jika Anda memilih opsi itu.

8. Biarkan Izinkan Amazon Q untuk memodifikasi pemilih file alur kerja dalam keadaan tidak aktif.
9. Geser Izinkan Amazon Q untuk menyarankan membuat pemilih tugas ke status aktif.
10. Pilih Buat masalah. Tampilan Anda berubah ke papan Masalah.
11. Pilih Buat masalah untuk membuat masalah lain, kali ini dengan judul**`Change the get_all_mysfits() API to return mysfits sorted by the Age attribute`**. Tetapkan masalah ini ke Amazon Q dan buat masalah.
12. Pilih Buat masalah untuk membuat masalah lain, kali ini dengan judul**`Update the OnPullRequest workflow to include a branch named test in its triggers`**. Secara opsional link ke alur kerja dalam deskripsi. Tetapkan masalah ini ke Amazon Q tetapi kali ini pastikan bahwa pemilih Izinkan Amazon Q memodifikasi file alur kerja disetel ke status aktif. Buat masalah untuk kembali ke papan Masalah.

> **Tip**
>
> Anda dapat mencari file, termasuk file alur kerja, dengan memasukkan simbol at (@) dan memasukkan nama file.

Setelah Anda membuat dan menetapkan masalah, masalah akan pindah ke Sedang berlangsung. Amazon Q akan menambahkan komentar yang melacak kemajuannya di dalam masalah dalam komentar yang disematkan. Jika mampu mendefinisikan pendekatan untuk solusi, itu akan memperbarui deskripsi masalah dengan bagian Latar Belakang yang berisi analisis basis kode dan bagian Pendekatan yang merinci pendekatan yang diusulkan untuk membuat solusi. Jika Amazon

Q berhasil menemukan solusi untuk masalah yang dijelaskan dalam masalah ini, Amazon Q akan membuat cabang dan perubahan kode di cabang yang mengimplementasikan solusi yang diusulkan. Jika kode yang diusulkan berisi kesamaan dengan kode sumber terbuka yang diketahui Amazon Q, itu akan menyediakan file yang menyertakan tautan ke kode itu sehingga Anda dapat memeriksanya. Setelah kode siap, ia membuat permintaan tarik sehingga Anda dapat meninjau perubahan kode yang disarankan, menambahkan tautan ke permintaan tarik itu ke masalah, dan memindahkan masalah ke Dalam tinjauan.

> **Important**
>
> Anda harus selalu meninjau setiap perubahan kode dalam permintaan tarik sebelum menggabungkannya. Menggabungkan perubahan kode yang dibuat oleh Amazon Q, seperti perubahan kode lainnya, dapat berdampak negatif pada basis kode dan kode infrastruktur Anda jika kode gabungan tidak ditinjau dengan benar dan berisi kesalahan saat digabungkan.

Untuk meninjau masalah dan permintaan tarik tertaut yang berisi perubahan yang dibuat oleh Amazon Q

1. Dalam Masalah, pilih masalah yang ditetapkan ke Amazon Q yang sedang berlangsung. Tinjau komentar untuk memantau kemajuan Amazon Q. Jika ada, tinjau latar belakang dan dekati catatan dalam deskripsi masalah. Jika Anda memilih untuk mengizinkan Amazon Q menyarankan tugas, tinjau tugas apa pun yang diusulkan dan lakukan tindakan apa pun yang diperlukan. Misalnya, jika Amazon Q menyarankan tugas dan Anda ingin mengubah urutan atau menetapkan tugas ke pengguna tertentu, pilih Ubah, tambahkan, atau susun ulang tugas dan lakukan pembaruan apa pun yang diperlukan. Setelah selesai melihat masalah, pilih X untuk menutup panel masalah.

   > **Tip**
   >
   > Untuk melihat kemajuan tugas, pilih tugas dari daftar tugas dalam masalah. Tugas tidak muncul sebagai item terpisah di papan tulis dan hanya dapat diakses melalui masalah. Jika tugas ditetapkan ke Amazon Q, Anda harus membuka tugas untuk menyetujui tindakan apa pun yang ingin dilakukannya. Anda juga harus membuka tugas untuk melihat permintaan tarik tertaut karena tidak akan muncul sebagai tautan dalam masalah, hanya dalam tugas. Untuk kembali ke masalah dari tugas, pilih tautan ke masalah.

2. Sekarang pilih masalah yang ditetapkan ke Amazon Q yang ada di Dalam ulasan. Tinjau latar belakang dan dekati catatan dalam deskripsi masalah. Tinjau komentar untuk memahami tindakan yang dilakukannya. Tinjau tugas apa pun yang dibuat untuk pekerjaan yang terkait dengan masalah ini, termasuk kemajuannya, tindakan apa pun yang mungkin perlu Anda ambil, dan komentar apa pun. Di Permintaan tarik, pilih tautan ke permintaan tarik di sebelah label Buka untuk meninjau kode.

   > **Tip**
   >
   > Permintaan tarik yang dihasilkan untuk tugas hanya muncul sebagai permintaan tarik tertaut dalam tampilan tugas. Mereka tidak muncul sebagai permintaan tarik tertaut untuk masalah ini.

3. Dalam permintaan tarik, tinjau perubahan kode. Untuk informasi selengkapnya, lihat Meninjau permintaan tarik. Tinggalkan komentar pada permintaan tarik jika Anda ingin Amazon Q mengubah kode yang disarankan. Bersikaplah spesifik saat meninggalkan komentar untuk Amazon Q untuk hasil terbaik.

   Misalnya, saat meninjau permintaan tarik yang dibuat untuk**`Create another mysfit named Quokkapus`**, Anda mungkin memperhatikan bahwa ada kesalahan ketik dalam deskripsi. Anda dapat meninggalkan komentar untuk Amazon Q yang menyatakan “Ubah deskripsi untuk memperbaiki kesalahan ketik” needsa “dengan menambahkan spasi antara “kebutuhan” dan “a”.” Atau, Anda dapat meninggalkan komentar yang memberi tahu Amazon Q untuk memperbarui deskripsi dan memberikan seluruh deskripsi yang direvisi untuk digabungkan.

   Jika Anda mengunggah gambar untuk mysfit baru ke situs web, Anda dapat meninggalkan komentar untuk Amazon Q untuk memperbarui mysfit dengan pointer ke gambar dan thumbnail yang akan digunakan untuk mysfit baru.

   > **Note**
   >
   > Amazon Q tidak akan menanggapi komentar individual. Amazon Q hanya akan memasukkan umpan balik yang tersisa di komentar dalam permintaan tarik jika Anda memilih opsi default untuk berhenti setelah setiap langkah untuk persetujuan saat Anda membuat masalah.

4. (Opsional) Setelah Anda dan pengguna proyek lainnya meninggalkan semua komentar yang Anda inginkan untuk perubahan kode, pilih Buat revisi agar Amazon Q membuat revisi

permintaan tarik yang menggabungkan perubahan yang Anda minta dalam komentar. Kemajuan kemajuan pembuatan revisi akan dilaporkan oleh Amazon Q di Ikhtisar, bukan dalam Perubahan. Pastikan untuk menyegarkan browser Anda untuk melihat pembaruan terbaru dari Amazon Q tentang membuat revisi.

> **Note**
>
> Hanya pengguna yang membuat masalah yang dapat membuat revisi permintaan tarik. Anda hanya dapat meminta satu revisi permintaan tarik. Pastikan bahwa Anda telah mengatasi semua masalah dengan komentar, dan bahwa Anda puas dengan konten komentar, sebelum Anda memilih Buat revisi.

5. Alur kerja dijalankan untuk setiap permintaan tarik dalam proyek contoh ini. Pastikan bahwa Anda melihat alur kerja yang berhasil dijalankan sebelum Anda menggabungkan permintaan tarik. Anda juga dapat memilih untuk membuat alur kerja dan lingkungan tambahan untuk menguji kode sebelum Anda menggabungkannya. Untuk informasi selengkapnya, lihat Memulai dengan alur kerja.
6. Bila Anda puas dengan revisi terbaru dari pull request, pilih Merge.

Pembersihan sumber daya

Setelah Anda menyelesaikan tutorial ini, pertimbangkan untuk mengambil tindakan berikut untuk membersihkan sumber daya apa pun yang Anda buat selama tutorial ini yang tidak lagi Anda butuhkan.

- Batalkan penugasan Amazon Q dari masalah apa pun yang tidak lagi dikerjakan. Jika Amazon Q telah menyelesaikan pekerjaannya pada suatu masalah atau tidak dapat menemukan solusi, pastikan untuk membatalkan penetapan Amazon Q untuk menghindari mencapai kuota maksimum untuk fitur AI generatif. Untuk informasi selengkapnya, lihat Mengelola fitur dan Harga AI generatif.
- Pindahkan masalah apa pun di mana pekerjaan selesai ke Selesai.
- Jika proyek tidak lagi diperlukan, hapus proyek.

Tutorial: Membuat aplikasi full-stack dengan cetak biru PDK yang dapat dikomposisi

Amazon CodeCatalyst menyediakan sejumlah cetak biru yang berbeda untuk membantu Anda memulai proyek Anda dengan cepat. Proyek yang dibuat dengan cetak biru mencakup sumber daya yang Anda butuhkan, termasuk repositori sumber, kode sumber sampel, alur kerja CI/CD, laporan pembuatan dan pengujian, dan alat pelacakan masalah terintegrasi. Namun, terkadang Anda mungkin ingin secara bertahap membangun proyek, atau menambahkan fungsionalitas ke proyek yang ada yang dibuat oleh cetak biru. Anda juga dapat melakukan ini dengan cetak biru. Tutorial ini menunjukkan bagaimana Anda dapat memulai dengan cetak biru tunggal yang menetapkan dasar dan memungkinkan Anda untuk menyimpan semua kode proyek Anda dalam satu repositori. Dari sana, Anda memiliki fleksibilitas untuk menggabungkan sumber daya dan infrastruktur tambahan dengan menerapkan cetak biru lain di atas cetak biru awal sesuai keinginan Anda. Melalui metode building-block ini, Anda dapat mengatasi persyaratan spesifik di beberapa proyek.

Tutorial ini menunjukkan cara menyusun beberapa cetak biru AWS Project Development Kit (AWS PDK) bersama-sama untuk membuat aplikasi yang terdiri dari situs web React, Smithy API, dan infrastruktur CDK pendukung untuk menerapkannya ke AWS. AWS PDK menyediakan blok bangunan untuk pola umum bersama dengan alat pengembangan untuk mengelola dan membangun proyek Anda. Untuk informasi selengkapnya, lihat repositori GitHub sumber AWS PDK.

Cetak biru PDK berikut dirancang untuk digunakan satu sama lain untuk membangun aplikasi dengan cara yang dapat dikomposisi:

- Monorepo - Membuat proyek tingkat akar yang mengelola saling ketergantungan antar proyek dalam monorepo. Proyek ini juga menyediakan build caching dan visualisasi ketergantungan.
- Type Safe API - Membuat API yang dapat didefinisikan dalam Smithy atau OpenAPI v3, dan mengelola pembuatan kode waktu pembuatan untuk memungkinkan Anda mengimplementasikan dan berinteraksi dengan API Anda dengan cara yang aman tipe. Menjual konstruksi CDK yang mengelola penerapan API Anda ke API Gateway dan mengonfigurasi validasi input otomatis.
- Situs web Cloudscape React - Membuat situs web berbasis React yang dibuat menggunakan Cloudscape yang sudah terintegrasi dengan Cognito Auth dan (opsional) API yang Anda buat, yang memberi Anda kemampuan untuk memanggil API Anda dengan aman.
- Infrastruktur - Membuat proyek yang menyiapkan semua infrastruktur terkait CDK yang diperlukan untuk menyebarkan aplikasi Anda. Itu juga datang pra-konfigurasi untuk menghasilkan diagram berdasarkan kode CDK Anda setiap kali Anda membangun.

- DevOps- Membuat DevOps alur kerja yang kompatibel dengan konstruksi yang ditemukan di AWS Project Development Kit (AWS PDK).

Tutorial ini juga mencakup langkah-langkah tentang cara melihat aplikasi yang diterapkan, mengundang pengguna lain untuk mengerjakannya, dan membuat perubahan pada kode dengan permintaan tarik yang secara otomatis dibuat dan digunakan ke sumber daya di akun AWS yang terhubung saat permintaan tarik digabungkan.

Saat Anda membuat proyek yang terdiri dari cetak biru PDK, proyek Anda dibuat dengan sumber daya berikut dalam proyek: CodeCatalyst

- Sebuah repositori sumber dikonfigurasi sebagai monorepo.
- Alur kerja yang menjalankan analisis kode statis dan pemeriksaan lisensi, serta membangun dan menerapkan kode sampel setiap kali perubahan dilakukan ke cabang default. Diagram arsitektur dihasilkan setiap kali Anda membuat perubahan pada kode.
- Papan masalah dan backlog yang dapat Anda gunakan untuk merencanakan dan melacak pekerjaan.
- Rangkaian laporan pengujian dengan laporan otomatis.

Topik

Prasyarat

Untuk membuat dan memperbarui proyek, Anda harus menyelesaikan tugas Siapkan dan masuk ke CodeCatalyst sebagai berikut:

- Memiliki ID AWS Pembangun untuk masuk CodeCatalyst.

- Milik ruang dan memiliki administrator Space atau peran pengguna Power yang ditetapkan untuk Anda di ruang itu. Lihat informasi selengkapnya di Menciptakan ruang, Memberikan izin ruang kepada pengguna, dan Peran administrator ruang.
- Miliki akun AWS yang terkait dengan ruang Anda dan miliki peran IAM yang Anda buat saat mendaftar. Misalnya, saat mendaftar, Anda memiliki opsi untuk memilih membuat peran layanan dengan kebijakan peran yang disebut kebijakan peran CodeCatalystWorkflowDevelopmentRole- ***SpaceName***. Peran akan memiliki nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* dengan pengenal unik ditambahkan. Untuk informasi selengkapnya tentang kebijakan peran dan peran, lihatMemahami peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan. Untuk langkah-langkah untuk membuat peran, lihatMembuat CodeCatalystWorkflowDevelopmentRole-***spaceName***peran untuk akun dan ruang Anda.

Langkah 1: Buat proyek monorepo

Mulailah dengan cetak biru PDK - Monorepo untuk membuat basis kode monorepo Anda yang bertindak sebagai fondasi, memungkinkan Anda menambahkan cetak biru PDK tambahan.

Untuk membuat proyek menggunakan cetak biru PDK - Monorepo

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, arahkan ke ruang tempat Anda ingin membuat proyek.
3. Di dasbor ruang, pilih Buat proyek.
4. Pilih Mulai dengan cetak biru.
5. Pilih cetak biru PDK - Monorepo, lalu pilih Berikutnya.
6. Di bawah Nama proyek Anda, masukkan nama yang ingin Anda tetapkan ke proyek Anda dan nama sumber daya yang terkait. Nama harus unik di dalam ruang Anda.
7. Di bawah sumber daya Proyek, lakukan hal berikut:

    a. Di bawah Bahasa pemrograman utama, pilih bahasa yang ingin Anda kembangkan kode proyek Anda. Anda dapat memilih dari TypeScript, Java, atau Python.

    b. Pilih Konfigurasi Kode

    c. Di bidang input teks Source Repository, masukkan nama repositori sumber, yang akan membuat repositori baru, atau pilih dari repositori tertaut yang ada. Repositori yang ada harus kosong. Untuk informasi selengkapnya, lihat Menautkan repositori sumber.

d. (Opsional) Dari menu tarik-turun Package Manager, pilih manajer paket. Ini hanya diperlukan jika Anda memilih TypeScript sebagai bahasa pemrograman utama Anda.

8. (Opsional) Untuk melihat pratinjau kode yang akan dihasilkan berdasarkan pilihan parameter proyek yang Anda buat, pilih Lihat kode dari Hasilkan pratinjau proyek.
9. (Opsional) Pilih Lihat detail dari kartu cetak biru untuk melihat detail spesifik tentang cetak biru, seperti ikhtisar arsitektur cetak biru, koneksi dan izin yang diperlukan, dan jenis sumber daya yang dibuat cetak biru.
10. Pilih Buat proyek untuk membuat proyek monorepo Anda. Proyek tingkat root yang dibuat mengelola saling ketergantungan antar proyek dalam monorepo, serta menyediakan caching build dan manajemen ketergantungan.

    Untuk informasi selengkapnya tentang cetak biru proyek, lihat. Membuat proyek yang komprehensif dengan CodeCatalyst cetak biru

Cetak biru PDK - Monorepo hanya menghasilkan fondasi proyek. Untuk membuat aplikasi yang bisa diterapkan menggunakan cetak biru, Anda perlu menambahkan cetak biru PDK lainnya seperti Type Safe API, Cloudscape React Website, Infrastructure, atau. DevOps Pada langkah berikutnya, Anda akan menerapkan Type Safe API ke proyek.

Langkah 2: Tambahkan Type Safe API ke proyek

Blueprint PDK - Type Safe API memungkinkan Anda mendefinisikan API menggunakan Smithy atau OpenAI v3. Ini menghasilkan paket runtime dari definisi API Anda, yang mencakup klien untuk berinteraksi dengan API dan kode sisi server untuk mengimplementasikan API Anda. Cetak biru ini juga menghasilkan konstruksi CDK dengan keamanan tipe untuk setiap operasi API. Anda dapat menerapkan cetak biru ke proyek monorepo PDK yang ada untuk menambahkan kemampuan API ke proyek.

Untuk menerapkan cetak biru PDK - Type Safe API

1. Di panel navigasi proyek monorepo Anda, pilih Blueprints, lalu pilih Terapkan cetak biru.
2. Pilih blueprint PDK - Type Safe API, lalu pilih Next.
3. Di bawah Konfigurasi cetak biru, konfigurasikan parameter cetak biru:

   - Di bawah Model Language, pilih bahasa yang didefinisikan dalam model API.
   - Di bidang input teks Namespace, masukkan namespace untuk API Anda.

- Di bidang input teks nama API, masukkan nama untuk API Anda.
- Di bawah bahasa CDK, pilih bahasa pilihan Anda untuk menulis infrastruktur CDK untuk menerapkan API.
- Pilih menu tarik-turun bahasa Handler, lalu pilih bahasa yang ingin Anda terapkan penangan untuk operasi API.
- Pilih menu tarik-turun format Dokumentasi, lalu pilih format yang Anda inginkan untuk membuat dokumentasi API.

4. Di tab Perubahan kode, tinjau perubahan yang diusulkan. Perbedaan yang ditampilkan dalam permintaan tarik menunjukkan perubahan pada proyek Anda pada saat permintaan tarik dibuat.
5. Jika Anda puas dengan perubahan yang diusulkan yang akan dibuat saat cetak biru diterapkan, pilih Terapkan cetak biru.

   Setelah permintaan tarik dibuat, Anda dapat menambahkan komentar. Komentar dapat ditambahkan ke permintaan tarik atau ke baris individual dalam file serta permintaan tarik keseluruhan. Anda dapat menambahkan tautan ke sumber daya seperti file dengan menggunakan `@` tanda, diikuti dengan nama file.

   > **Note**
   >
   > Cetak biru tidak akan diterapkan sampai permintaan tarik disetujui dan digabungkan. Untuk informasi selengkapnya, lihat Meninjau permintaan tarik dan Menggabungkan permintaan tarik.

6. Dari kolom Status, pilih Permintaan tarik tertunda untuk baris cetak biru PDK - Type Safe API, lalu pilih tautan permintaan tarik terbuka.
7. Pilih Gabung, pilih strategi penggabungan pilihan Anda, lalu pilih Gabung untuk menggabungkan perubahan dari cetak biru yang diterapkan.

   Setelah permintaan tarik digabungkan, `packages/apis/mypdkapi` folder baru dibuat dalam proyek monorepo Anda, yang berisi semua kode sumber terkait API untuk Type Safe API yang dikonfigurasi.
8. Di panel navigasi, pilih Blueprints untuk mengonfirmasi bahwa Status PDK - Type Safe API ditampilkan Terbaru.

Langkah 3: Tambahkan Situs Web Cloudscape React untuk proyek

Cetak biru PDK - Cloudscape React Website menghasilkan situs web. Anda dapat mengaitkan parameter opsional (Type Safe API) untuk secara otomatis mengonfigurasi situs web Anda untuk menyiapkan klien aman tipe terautentikasi bersama dengan penjelajah API interaktif untuk menguji berbagai API Anda.

Untuk menerapkan cetak biru PDK - Cloudscape React Website

1. Di panel navigasi proyek monorepo Anda, pilih Blueprints, lalu pilih Terapkan cetak biru.
2. Pilih blueprint PDK - Cloudscape React Website, lalu pilih Next.
3. Di bawah Konfigurasi cetak biru, konfigurasikan parameter cetak biru:
   - Di kolom input teks nama situs web, masukkan nama untuk situs web Anda.
   - Pilih menu tarik-turun Type Safe API, lalu pilih cetak biru API yang ingin Anda integrasikan dalam situs web. Melewati API akan menyiapkan klien yang diautentikasi dan menambahkan dependensi yang diperlukan, penjelajah API, dan kemampuan lainnya.
4. Di tab Perubahan kode, tinjau perubahan yang diusulkan. Perbedaan yang ditampilkan dalam permintaan tarik menunjukkan perubahan pada proyek Anda pada saat permintaan tarik dibuat.
5. Jika Anda puas dengan perubahan yang diusulkan yang akan dibuat saat cetak biru diterapkan, pilih Terapkan cetak biru.

   Setelah permintaan tarik dibuat, Anda dapat menambahkan komentar. Komentar dapat ditambahkan ke permintaan tarik atau ke baris individual dalam file serta permintaan tarik keseluruhan. Anda dapat menambahkan tautan ke sumber daya seperti file dengan menggunakan @ tanda, diikuti dengan nama file.

   > **Note**
   >
   > Cetak biru tidak akan diterapkan sampai permintaan tarik disetujui dan digabungkan. Untuk informasi selengkapnya, lihat Meninjau permintaan tarik dan Menggabungkan permintaan tarik.

6. Dari kolom Status, pilih Permintaan tarik tertunda untuk baris cetak biru PDK - Cloudscape React Website, lalu pilih tautan permintaan tarik terbuka.
7. Pilih Gabung, pilih strategi penggabungan pilihan Anda, lalu pilih Gabung untuk menggabungkan perubahan dari cetak biru yang diterapkan.

Setelah permintaan tarik digabungkan, `packages/websites/my-website-name` folder baru dihasilkan dalam proyek monorepo Anda, yang berisi semua kode sumber untuk situs web baru Anda.

8. Di panel navigasi, pilih Blueprints untuk mengonfirmasi bahwa Status Situs Web PDK - Cloudscape React muncul Terbaru.

Selanjutnya, Anda akan menerapkan cetak biru PDK - Infrastructure untuk menghasilkan infrastruktur untuk menyebarkan situs web Anda ke AWS cloud.

Langkah 4: Hasilkan infrastruktur untuk menyebarkan aplikasi ke AWS cloud

Cetak biru PDK - Infrastruktur menyiapkan paket yang berisi semua kode CDK Anda untuk menyebarkan situs web dan API Anda. Ini juga menyediakan pembuatan diagram dan kesesuaian dengan prototyping nag pack secara default.

Untuk menerapkan cetak biru PDK - Infrastruktur

1. Di panel navigasi proyek monorepo Anda, pilih Blueprints, lalu pilih Terapkan cetak biru.
2. Pilih PDK - Infrastructure blueprint, lalu pilih Next.
3. Di bawah Konfigurasi cetak biru, konfigurasikan parameter cetak biru:

   - Di bawah bahasa CDK, pilih bahasa yang Anda inginkan untuk mengembangkan infrastruktur Anda.
   - Di bidang input teks nama Stack, masukkan nama CloudFormation tumpukan yang dihasilkan untuk cetak biru Anda.

     > **Note**
     >
     > Catat nama tumpukan ini untuk langkah berikutnya, di mana Anda akan mengatur DevOps alur kerja.

   - Pilih menu tarik-turun Type Safe API, lalu pilih cetak biru API yang ingin Anda integrasikan dalam situs web.
   - Pilih menu dropdown Cloudscape React TS Websites, lalu pilih cetak biru situs web yang ingin Anda terapkan dalam infrastruktur Anda (misalnya, PDK - Cloudscape React Website).

4. Di tab Perubahan kode, tinjau perubahan yang diusulkan. Perbedaan yang ditampilkan dalam permintaan tarik menunjukkan perubahan pada proyek Anda pada saat permintaan tarik dibuat.
5. Jika Anda puas dengan perubahan yang diusulkan yang akan dibuat saat cetak biru diterapkan, pilih Terapkan cetak biru.

   Setelah permintaan tarik dibuat, Anda dapat menambahkan komentar. Komentar dapat ditambahkan ke permintaan tarik atau ke baris individual dalam file serta permintaan tarik keseluruhan. Anda dapat menambahkan tautan ke sumber daya seperti file dengan menggunakan @ tanda, diikuti dengan nama file.

   > **Note**
   >
   > Cetak biru tidak akan diterapkan sampai permintaan tarik disetujui dan digabungkan. Untuk informasi selengkapnya, lihat Meninjau permintaan tarik dan Menggabungkan permintaan tarik.

6. Dari kolom Status, pilih Permintaan tarik tertunda untuk baris cetak biru PDK - Infrastruktur, lalu pilih tautan permintaan tarik terbuka.
7. Pilih Gabung, pilih strategi penggabungan pilihan Anda, lalu pilih Gabung untuk menggabungkan perubahan dari cetak biru yang diterapkan.

   Setelah permintaan tarik digabungkan, `packages/infra` folder baru dibuat dalam proyek monorepo Anda, yang berisi infrastruktur yang akan menyebarkan proyek Anda ke AWS cloud.
8. Di panel navigasi, pilih Blueprints untuk mengonfirmasi bahwa Status PDK - Infrastruktur ditampilkan Terbaru.

Selanjutnya, Anda akan menerapkan PDK - DevOps blueprint untuk menyebarkan aplikasi Anda.

Langkah 5: Siapkan DevOps alur kerja untuk menyebarkan proyek Anda

DevOpsCetak biru PDK menghasilkan DevOps alur kerja yang diperlukan untuk membangun dan menerapkan proyek Anda menggunakan akun AWS dan peran yang ditentukan dalam konfigurasi.

Untuk menerapkan PDK - cetak biru DevOps

1. Di panel navigasi proyek monorepo Anda, pilih Blueprints, lalu pilih Terapkan cetak biru.
2. Pilih PDK - DevOps cetak biru, lalu pilih Berikutnya.
3. Di bawah Konfigurasi cetak biru, konfigurasikan parameter cetak biru:

- Pilih Bootstrap CDK di lingkungan saat ini.
- Di bidang input teks nama Stack, masukkan nama CloudFormation tumpukan yang ingin Anda gunakan. Ini harus cocok dengan nama tumpukan yang dikonfigurasi Langkah 4: Hasilkan infrastruktur untuk menyebarkan aplikasi ke AWS cloud untuk cetak biru PDK - Infrastructure.
- Pilih menu tarik-turun koneksi akun AWS, lalu pilih akun AWS yang ingin Anda gunakan untuk sumber daya. Untuk informasi selengkapnya, lihat Menambahkan Akun AWS ke spasi.
- Pilih Peran yang akan digunakan untuk menerapkan menu tarik-turun aplikasi Anda, lalu pilih peran IAM yang ingin Anda gunakan untuk menerapkan aplikasi proyek Anda.

> **Note**
>
> Saat membuat peran IAM, batasi `SourceArn` ke saat ini yang `ProjectID` ditemukan dalam pengaturan proyek. Untuk informasi selengkapnya, lihat Memahami peran CodeCatalystWorkflowDevelopmentRole-*spaceName*layanan.

- Pilih menu tarik-turun Wilayah, lalu pilih wilayah yang ingin Anda gunakan proyek monorepo Anda. Penerapan hanya berfungsi di wilayah di mana layanan AWS yang diperlukan ada. Untuk informasi selengkapnya, lihat AWS Services by Region.

4. Di tab Perubahan kode, tinjau perubahan yang diusulkan. Perbedaan yang ditampilkan dalam permintaan tarik menunjukkan perubahan pada proyek Anda pada saat permintaan tarik dibuat.
5. Jika Anda puas dengan perubahan yang diusulkan yang akan dibuat saat cetak biru diterapkan, pilih Terapkan cetak biru.

   Setelah permintaan tarik dibuat, Anda dapat menambahkan komentar. Komentar dapat ditambahkan ke permintaan tarik atau ke baris individual dalam file serta permintaan tarik keseluruhan. Anda dapat menambahkan tautan ke sumber daya seperti file dengan menggunakan @ tanda, diikuti dengan nama file.

   > **Note**
   >
   > Cetak biru tidak akan diterapkan sampai permintaan tarik disetujui dan digabungkan. Untuk informasi selengkapnya, lihat Meninjau permintaan tarik dan Menggabungkan permintaan tarik.

6. Dari kolom Status, pilih Permintaan tarik tertunda untuk baris cetak biru PDK - Infrastruktur, lalu pilih tautan permintaan tarik terbuka.

7. Pilih Gabung, pilih strategi penggabungan pilihan Anda, lalu pilih Gabung untuk menggabungkan perubahan dari cetak biru yang diterapkan.

   Setelah permintaan tarik digabungkan, `.codecatalyst/workflows` folder baru dihasilkan dalam proyek monorepo Anda.

8. Di panel navigasi, pilih Blueprints untuk mengonfirmasi bahwa Status PDK - DevOps muncul Update.

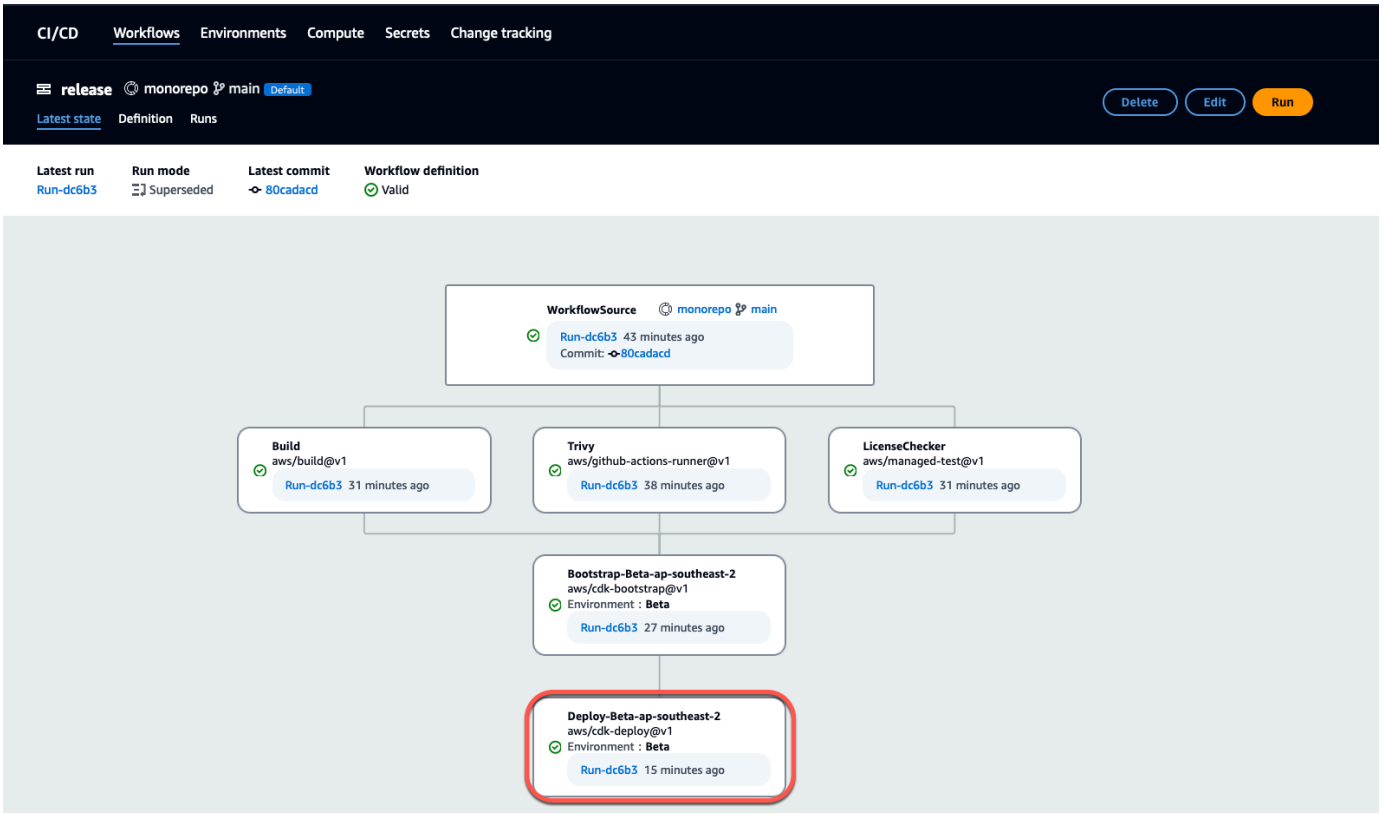
> **Note**
>
> DevOpsCetak biru PDK dan semua perubahan selanjutnya pada cetak biru PDK akan jauh lebih lambat sejak saat ini karena file kunci di balik layar dihasilkan untuk memastikan bahwa build dan penerapan dapat diulang di masa mendatang. Ini akan menghasilkan file kunci untuk semua paket di salah satu bahasa yang didukung.

Langkah 6: Konfirmasikan alur kerja rilis dan lihat situs web Anda

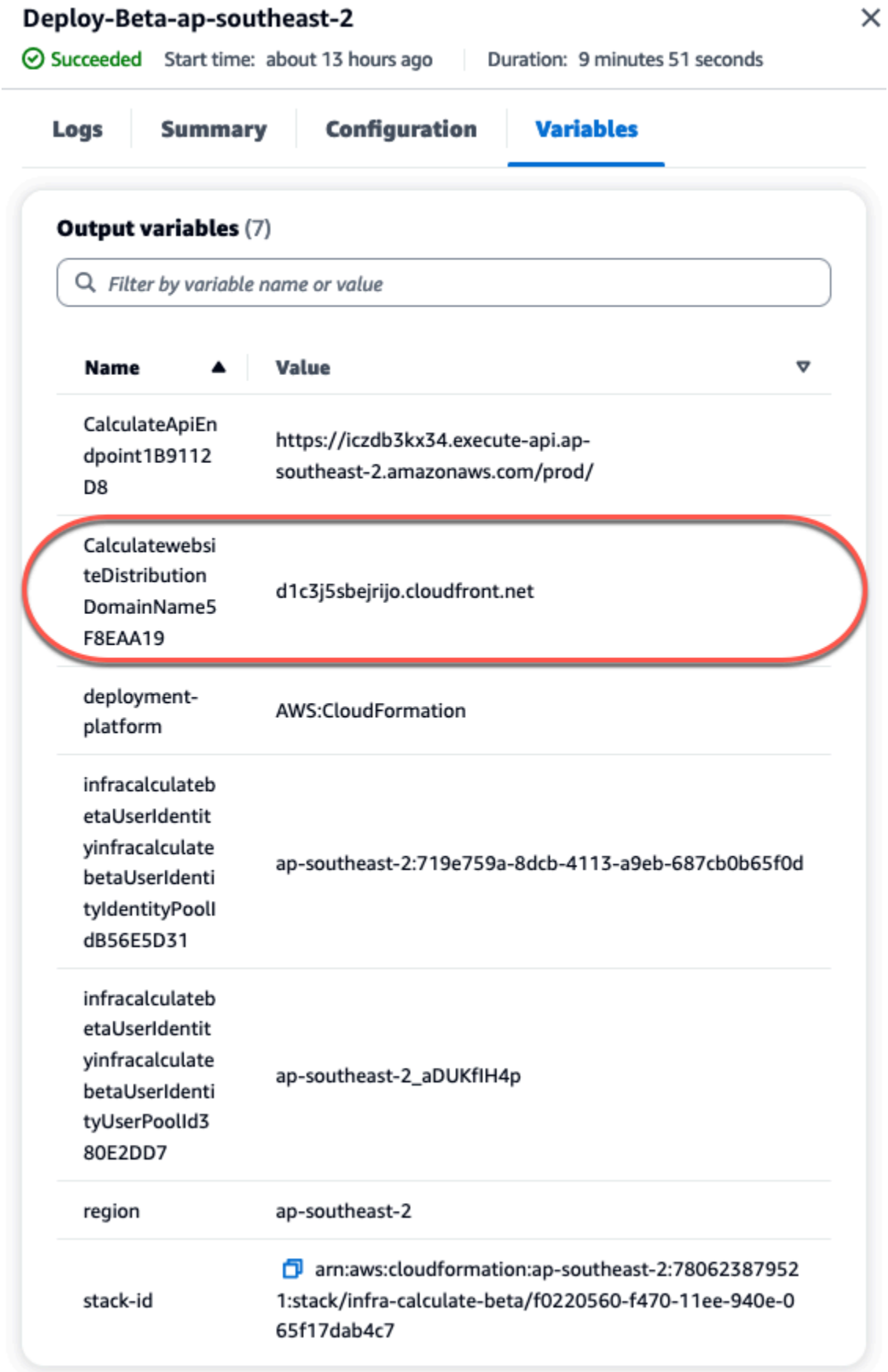
Setelah Anda menyelesaikan langkah-langkah sebelumnya, Anda dapat mengonfirmasi alur kerja rilis untuk memastikan proyek sedang dibangun.

Untuk mengonfirmasi alur kerja rilis dan melihat situs web Anda

1. Di panel navigasi proyek monorepo Anda, pilih CI/CD, lalu pilih Alur kerja.
2. Untuk alur kerja rilis, pilih alur kerja terbaru yang dijalankan untuk melihat detailnya. Untuk informasi selengkapnya, lihat Melihat status dan detail dari satu run.
3. Setelah alur kerja berhasil diselesaikan, pilih tindakan terakhir dalam alur kerja (misalnya, Deploy-B eta-ap-souteast -2, lalu pilih Variabel.



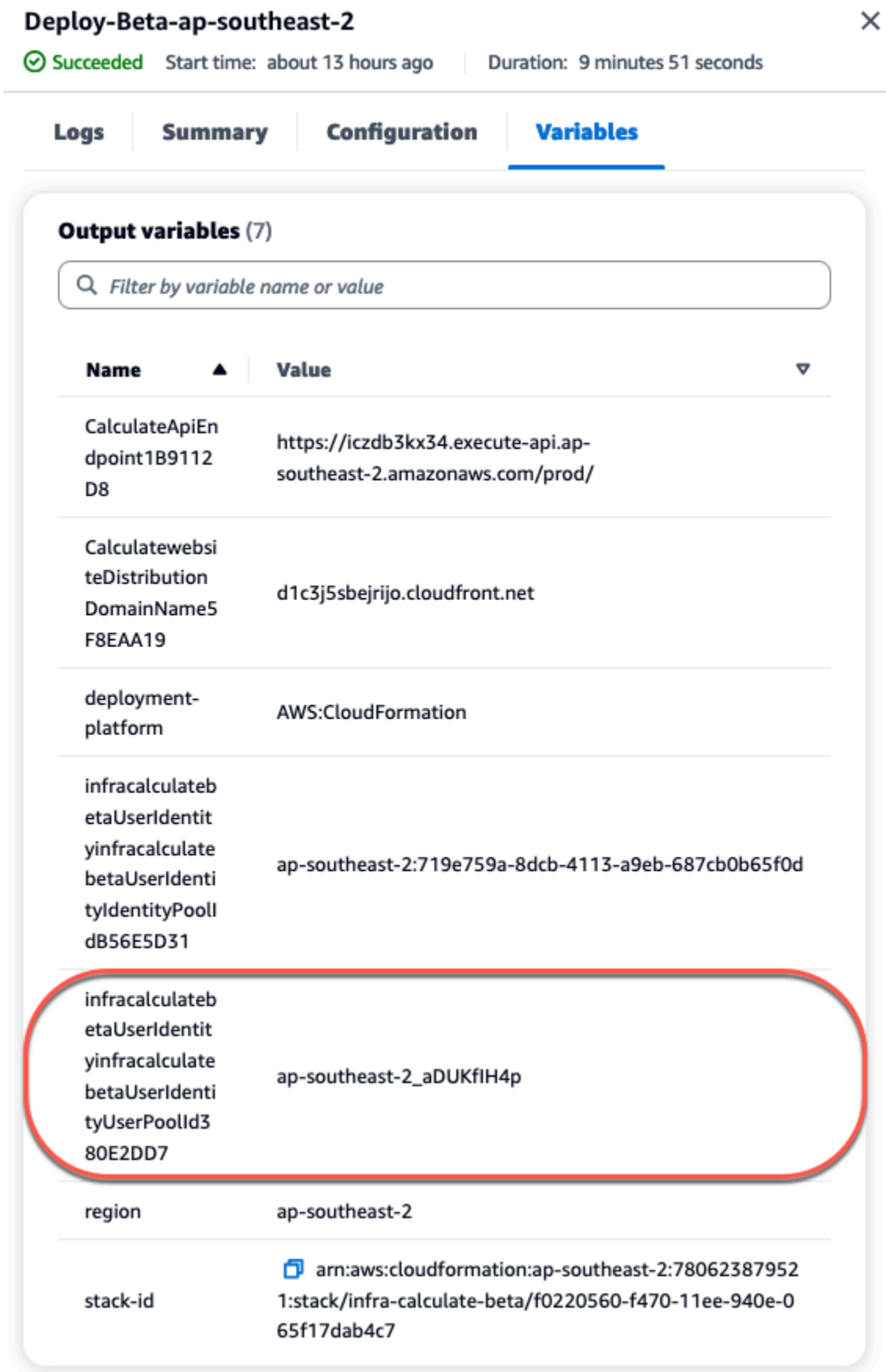
4. Lihat situs web yang digunakan dengan menyalin dan menempelkan tautan yang ditemukan di tabel Variabel (misalnya, ***MyPDKapi*** websiteDistributionDomain NameXXxx) ke jendela browser baru.


Deploy-Beta-ap-southeast-2
Succeeded
Start time: about 13 hours ago
Duration: 9 minutes 51 seconds
Logs
Summary
Configuration
Variables
Output variables (7)
Filter by variable name or value
Name
Value
CalculateApiEndpoint1B9112D8
https://iczdb3kx34.execute-api.ap-southeast-2.amazonaws.com/prod/
CalculatewebsiteDistributionDomainName5F8EAA19
d1c3j5sbejrijo.cloudfront.net
deployment-platform
AWS:CloudFormation
infracalculatebetaUserIdentityinfracalculatebetaUserIdentityIdentityPooldB56E5D31
ap-southeast-2:719e759a-8dcb-4113-a9eb-687cb0b65f0d
infracalculatebetaUserIdentityinfracalculatebetaUserIdentityUserPoolId380E2DD7
ap-southeast-2_aDUKfIH4p
region
ap-southeast-2
stack-id
arn:aws:cloudformation:ap-southeast-2:780623879521:stack/infra-calculate-beta/f0220560-f470-11ee-940e-065f17dab4c7

Anda memerlukan akun Amazon Cognito untuk masuk ke situs web Anda. Secara default, kumpulan pengguna tidak diatur untuk mengizinkan pendaftaran mandiri.

a. Arahkan ke konsol AWS Cognito.

b. *Dari tabel **kumpulan Pengguna**, pilih **nama kumpulan Pengguna** yang cocok dengan kumpulan pengguna yang dibuat oleh **PDK - DevOps** cetak biru, yang dapat ditemukan di tabel **Variabel** (misalnya, hitung **infra** menghitung XXXXX. betaUserIdentityinfra* betaUserIdentity IdentityPoolId Untuk informasi selengkapnya, lihat Memulai dengan kumpulan pengguna.


Deploy-Beta-ap-southeast-2
Succeeded
Start time: about 13 hours ago
Duration: 9 minutes 51 seconds
Logs
Summary
Configuration
Variables
Output variables (7)
Filter by variable name or value
Name
Value
CalculateApiEndpoint1B9112D8
https://iczdb3kx34.execute-api.ap-southeast-2.amazonaws.com/prod/
CalculatewebsiteDistributionDomainName5F8EAA19
d1c3j5sbejrijo.cloudfront.net
deployment-platform
AWS:CloudFormation
infracalculatebetaUserIdentityinfracalculatebetaUserIdentityIdentityPoolIdB56E5D31
ap-southeast-2:719e759a-8dcb-4113-a9eb-687cb0b65f0d
infracalculatebetaUserIdentityinfracalculatebetaUserIdentityUserPoolId380E2DD7
ap-southeast-2_aDUKfIH4p
region
ap-southeast-2
stack-id
arn:aws:cloudformation:ap-southeast-2:780623879521:stack/infra-calculate-beta/f0220560-f470-11ee-940e-065f17dab4c7

c. Pilih Create user (Buat pengguna).

d. Konfigurasikan parameter informasi pengguna:

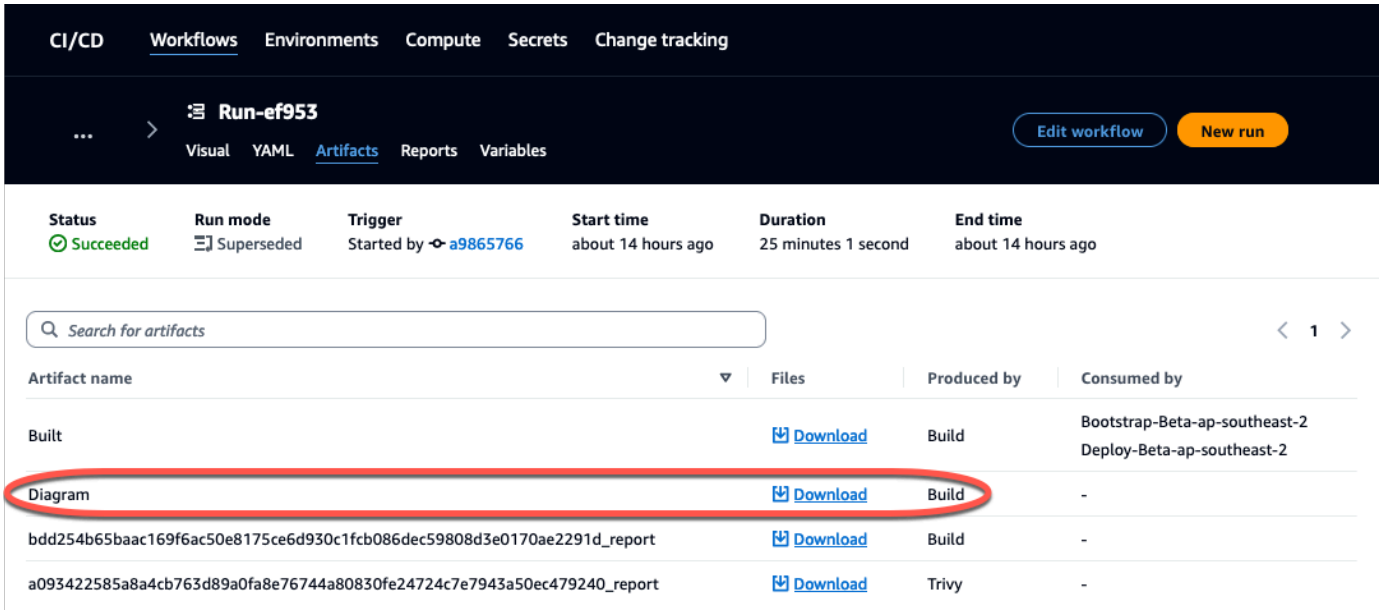
- Di bawah Pesan undangan, pilih Kirim undangan email.
- Di bidang input teks nama pengguna, masukkan nama pengguna.
- Di bidang masukan teks alamat email, masukkan nama pengguna.
- Di bawah Kata sandi sementara, pilih Buat kata sandi.

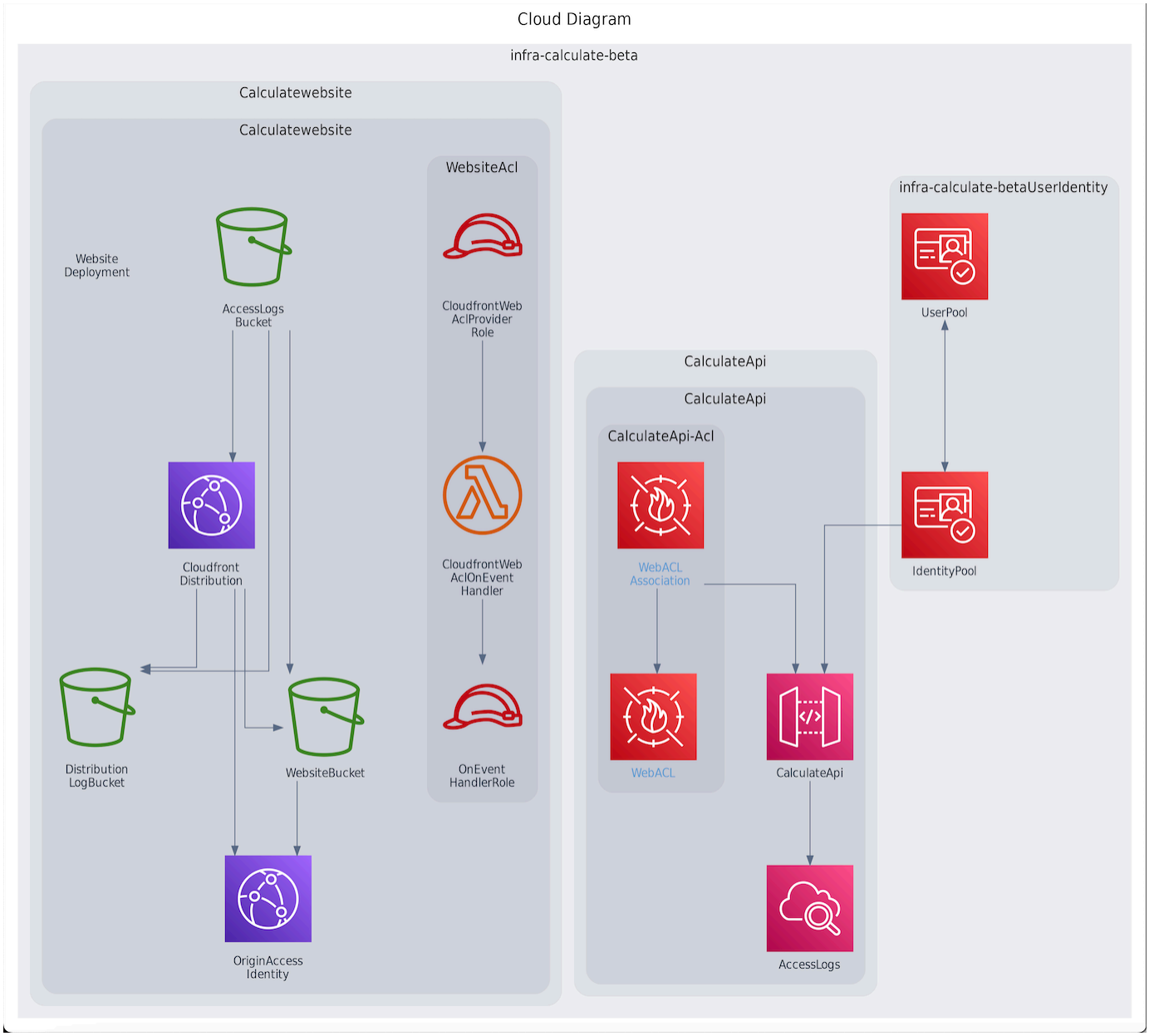
e. Pilih Create user (Buat pengguna).

f. Arahkan ke akun email yang Anda masukkan untuk parameter informasi Pengguna, buka email dengan kata sandi sementara. Catat kata sandi.

g. Arahkan kembali ke situs web yang digunakan, masukkan nama pengguna yang Anda buat dan kata sandi sementara yang Anda terima, lalu pilih Masuk.

5. (Opsional) Setelah alur kerja berhasil diselesaikan, Anda juga dapat melihat diagram yang dihasilkan. Pilih tab Artefak di CodeCatalyst, pilih Unduh untuk baris Diagram, lalu buka file yang diunduh.





Berkolaborasi dan iterasi pada proyek PDK

Setelah proyek Anda disiapkan, Anda dapat membuat perubahan pada kode sumber. Anda juga dapat mengundang anggota luar angkasa lainnya untuk mengerjakan proyek tersebut. Cetak biru PDK memungkinkan Anda untuk membangun aplikasi Anda secara iteratif, hanya menambahkan apa yang Anda butuhkan, saat Anda membutuhkannya, sambil mempertahankan kontrol penuh dari setiap konfigurasi cetak biru.

Topik

- Langkah 1: Undang anggota ke proyek Anda
- Langkah 2: Buat masalah untuk berkolaborasi dan melacak pekerjaan
- Langkah 3: Lihat repositori sumber Anda
- Langkah 4: Buat Lingkungan Pengembang dan buat perubahan kode
- Langkah 5: Dorong dan gabungkan perubahan kode

Langkah 1: Undang anggota ke proyek Anda

Anda dapat menggunakan konsol untuk mengundang pengguna ke proyek Anda. Anda dapat mengundang anggota ruang Anda atau menambahkan nama dari luar ruang Anda.

Untuk mengundang pengguna ke proyek Anda, Anda harus masuk dengan peran Administrator proyek atau administrator Space.

Anda tidak perlu mengundang pengguna dengan peran administrator Space ke proyek Anda karena mereka sudah memiliki akses implisit ke semua proyek di ruang tersebut.

Saat Anda mengundang pengguna ke proyek Anda (tanpa menetapkan peran administrator Space), pengguna akan ditampilkan dalam tabel Anggota proyek di bawah proyek dan dalam tabel anggota Proyek di bawah spasi.

Untuk mengundang anggota ke proyek Anda dari tab Pengaturan proyek

1. Arahkan ke proyek Anda.

   > **Tip**
   >
   > Anda dapat memilih proyek mana yang akan dilihat di bilah navigasi atas.

2. Di panel navigasi, pilih Pengaturan proyek.
3. Pilih tab Anggota.
4. Di anggota Project, pilih Undang anggota baru.
5. Ketik alamat email anggota baru, pilih peran untuk anggota ini, lalu pilih Undang. Untuk informasi lebih lanjut tentang peran, lihat Memberikan akses dengan peran pengguna.

Untuk mengundang anggota ke proyek Anda dari halaman ikhtisar Proyek

1. Arahkan ke proyek Anda.

> **Tip**
>
> Anda dapat memilih proyek mana yang akan dilihat di bilah navigasi atas.

2. Pilih tombol Anggota +.
3. Ketik alamat email anggota baru, pilih peran untuk anggota ini, lalu pilih Undang. Untuk informasi lebih lanjut tentang peran, lihat Memberikan akses dengan peran pengguna.

Langkah 2: Buat masalah untuk berkolaborasi dan melacak pekerjaan

CodeCatalyst membantu Anda melacak fitur, tugas, bug, dan pekerjaan lain yang terlibat dalam proyek Anda dengan masalah. Anda dapat membuat masalah untuk melacak pekerjaan dan ide yang dibutuhkan. Secara default, ketika Anda membuat masalah itu ditambahkan ke backlog Anda. Anda dapat memindahkan masalah ke papan tempat Anda melacak pekerjaan yang sedang berlangsung. Anda juga dapat menetapkan masalah ke anggota proyek tertentu. Pada langkah ini, buat masalah untuk membuat perubahan pada proyek PDK Anda.

Untuk membuat masalah

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek monorepo Anda di mana Anda ingin membuat masalah.
3. Pada halaman beranda proyek, pilih Buat masalah. Atau, di panel navigasi, pilih Masalah.
4. Pilih Buat masalah.

> **Note**
>
> Anda juga dapat menambahkan masalah sebaris saat menggunakan tampilan kisi.

5. Masukkan judul untuk masalah ini.
6. (Opsional) Masukkan Deskripsi. Untuk masalah ini, masukkan deskripsi berikut:`a change in the src/mysfit_data.json file.`. Anda dapat menggunakan Markdown untuk menambahkan pemformatan.
7. (Opsional) Pilih Status, Prioritas, Estimasi untuk masalah ini.
8. (Opsional) Tambahkan label yang ada atau buat label baru dan tambahkan dengan memilih + Tambahkan label.

a. Untuk menambahkan label yang ada, pilih label dari daftar. Anda dapat memasukkan istilah pencarian di bidang untuk mencari semua label yang berisi istilah itu dalam proyek.

b. Untuk membuat label baru dan menambahkannya, masukkan nama label yang ingin Anda buat di bidang pencarian dan tekan enter.

9. (Opsional) Tambahkan penerima tugas dengan memilih + Tambahkan penerima tugas. Anda dapat dengan cepat menambahkan diri Anda sebagai penerima tugas dengan memilih + Tambahkan saya.

> **Tip**
>
> Anda dapat memilih untuk menetapkan masalah ke Amazon Q agar Amazon Q mencoba menyelesaikan masalah tersebut. Untuk informasi selengkapnya, lihat Tutorial: Menggunakan fitur AI CodeCatalyst generatif untuk mempercepat pekerjaan pengembangan Anda.
> Fungsionalitas ini mengharuskan fitur AI generatif diaktifkan untuk ruang tersebut. Untuk informasi selengkapnya, lihat Mengelola fitur AI generatif.

10. (Opsional) Tambahkan bidang kustom yang ada atau buat bidang kustom baru. Masalah dapat memiliki beberapa bidang khusus.

a. Untuk menambahkan bidang kustom yang ada, pilih bidang kustom dari daftar. Anda dapat memasukkan istilah pencarian di bidang untuk mencari semua bidang kustom yang berisi istilah itu dalam proyek.

b. Untuk membuat bidang kustom baru dan menambahkannya, masukkan nama bidang khusus yang ingin Anda buat di bidang pencarian dan tekan enter. Kemudian pilih jenis bidang khusus yang ingin Anda buat dan tetapkan nilainya.

11. Pilih Buat masalah. Pemberitahuan muncul di sudut kanan bawah: Jika masalah berhasil dibuat, pesan konfirmasi muncul yang mengatakan masalah berhasil dibuat. Jika masalah tidak berhasil dibuat, pesan kesalahan dengan alasan kegagalan muncul. Anda kemudian dapat memilih Coba lagi untuk mengedit dan mencoba lagi membuat masalah, atau memilih Buang untuk membuang masalah. Kedua opsi akan mengabaikan notifikasi.

> **Note**
>
> Anda tidak dapat menautkan permintaan tarik ke masalah saat Anda membuatnya. Namun, Anda dapat mengeditnya setelah Anda membuatnya untuk menambahkan tautan ke permintaan tarik.

Untuk informasi selengkapnya, lihat Lacak dan atur pekerjaan dengan masalah di CodeCatalyst.

Langkah 3: Lihat repositori sumber Anda

Anda dapat melihat repositori sumber yang terkait dengan proyek di Amazon. CodeCatalyst Untuk repositori sumber di CodeCatalyst, halaman ikhtisar untuk repositori memberikan gambaran singkat tentang informasi dan aktivitas di repositori tersebut, termasuk:

- Deskripsi repositori, jika ada
- Jumlah cabang di repositori
- Jumlah permintaan tarik terbuka untuk repositori
- Jumlah alur kerja terkait untuk repositori
- File dan folder di cabang default, atau cabang yang Anda pilih
- Judul, penulis, dan tanggal komit terakhir ke cabang yang ditampilkan
- Isi file README.md dirender dalam Markdown, jika ada file README.md disertakan

Halaman ini juga menyediakan tautan ke komit, cabang, dan permintaan tarik untuk repositori, serta cara cepat untuk membuka, melihat, dan mengedit file individual.

> **Note**
>
> Anda tidak dapat melihat informasi ini tentang repositori tertaut di konsol. CodeCatalyst Untuk melihat informasi tentang repositori tertaut, pilih tautan dalam daftar repositori untuk membuka repositori itu di layanan yang menghostingnya.

Untuk menavigasi ke repositori sumber untuk sebuah proyek

1. Arahkan ke proyek Anda, dan lakukan salah satu hal berikut:

- Pada halaman ringkasan untuk proyek Anda, pilih repositori yang Anda inginkan dari daftar, lalu pilih Lihat repositori.
- Di panel navigasi, pilih Kode, lalu pilih Repositori sumber. Di repositori Sumber, pilih nama repositori dari daftar. Anda dapat memfilter daftar repositori dengan mengetikkan bagian dari nama repositori di bilah filter.

2. Pada halaman beranda untuk repositori, lihat isi repositori dan informasi tentang sumber daya terkait seperti jumlah permintaan tarik dan alur kerja. Secara default, konten untuk cabang default ditampilkan. Anda dapat mengubah tampilan dengan memilih cabang yang berbeda dari daftar drop-down.

> **Tip**
>
> Anda juga dapat dengan cepat menavigasi ke repositori proyek Anda dengan memilih Lihat kode proyek dari halaman ringkasan proyek.

Langkah 4: Buat Lingkungan Pengembang dan buat perubahan kode

Pada langkah ini, buat Dev Environment dan buat perubahan kode yang kemudian digabungkan ke cabang utama. Meskipun tutorial ini memandu Anda melalui proyek AWS PDK sederhana, Anda juga dapat mengikuti contoh yang lebih kompleks yang disediakan di repositori AWS PDK GitHub .

Untuk membuat Lingkungan Dev dengan cabang baru

1. Di panel navigasi proyek monorepo Anda, lakukan salah satu hal berikut:
   - Pilih Ikhtisar, lalu arahkan ke bagian My Dev Environments.
   - Pilih Kode, lalu pilih Lingkungan Dev.
   - Pilih Kode, pilih repositori Sumber, dan kemudian pilih repositori monorepo yang ingin Anda buat Lingkungan Dev.
2. Pilih IDE yang didukung dari menu tarik-turun. Untuk informasi selengkapnya, lihat Lingkungan pengembangan terintegrasi yang didukung untuk Lingkungan Dev.
3. Pilih Kloning repositori.
4. Pilih repositori untuk dikloning, pilih Bekerja di cabang baru, masukkan nama cabang ke bidang Nama cabang, dan pilih cabang untuk membuat cabang baru dari Buat cabang dari menu drop-down.

> **Note**
>
> Jika Anda membuat Lingkungan Dev dari halaman repositori Sumber atau dari repositori sumber tertentu, Anda tidak perlu memilih repositori. Lingkungan Dev akan dibuat dari repositori sumber yang Anda pilih dari halaman repositori Sumber.

5. (Opsional) Di Alias - opsional, masukkan alias untuk Lingkungan Pengembang.
6. (Opsional) Pilih tombol edit konfigurasi Lingkungan Dev untuk mengedit konfigurasi komputasi, penyimpanan, atau batas waktu Dev Environment.
7. (Opsional) Di Amazon Virtual Private Cloud (Amazon VPC) - opsional, pilih koneksi VPC yang ingin Anda kaitkan dengan Lingkungan Dev Anda dari menu tarik-turun.

   Jika VPC default disetel untuk ruang Anda, Lingkungan Pengembang Anda akan berjalan terhubung ke VPC tersebut. Anda dapat mengganti ini dengan mengaitkan koneksi VPC yang berbeda. Juga, perhatikan bahwa Lingkungan Dev yang terhubung dengan VPC tidak mendukung AWS Toolkit.

   > **Note**
   >
   > Saat Anda membuat Lingkungan Dev dengan koneksi VPC, antarmuka jaringan baru dibuat di dalam VPC. CodeCatalyst berinteraksi dengan antarmuka ini menggunakan peran VPC terkait. Juga, pastikan bahwa blok CIDR IPv4 Anda tidak dikonfigurasi ke rentang `172.16.0.0/12` alamat IP.

8. Pilih Buat. Saat Dev Environment Anda sedang dibuat, kolom status Dev Environment akan menampilkan Mulai, dan kolom status akan ditampilkan Running setelah Dev Environment telah dibuat.

Setelah Lingkungan Dev berjalan, Anda dapat bekerja dengan aplikasi sampel yang dihasilkan CodeCatalyst dengan membuat perubahan pada kode dengan permintaan tarik yang secara otomatis dibuat dan diterapkan ke sumber daya di akun AWS yang terhubung saat permintaan tarik digabungkan. Monorepo menjual devfile sehingga semua dependen dan runtime global yang diperlukan hadir secara otomatis.

Untuk mengubah kode dalam proyek Anda

1. Di terminal kerja Dev Environment Anda, navigasikan ke proyek monorepo Anda, lalu instal dependensi proyek Anda dengan menjalankan perintah berikut:

```
npx projen install
```

2. Arahkan kepackages/apis/*mypdkapi*/model/src/main/smithy/operations/say-hello.smithy, yang mendefinisikan operasi API contoh. Dalam tutorial ini, Anda akan membangun `Calculate` operasi sederhana yang menambahkan dua angka bersama-sama. Buat perubahan pada kode untuk menentukan operasi ini, termasuk input dan outputnya.

   Contoh:

```
$version: "2"
namespace com.aws

@http(method: "POST", uri: "/calculate")
@handler(language: "typescript")
operation Calculate {
    input := {
        @required
        numberA: Integer
        @required
        numberB: Integer
    }
    output := {
        @required
        result: Integer
    }
}
```

   `@handler`Sifat ini memberi tahu Type Safe API bahwa Anda akan mengimplementasikan operasi ini sebagai penangan AWS Lambda yang ditulis. TypeScript Type Safe API akan menghasilkan rintisan untuk operasi ini untuk Anda terapkan. TypeScript `@required`Sifat ditambahkan, yang berarti akan diberlakukan saat runtime oleh gateway API yang akan diterapkan. Untuk informasi lebih lanjut, lihat dokumentasi Smithy.

3. Ganti nama `/say-hello.smithy` file dengan nama yang sejajar dengan perubahan kode Anda (misalnya,). `calculate.smithy`

4. Arahkan kepackages/apis/*mypdkapi*/model/src/main/smithy/main.smithy, dan buat perubahan pada kode untuk menghubungkan operasi. Anda dapat mengekspos

`Calculate` operasi yang ditentukan dalam `/calculate.smithy` dengan mencantumkannya di `operations` bidang file ini.

Contoh:

```
$version: "2"
namespace com.aws

use aws.protocols#restJson1

/// A sample smithy api
@restJson1
service MyPDKApi {
    version: "1.0"
    operations: [Calculate]
    errors: [
      BadRequestError
      NotAuthorizedError
      InternalFailureError
    ]
}
```

5. Bangun perubahan dengan menjalankan perintah berikut:

```
npx projen build
```

> **Note**
>
> Secara opsional, Anda dapat meneruskan `--parallel X` flag, yang akan mendistribusikan build di seluruh X core.

Sejak `@handler` sifat ditambahkan, file berikut dibuat setelah build selesai:

- `/packages/apis/mypdkapi/handlers/typescript/src/calculate.ts`
- `/packages/apis/mypdkapi/handlers/typescript/test/calculate.test.ts`

6. Arahkan kepackages/apis/*mypdkapi*/handlers/typescript/src/*calculate.ts*, dan buat perubahan pada kode. File ini adalah penangan server yang dipanggil untuk API.

```
import {
```

```
  calculateHandler,
  CalculateChainedHandlerFunction,
  INTERCEPTORS,
  Response,
  LoggingInterceptor,
} from 'mypdkapi-typescript-runtime';

/**
 * Type-safe handler for the Calculate operation
 */
export const calculate: CalculateChainedHandlerFunction = async (request) => {
  LoggingInterceptor.getLogger(request).info('Start Calculate Operation');

  const { input } = request;

  return Response.success({
    result: input.body.numberA + input.body.numberB,
  });
};

/**
 * Entry point for the AWS Lambda handler for the Calculate operation.
 * The calculateHandler method wraps the type-safe handler and manages marshalling
 inputs and outputs
 */
export const handler = calculateHandler(...INTERCEPTORS, calculate);
```

7. Arahkan ke `/packages/apis/mypdkapi/handlers/typescript/test/calculate.test.ts` file, dan buat perubahan pada kode untuk memperbarui pengujian unit.

   Contoh:

```
import {
  CalculateChainedRequestInput,
  CalculateResponseContent,
} from 'mypdkapi-typescript-runtime';
import {
  calculate,
} from '../src/calculate';

// Common request arguments
const requestArguments = {
```

```
  chain: undefined as never,
  event: {} as any,
  context: {} as any,
  interceptorContext: {
    logger: {
      info: jest.fn(),
    },
  },
} satisfies Omit<CalculateChainedRequestInput, 'input'>;

describe('Calculate', () => {

  it('should return correct sum', async () => {
    const response = await calculate({
      ...requestArguments,
      input: {
        requestParameters: {},
        body: {
          numberA: 1,
          numberB: 2
        }
      },
    });

    expect(response.statusCode).toBe(200);
    expect((response.body as CalculateResponseContent).result).toEqual(3);
  });

});
```

8. Arahkan ke `/packages/infra/main/src/constructs/apis/mypdkapi.ts` file, dan buat perubahan pada kode untuk menambahkan integrasi `Calculate` operasi di infrastruktur CDK Anda. Konstruksi API memiliki properti integrasi, di mana Anda dapat meneruskan implementasi yang Anda tambahkan sebelumnya. Karena Anda menggunakan `@handler` sifat dalam model Smithy Anda untuk `Calculate` operasi, Anda dapat menggunakan konstruksi `CalculateFunction` CDK yang dihasilkan, yang telah dikonfigurasi sebelumnya, untuk menunjuk ke implementasi handler Anda.

   Contoh:

```
import { UserIdentity } from "@aws/pdk/identity";
import { Authorizers, Integrations } from "@aws/pdk/type-safe-api";
```

```
import { Stack } from "aws-cdk-lib";
import { Cors } from "aws-cdk-lib/aws-apigateway";
import {
  AccountPrincipal,
  AnyPrincipal,
  Effect,
  PolicyDocument,
  PolicyStatement,
} from "aws-cdk-lib/aws-iam";
import { Construct } from "constructs";
import { Api, CalculateFunction } from "calculateapi-typescript-infra";

/**
 * Api construct props.
 */
export interface CalculateApiProps {
  /**
   * Instance of the UserIdentity.
   */
  readonly userIdentity: UserIdentity;
}

/**
 * Infrastructure construct to deploy a Type Safe API.
 */
export class CalculateApi extends Construct {
  /**
   * API instance
   */
  public readonly api: Api;

  constructor(scope: Construct, id: string, props?: CalculateApiProps) {
    super(scope, id);

    this.api = new Api(this, id, {
      defaultAuthorizer: Authorizers.iam(),
      corsOptions: {
        allowOrigins: Cors.ALL_ORIGINS,
        allowMethods: Cors.ALL_METHODS,
      },
      integrations: {
        calculate: {
          integration: Integrations.lambda(new CalculateFunction(this,
 "CalculateFunction"))
```

```
        }
      },
      policy: new PolicyDocument({
        statements: [
          // Here we grant any AWS credentials from the account that the prototype
 is deployed in to call the api.
          // Machine to machine fine-grained access can be defined here using more
 specific principals (eg roles or
          // users) and resources (ie which api paths may be invoked by which
 principal) if required.
          // If doing so, the cognito identity pool authenticated role must still
 be granted access for cognito users to
          // still be granted access to the API.
          new PolicyStatement({
            effect: Effect.ALLOW,
            principals: [new AccountPrincipal(Stack.of(this).account)],
            actions: ["execute-api:Invoke"],
            resources: ["execute-api:/*"],
          }),
          // Open up OPTIONS to allow browsers to make unauthenticated preflight
 requests
          new PolicyStatement({
            effect: Effect.ALLOW,
            principals: [new AnyPrincipal()],
            actions: ["execute-api:Invoke"],
            resources: ["execute-api:/*/OPTIONS/*"],
          }),
        ],
      }),
    });

    // Grant authenticated users access to invoke the api
    props?.userIdentity.identityPool.authenticatedRole.addToPrincipalPolicy(
      new PolicyStatement({
        effect: Effect.ALLOW,
        actions: ["execute-api:Invoke"],
        resources: [this.api.api.arnForExecuteApi("*", "/*", "*")],
      }),
    );
  }
}
```

9. Bangun perubahan dengan menjalankan perintah berikut:

```
npx projen build
```

Setelah proyek Anda selesai dibangun, Anda dapat melihat diagram yang dihasilkan diperbarui, yang dapat ditemukan di`/packages/infra/main/cdk.out/cdkgraph/diagram.png`. Diagram menunjukkan bagaimana fungsi ditambahkan dan dihubungkan ke API yang dibuat. Saat kode CDK dimodifikasi, diagram ini juga diperbarui.

Anda sekarang dapat menerapkan perubahan Anda dengan mendorong dan menggabungkannya ke cabang utama repositori Anda.

Langkah 5: Dorong dan gabungkan perubahan kode

Komit dan dorong perubahan kode Anda, yang kemudian dapat digabungkan ke cabang utama repositori sumber Anda.

Untuk mendorong perubahan ke cabang fitur Anda

- Komit dan dorong perubahan ke cabang fitur Anda dengan menjalankan perintah berikut:

```
git add .
```

```
git commit -m "my commit message"
```

```
git push
```

Mendorong perubahan memicu alur kerja baru untuk cabang fitur Anda, yang dapat Anda lihat di CodeCatalyst konsol. Anda kemudian dapat membuat permintaan tarik untuk menggabungkan perubahan ke cabang utama repositori sumber Anda. Menggabungkan cabang fitur ke cabang utama Anda memicu alur kerja rilis. Anda juga dapat menautkan permintaan tarik ke masalah Anda.

Untuk membuat permintaan tarik dan menautkannya ke masalah Anda

1. Dalam proyek monorepo Anda, lakukan salah satu hal berikut:
   - Di panel navigasi, pilih Kode, pilih Tarik permintaan, lalu pilih Buat permintaan tarik.
   - Pada halaman beranda repositori, pilih Lainnya, lalu pilih Buat permintaan tarik.

- Pada halaman proyek, pilih Buat permintaan tarik.

2. Di repositori Sumber, pastikan bahwa repositori sumber yang ditentukan adalah yang berisi kode yang dikomit. Opsi ini hanya muncul jika Anda tidak membuat permintaan tarik dari halaman utama repositori.
3. Di cabang Tujuan, pilih cabang utama untuk menggabungkan kode setelah ditinjau.
4. Di cabang Sumber, pilih cabang fitur yang berisi kode komit.
5. Dalam judul permintaan tarik, masukkan judul yang membantu pengguna lain memahami apa yang perlu ditinjau dan alasannya.
6. (Opsional) Dalam deskripsi permintaan Tarik, berikan informasi seperti tautan ke masalah atau deskripsi perubahan Anda.

> **Tip**
>
> Anda dapat memilih Tulis deskripsi agar saya CodeCatalyst secara otomatis menghasilkan deskripsi tentang perubahan yang terkandung dalam permintaan tarik. Anda dapat membuat perubahan pada deskripsi yang dibuat secara otomatis setelah Anda menambahkannya ke permintaan tarik.
> Fungsionalitas ini mengharuskan fitur AI generatif diaktifkan untuk ruang tersebut. Untuk informasi selengkapnya, lihat Mengelola fitur AI generatif di Amazon CodeCatalyst.

7. Dalam Masalah, pilih Masalah tautan, lalu pilih masalah yang Anda buatLangkah 2: Buat masalah untuk berkolaborasi dan melacak pekerjaan. Untuk memutuskan tautan masalah, pilih ikon batalkan tautan.
8. (Opsional) Di Reviewer wajib, pilih Tambahkan pengulas yang diperlukan. Pilih dari daftar anggota proyek untuk menambahkannya. Pengulas yang diperlukan harus menyetujui perubahan sebelum permintaan tarik dapat digabungkan ke cabang tujuan.

> **Note**
>
> Anda tidak dapat menambahkan reviewer sebagai reviewer wajib dan reviewer opsional. Anda tidak dapat menambahkan diri Anda sebagai reviewer.

9. (Opsional) Di pengulas opsional, pilih Tambahkan pengulas opsional. Pilih dari daftar anggota proyek untuk menambahkannya. Peninjau opsional tidak harus menyetujui perubahan sebagai persyaratan sebelum permintaan tarik dapat digabungkan ke cabang tujuan.

10. Permintaan tarik Anda harus ditinjau dan digabungkan ke cabang utama oleh pengulas atau Anda sendiri. Untuk informasi selengkapnya, lihat Menggabungkan permintaan tarik.

    Saat perubahan Anda digabungkan ke cabang utama repositori sumber Anda, alur kerja baru akan dipicu secara otomatis.
11. Setelah penggabungan selesai, Anda dapat memindahkan masalah Anda ke Selesai.

    a. Di panel navigasi, pilih Masalah.
    b. Pilih masalah yang dibuatLangkah 2: Buat masalah untuk berkolaborasi dan melacak pekerjaan, pilih tarik-turun Status, lalu pilih Selesai.

Alur kerja rilis menyebarkan aplikasi Anda setelah dijalankan dengan sukses, sehingga Anda dapat melihat perubahan.

Untuk mengonfirmasi alur kerja rilis dan melihat situs web Anda

1. Di panel navigasi proyek monorepo Anda, pilih CI/CD, lalu pilih Alur kerja.
2. Untuk alur kerja rilis, pilih alur kerja terbaru yang dijalankan untuk melihat detailnya. Untuk informasi selengkapnya, lihat Melihat status dan detail dari satu run.
3. Setelah alur kerja berhasil diselesaikan, pilih tindakan terakhir dalam alur kerja (Deploy-B eta-ap-souteast -2), lalu pilih Variabel.
4. Lihat situs web yang digunakan dengan menyalin dan menempelkan tautan dari baris ***MyPDKAPI*** websiteDistributionDomain NameXXXXX ke jendela browser baru.
5. Masukkan nama pengguna dan kata sandi yang Anda buatLangkah 6: Konfirmasikan alur kerja rilis dan lihat situs web Anda, lalu pilih Masuk.
6. (Opsional) Uji perubahan dalam aplikasi Anda.

    a. Pilih menu dropdown POST.
    b. Masukkan dua nilai untuk `numberA` dan`number B`, lalu pilih Execute.
    c. Konfirmasikan hasilnya di badan Response.

Seiring waktu, versi katalog blueprints PDK dapat berubah. Anda dapat mengubah cetak biru proyek Anda ke versi katalog agar tetap up to date dengan perubahan terbaru. Anda dapat melihat perubahan kode dan lingkungan yang terpengaruh sebelum mengubah versi cetak biru proyek Anda. Untuk informasi selengkapnya, lihat Mengubah versi cetak biru dalam sebuah proyek.

Atur sumber daya dengan spasi di CodeCatalyst

Anda membuat ruang yang mewakili Anda, perusahaan, departemen, atau grup Anda, dan menyediakan tempat di mana tim pengembangan Anda dapat mengelola proyek. Anda harus membuat ruang untuk menambahkan proyek, anggota, dan sumber daya cloud terkait yang Anda buat di Amazon CodeCatalyst.

> **Note**
>
> Nama ruang harus unik di seberang CodeCatalyst. Anda tidak dapat menggunakan kembali nama spasi yang dihapus.

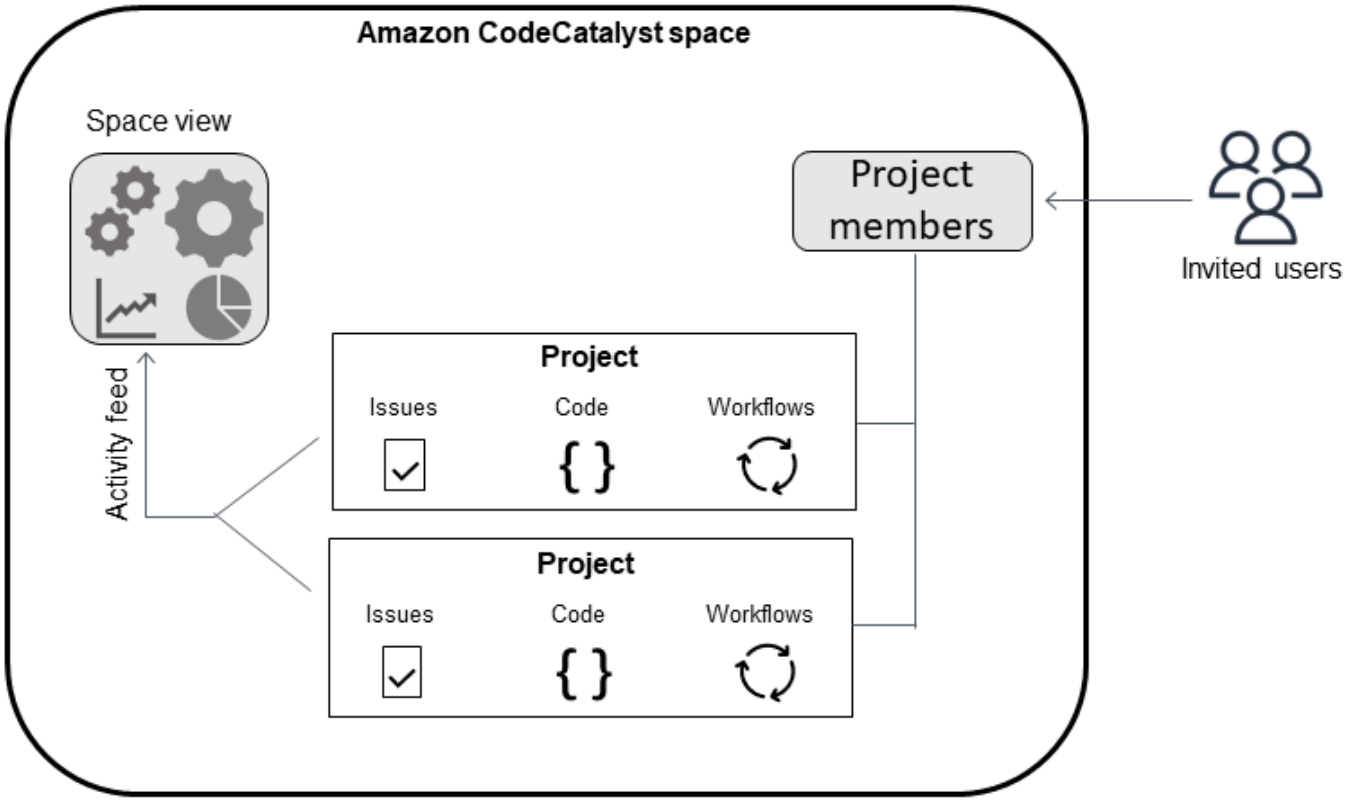
Saat Anda membuat spasi, Anda secara otomatis diberi peran administrator Space. Anda dapat menambahkan peran ini ke pengguna lain di ruang tersebut.

Dengan peran administrator Space, Anda dapat mengelola ruang sebagai berikut:

- Tambahkan administrator ruang lain ke ruang
- Ubah peran dan izin anggota
- Edit atau hapus spasi
- Buat proyek dan undang anggota ke proyek
- Lihat daftar semua proyek di ruang
- Melihat umpan aktivitas untuk semua proyek di ruang

Saat Anda membuat spasi, Anda secara otomatis ditambahkan ke ruang dengan dua peran: peran Administrator ruang, dan peran administrator Proyek untuk proyek yang Anda buat sebagai bagian dari pembuatan ruang. Pengguna tambahan ditambahkan sebagai anggota ke ruang secara otomatis ketika mereka menerima undangan ke proyek. Keanggotaan di ruang ini tidak memberikan izin apa pun di ruang tersebut. Apa yang dapat dilakukan pengguna dalam suatu ruang ditentukan oleh peran yang dimiliki pengguna dalam proyek tertentu.

Untuk informasi lebih lanjut tentang peran, lihat Memberikan akses dengan peran pengguna.



Berikut ini adalah pertimbangan tambahan untuk akun tambahan:

- Ada one-to-one pemetaan koneksi akun Akun AWS untuk ruang. Satu Akun AWS dapat ditambahkan ke beberapa ruang yang berbeda. Akun AWS yang Anda gunakan tidak perlu unik dan dapat digunakan oleh lebih dari satu spasi.
- Akun AWS ditambahkan ke CodeCatalyst ruang dapat digunakan dalam proyek apa pun di ruang itu.
- Meskipun setiap lingkungan dapat mendukung beberapa Akun AWS, Anda hanya dapat menggunakan satu akun per lingkungan dalam suatu tindakan.
- Penagihan dikonfigurasi pada tingkat ruang. Beberapa akun dapat dikonfigurasi untuk penagihan, tetapi hanya satu yang dapat aktif dalam satu CodeCatalyst spasi. Hanya satu yang Akun AWS dapat digunakan sebagai akun penagihan untuk ruang di CodeCatalyst. Jika akun sudah digunakan untuk ruang, Anda harus menggunakan akun penagihan yang berbeda untuk ruang tambahan.
- Setelah membuat koneksi, Anda harus menambahkan peran AWS IAM ke koneksi jika alur kerja Anda harus mengakses peran IAM tersebut dengan lingkungan Anda. CodeCatalyst Untuk

informasi selengkapnya tentang bagaimana lingkungan digunakan, lihatMenyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst.

Topik

Menciptakan ruang

Saat pertama kali mendaftar di Amazon CodeCatalyst dengan AWS Builder ID, Anda diharuskan membuat spasi. Untuk informasi selengkapnya, lihat Siapkan dan masuk ke CodeCatalyst. Anda dapat memilih untuk membuat ruang tambahan untuk memenuhi kebutuhan bisnis Anda.

> **Note**
>
> Nama ruang harus unik di seberang CodeCatalyst. Anda tidak dapat menggunakan kembali nama spasi yang dihapus.

Informasi dalam panduan ini disediakan untuk membuat spasi CodeCatalyst yang mendukung pengguna AWS Builder ID. Langkah-langkah untuk mengatur dan mengelola ruang yang mendukung federasi identitas disediakan dalam Panduan CodeCatalyst Administrator. Untuk bekerja dengan spasi yang disiapkan untuk federasi identitas, lihat Pengaturan dan administrasi untuk CodeCatalyst spasi di Panduan CodeCatalyst Administrator Amazon.

Untuk membuat spasi tambahan yang mendukung pengguna AWS Builder ID, Anda harus diberi peran administrator Space.

> **Note**
>
> Saat Anda membuat ruang tambahan, Anda tidak diminta untuk membuat proyek. Untuk mempelajari cara membuat proyek di ruang, lihatMembuat proyek.

Untuk membuat ruang lain

1. Di AWS Management Console, pastikan Anda masuk dengan hal yang sama Akun AWS yang ingin Anda kaitkan dengan CodeCatalyst ruang Anda.
2. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
3. Arahkan ke ruang Anda.

   > **Tip**
   >
   > Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

4. Pilih Buat ruang.
5. Pada halaman Buat spasi, dalam nama Spasi, masukkan nama untuk spasi. Anda tidak dapat mengubah ini nanti.

   > **Note**
   >
   > Nama ruang harus unik di seberang CodeCatalyst. Anda tidak dapat menggunakan kembali nama spasi yang dihapus.

6. Di Wilayah AWS, pilih Wilayah tempat Anda ingin menyimpan ruang dan data proyek Anda. Anda tidak dapat mengubah ini nanti.
7. Di Akun AWS ID, masukkan ID dua belas digit untuk akun yang ingin Anda sambungkan ke ruang Anda.

   Dalam token verifikasi AWS akun, salin ID token yang dihasilkan. Token secara otomatis disalin untuk Anda, tetapi Anda mungkin ingin menyimpannya saat Anda menyetujui permintaan AWS koneksi.
8. Pilih Verifikasi masuk AWS.

9. Halaman CodeCatalyst spasi Verifikasi Amazon terbuka di AWS Management Console. Ini adalah halaman Amazon CodeCatalyst Spaces. Anda mungkin perlu masuk untuk mengakses halaman.

   Di dalam AWS Management Console, pastikan untuk memilih yang sama Wilayah AWS di mana Anda ingin membuat ruang Anda.

   Untuk mengakses halaman secara langsung, masuk ke Amazon CodeCatalyst Spaces AWS Management Console di https://console.aws.amazon.com/codecatalyst/home/.

   Token verifikasi secara otomatis dimasukkan dalam token Verifikasi. Spanduk sukses menunjukkan pesan bahwa token adalah token yang valid.

10. Pilih Verifikasi ruang.

    Pesan sukses terverifikasi Akun ditampilkan untuk menunjukkan bahwa akun telah ditambahkan ke ruang.

11. Tetap di halaman Verifikasi Amazon CodeCatalyst spasi. Pilih tautan berikut: Untuk menambahkan peran IAM untuk ruang ini, lihat detail spasi.

    Halaman detail CodeCatalyst spasi terbuka di AWS Management Console. Ini adalah halaman Amazon CodeCatalyst Spaces. Anda mungkin perlu masuk untuk mengakses halaman.

12. Di bawah peran IAM yang tersedia CodeCatalyst, pilih Tambahkan peran IAM.

    Peran Add IAM tersedia untuk tampilan CodeCatalyst halaman.

13. Pilih Buat peran administrator CodeCatalyst pengembangan di IAM. Opsi ini membuat peran layanan yang berisi kebijakan izin dan kebijakan kepercayaan untuk peran pengembangan.

    Peran pengembang adalah peran AWS IAM yang memungkinkan CodeCatalyst alur kerja Anda mengakses AWS sumber daya seperti Amazon S3, Lambda, dan. AWS CloudFormation Peran akan memiliki nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* dengan pengenal unik ditambahkan. Untuk informasi selengkapnya tentang kebijakan peran dan peran, lihatMemahami peran CodeCatalystWorkflowDevelopmentRole-***`spaceName`***layanan.

14. Pilih Buat peran pengembangan.

15. Pada halaman koneksi, di bawah peran IAM yang tersedia untuk CodeCatalyst, lihat peran pengembang dalam daftar peran IAM yang ditambahkan ke akun Anda.

16. Pilih Pergi ke Amazon CodeCatalyst.

17. Pada halaman pembuatan di CodeCatalyst, pilih Buat ruang.

Mengedit ruang

Anda dapat mengubah deskripsi spasi untuk membantu pengguna lebih memahami untuk apa ruang tersebut.

Anda harus memiliki peran administrator Space untuk mengedit detail ruang.

Informasi dalam panduan ini disediakan untuk mengedit spasi CodeCatalyst yang mendukung pengguna AWS Builder ID. Untuk mempelajari selengkapnya tentang langkah-langkah menyiapkan dan mengelola ruang yang mendukung federasi identitas, lihat Pengaturan dan administrasi untuk CodeCatalyst spasi di Panduan CodeCatalyst Administrator Amazon.

Untuk mengedit deskripsi spasi

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda.

   > **Tip**
   >
   > Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Pada tab Pengaturan ruang, pilih Edit. Buat perubahan yang Anda inginkan pada deskripsi spasi, lalu pilih Simpan.

Menghapus spasi

Anda dapat menghapus spasi untuk menghapus akses ke semua sumber daya ruang. Anda harus memiliki peran administrator Space untuk menghapus spasi.

> **Note**
>
> Anda tidak dapat membatalkan penghapusan ruang.

Setelah Anda menghapus spasi, semua anggota ruang tidak akan dapat mengakses sumber daya ruang. Penagihan untuk sumber daya ruang juga akan berhenti, dan alur kerja apa pun yang diminta oleh repositori sumber pihak ketiga akan dihentikan.

> **Note**
>
> Nama ruang harus unik di seberang CodeCatalyst. Anda tidak dapat menggunakan kembali nama spasi yang dihapus.

Informasi dalam panduan ini disediakan untuk menghapus spasi CodeCatalyst yang mendukung pengguna AWS Builder ID. Untuk mempelajari selengkapnya tentang langkah-langkah menyiapkan dan mengelola ruang yang mendukung federasi identitas, lihat Pengaturan dan administrasi untuk CodeCatalyst spasi di Panduan CodeCatalyst Administrator Amazon.

Untuk menghapus spasi

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda.

   > **Tip**
   >
   > Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Pilih Pengaturan, lalu pilih Hapus.
4. Ketik **`delete`** untuk mengonfirmasi penghapusan.
5. Pilih Hapus.

   > **Note**
   >
   > Jika Anda memiliki lebih dari satu spasi, Anda akan diarahkan ke halaman ikhtisar ruang. Jika Anda termasuk dalam satu spasi, Anda diarahkan ke halaman pembuatan ruang.

Memantau aktivitas untuk pengguna dan sumber daya di suatu ruang

Untuk melihat proyek yang baru dibuat dan pembaruan status, Anda dapat menggunakan CodeCatalyst konsol untuk melihat umpan aktivitas yang menampilkan pembaruan untuk sumber daya ruang.

Di feed aktivitas, Anda dapat melihat metrik seperti alur kerja gagal berjalan dan proyek yang dibuat.

Untuk melihat aktivitas di ruang Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.

> **Tip**
>
> Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Pilih Aktivitas.
4. Lihat informasi dalam Aktivitas.
5. Untuk memfilter berdasarkan aktivitas, pilih pemilih di kanan atas.
6. Untuk melihat semua aktivitas di ruang Anda, pilih Jenis aktivitas apa pun.

Memungkinkan akses ke AWS sumber daya yang terhubung Akun AWS

Anda dapat menggunakan sumber daya dari Akun AWS CodeCatalyst ruang Amazon Anda. Untuk melakukannya, Anda harus mengatur koneksi antara ruang Akun AWS dan ruang Anda CodeCatalyst. Membuat koneksi seperti ini berarti bahwa proyek dan alur kerja di dalam CodeCatalyst ruang Anda dapat berinteraksi dengan sumber daya di ruang Anda Akun AWS. Anda harus membuat satu koneksi untuk setiap yang ingin Akun AWS Anda gunakan dengan CodeCatalyst ruang Anda.

Setelah Anda membuat koneksi, Anda dapat memilih untuk mengaitkan peran AWS IAM dengannya.

Topik

- Menambahkan Akun AWS ke spasi
- Menambahkan peran IAM ke koneksi akun
- Menambahkan koneksi akun dan peran IAM ke lingkungan penerapan Anda
- Melihat koneksi akun
- Menghapus akun dari spasi (in CodeCatalyst)
- Mengkonfigurasi akun penagihan untuk spasi

Anda dapat mengatur CodeCatalyst untuk menggunakan otorisasi Akun AWS dengan menambahkan akun ke ruang Anda. Dengan Akun AWS menambahkan CodeCatalyst ruang Anda, Anda dapat memberikan alur kerja proyek Anda akses ke Akun AWS sumber daya dan konfigurasi penagihan Anda.

Menambahkan Akun AWS membuat koneksi yang mengizinkan CodeCatalyst untuk menggunakan akun ini. Anda dapat menggunakan add Akun AWS untuk melakukan hal berikut:

- Siapkan penagihan untuk CodeCatalyst spasi. Lihat Mengelola penagihan di Panduan CodeCatalyst Administrator Amazon.
- Izinkan CodeCatalyst untuk mengambil peran IAM untuk mengakses AWS sumber daya dan menyebarkan ke Layanan AWS dalam akun. Lihat Memungkinkan akses ke AWS sumber daya yang terhubung Akun AWS.

Koneksi akun dibuat dengan menyelesaikan otorisasi dengan. Akun AWS Setelah koneksi dibuat, Anda selanjutnya mengonfigurasi koneksi untuk alur kerja dan proyek yang akan digunakan dengan menambahkan peran IAM.

Menambahkan Akun AWS ke spasi

Anda menggunakan CodeCatalyst konsol dan AWS Management Console untuk menghubungkan ruang Anda ke file Akun AWS.

Sebelum menambahkan Akun AWS ke spasi CodeCatalyst, lengkapi prasyarat berikut:

- Buat Akun AWS dan dapatkan izin untuk membuat peran AWS IAM di akun yang ingin Anda sambungkan.
- Buat peran atau peran IAM yang ingin Anda kaitkan dengan koneksi akun Anda, termasuk kebijakan IAM dengan izin untuk peran tersebut.
- Dapatkan peran administrator CodeCatalyst Space di ruang tempat Anda ingin membuat koneksi.

Topik

- Langkah selanjutnya: Buat peran IAM tambahan untuk koneksi akun Anda

Langkah 1: Membuat permintaan koneksi

Membuat permintaan koneksi di CodeCatalyst konsol menghasilkan token koneksi yang dapat Anda gunakan untuk menyelesaikan otorisasi.

Anda harus memiliki peran administrator Space atau Power user di CodeCatalyst ruang tempat Anda ingin membuat koneksi. Anda juga harus memiliki izin administratif untuk yang ingin Akun AWS Anda tambahkan.

Untuk membuat koneksi

1. Di AWS Management Console, pastikan Anda masuk dengan akun yang sama dengan yang ingin Anda buat koneksi.
2. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
3. Arahkan ke CodeCatalyst ruang Anda. Pilih Pengaturan, lalu pilih Akun AWS.
4. Pilih Tambahkan file Akun AWS.
5. Pada CodeCatalyst halaman Associate Akun AWS with Amazon, di Akun AWS ID, masukkan ID dua belas digit untuk akun yang ingin Anda sambungkan ke ruang Anda. Untuk informasi tentang menemukan Akun AWS ID Anda, lihat Akun AWS ID Anda dan aliasnya.
6. Di nama CodeCatalyst tampilan Amazon, masukkan nama referensi untuk akun tersebut.
7. (Opsional) Dalam deskripsi Koneksi, masukkan deskripsi untuk akun yang akan membantu Anda memilih proyek tempat akun dan peran atau peran akan diterapkan.
8. Pilih Kaitkan Akun AWS.
9. Halaman kembali ke halaman Akun AWS detail tempat spanduk sukses ditampilkan.

Langkah 2: Menerima permintaan koneksi akun

Setelah Anda mengirimkan permintaan di CodeCatalyst konsol untuk terhubung ke Anda Akun AWS, Anda bekerja dengan AWS administrator Anda untuk menerima permintaan koneksi dengan mengirimkannya dengan token koneksi yang disediakan.

Pastikan Anda memiliki izin administrator untuk akun Anda, dan Anda masuk ke yang sama AWS Management Console Akun AWS dengan saat Anda membuat koneksi.

Untuk menyetujui permintaan koneksi (konsol)

1. Di AWS Management Console, pastikan Anda masuk dengan akun yang sama dengan yang ingin Anda buat koneksi.
2. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
3. Arahkan ke CodeCatalyst ruang Anda. Pilih Pengaturan, lalu pilih Akun AWS.
4. Pada halaman Akun AWS detail, pilih Penyiapan lengkap di AWS Management Console.
5. Halaman CodeCatalyst spasi Verifikasi Amazon terbuka di AWS Management Console. Ini adalah halaman Amazon CodeCatalyst Spaces. Anda mungkin perlu masuk untuk mengakses halaman.

   Untuk mengakses halaman secara langsung, masuk ke Amazon CodeCatalyst Spaces AWS Management Console di https://console.aws.amazon.com/codecatalyst/home/.

   Token verifikasi secara otomatis dimasukkan dalam token Verifikasi. Pesan sukses menunjukkan pesan bahwa token adalah token yang valid.
6. (Opsional) Di bawah tingkatan berbayar resmi, pilih Otorisasi tingkatan berbayar (Standar, Perusahaan) untuk mengaktifkan tingkatan berbayar untuk akun penagihan Anda.

   > **Note**
   >
   > Ini tidak meningkatkan tingkat penagihan ke tingkat berbayar. Namun, ini mengonfigurasi Akun AWS sehingga Anda dapat mengubah tingkat penagihan untuk ruang Anda kapan saja. CodeCatalyst Anda dapat mengaktifkan tingkatan berbayar kapan saja. Tanpa membuat perubahan ini, ruang hanya dapat menggunakan tingkat Gratis.
7. Pilih Verifikasi ruang.

   Pesan sukses terverifikasi Akun ditampilkan untuk menunjukkan bahwa akun telah ditambahkan ke ruang.

Langkah 3: Tinjau koneksi yang disetujui

Setelah mendapatkan koneksi disetujui, Anda dapat melihat koneksi di konsol, bersama dengan peran IAM yang Anda tambahkan ke dalamnya.

Untuk meninjau koneksi yang disetujui

1. Arahkan ke CodeCatalyst ruang Anda. Pilih Pengaturan, lalu pilih Akun AWS.
2. Koneksi akun terdaftar dengan tanggal pembuatannya.
3. Pilih nama tampilan akun. Halaman Akun AWS detail ditampilkan.

Langkah 4: Tambahkan peran IAM ke koneksi Anda

Jika Anda menggunakan peran IAM yang dikonfigurasi untuk tindakan CodeCatalyst penerapan, tambahkan peran tersebut ke lingkungan penerapan Anda. Untuk informasi selengkapnya, lihat Menambahkan peran IAM ke koneksi akun.

Langkah selanjutnya: Buat peran IAM tambahan untuk koneksi akun Anda

Setelah Anda membuat koneksi, Anda dapat membuat peran IAM tambahan untuk ditambahkan ke dalamnya. Peran IAM yang Anda tambahkan bergantung pada alur kerja Anda. Misalnya, tindakan CodeCatalyst build membutuhkan peran CodeCatalyst build.

Untuk menghubungkan akun Anda, Anda memerlukan Nama Sumber Daya Amazon (ARN) untuk peran yang Anda buat. Salin ARN untuk peran atau peran Anda seperti yang dijelaskan di sini. Untuk informasi selengkapnya tentang bekerja dengan ARN untuk peran IAM, lihat Amazon Resource Name (ARN).

Untuk mengakses peran IAM Anda ARN

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi, pilih Peran.
3. Di kotak pencarian, masukkan nama peran yang ingin Anda tambahkan.
4. Pilih peran dari daftar.

   Halaman Ringkasan peran muncul.
5. Di bagian atas, salin nilai ARN Peran.

Menambahkan peran IAM ke koneksi akun

Bagian dari membuat koneksi akun Anda termasuk menambahkan peran atau peran IAM yang ingin Anda gunakan dengan proyek di CodeCatalyst ruang Anda.

> **Note**
>
> Untuk menggunakan peran IAM dengan koneksi akun, pastikan bahwa kebijakan kepercayaan diperbarui untuk menggunakan prinsip CodeCatalyst layanan.

Tambahkan peran IAM ke koneksi akun (konsol)

1. Di dalam AWS Management Console, pastikan Anda masuk dengan akun yang sama yang ingin Anda kelola.
2. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
3. Arahkan ke CodeCatalyst ruang Anda. Pilih Pengaturan, lalu pilih Akun AWS.
4. Pilih nama CodeCatalyst tampilan Amazon dari koneksi akun Anda, lalu pilih Kelola peran dari AWS Management Console.

   Peran Tambahkan IAM ke halaman CodeCatalyst ruang Amazon ditampilkan.
5. Lakukan salah satu hal berikut ini:

   - Untuk membuat peran layanan yang berisi kebijakan izin dan kebijakan kepercayaan untuk peran pengembang, pilih Buat peran administrator CodeCatalyst pengembangan di IAM. Peran akan memiliki nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* dengan pengenal unik ditambahkan. Untuk informasi selengkapnya tentang kebijakan peran dan peran, lihatMemahami peran CodeCatalystWorkflowDevelopmentRole-***`spaceName`***layanan.

     Pilih Buat peran pengembangan.
   - Untuk menambahkan peran yang telah Anda buat di IAM, pilih Tambahkan peran IAM yang ada. Di Pilih peran IAM yang ada, pilih peran dari daftar drop-down.

     Pilih Tambahkan peran.

   Halaman terbuka di AWS Management Console. Anda mungkin perlu masuk untuk mengakses halaman.
6. Di panel navigasi halaman CodeCatalyst spasi Amazon, pilih Spasi.

   Untuk mengakses halaman secara langsung, masuk ke Amazon CodeCatalyst Spaces AWS Management Console di https://console.aws.amazon.com/codecatalyst/home/.

7. Pilih akun yang ditambahkan untuk CodeCatalyst ruang Anda. Halaman koneksi ditampilkan.
8. Pada halaman koneksi, di bawah peran IAM yang tersedia CodeCatalyst, lihat daftar peran IAM yang ditambahkan ke akun Anda. Pilih peran IAM Associate untuk CodeCatalyst.
9. Pada pop-up peran Associate an IAM, di ARN Peran, masukkan Nama Sumber Daya Amazon (ARN) dari peran IAM yang ingin Anda kaitkan dengan ruang Anda. CodeCatalyst

   Di bawah Tujuan, pilih tujuan peran yang menjelaskan cara Anda ingin menggunakan peran dalam koneksi akun Anda. Tentukan `RUNNER` peran yang Anda gunakan untuk menjalankan tindakan dalam alur kerja. Tentukan `SERVICE` peran yang Anda gunakan untuk mengakses layanan lain.

   Anda dapat menentukan lebih dari satu tujuan.

   > **Note**
   >
   > Memilih tujuan untuk peran ARN diperlukan.

10. Pilih Associate peran IAM. Ulangi langkah-langkah ini untuk peran IAM tambahan.

Menambahkan koneksi akun dan peran IAM ke lingkungan penerapan Anda

Untuk mengakses AWS sumber daya, seperti Amazon ECS atau AWS Lambda sumber daya untuk penerapan, tindakan CodeCatalyst build dan deploy memerlukan peran IAM dengan izin untuk mengakses sumber daya tersebut. Dengan administrator Space atau peran pengguna Power, Anda dapat menghubungkan CodeCatalyst akun Anda ke Akun AWS tempat sumber daya Anda dibuat. Anda kemudian menambahkan peran IAM ke koneksi akun Anda. Untuk tindakan penerapan, Anda kemudian harus menambahkan peran IAM ke lingkungan. CodeCatalyst

Anda harus menambahkan peran IAM yang ingin Anda gunakan dengan lingkungan penerapan di proyek Anda. Menambahkan peran ke koneksi akun tidak menambahkan peran dan koneksi ke lingkungan penerapan proyek. Untuk menambahkan koneksi akun dan peran IAM ke lingkungan penerapan Anda, pastikan koneksi dan peran akun dibuat seperti yang dijelaskan secara rinci. Langkah 4: Tambahkan peran IAM ke koneksi Anda

Kemudian, gunakan halaman Lingkungan di CodeCatalyst konsol untuk menambahkan koneksi akun dan peran IAM ke lingkungan penerapan dalam proyek.

> **Note**
>
> Anda hanya menambahkan peran IAM ke lingkungan jika peran IAM digunakan untuk CodeCatalyst tindakan yang memerlukan peran IAM. Semua tindakan alur kerja yang memerlukan peran IAM, termasuk tindakan build, harus menggunakan lingkungan. CodeCatalyst

Untuk menambahkan koneksi akun dan peran IAM ke lingkungan penerapan

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek dengan lingkungan penerapan tempat Anda ingin menambahkan koneksi akun dan peran IAM.
3. Perluas CI/CD, lalu pilih Lingkungan.
4. Pilih lingkungan Anda, dan kemudian tab tambahan ditampilkan.
5. Pilih tab Akun AWS koneksi. Di bawah nama Koneksi, akun yang telah ditambahkan ke lingkungan, jika ada, terdaftar.
6. Pilih Kaitkan Akun AWS. <environment_name>Tampilan halaman Associate Akun AWS with.
7. Di bawah Koneksi, pilih nama koneksi akun dengan peran IAM yang ingin Anda tambahkan. Pilih Kaitkan.

Melihat koneksi akun

Anda dapat melihat daftar koneksi Anda dan melihat detail tentang setiap koneksi.

Anda harus memiliki peran administrator Space atau Power user untuk mengelola koneksi untuk ruang Anda.

Untuk melihat semua koneksi untuk sebuah CodeCatalyst spasi

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang dengan koneksi akun yang ingin Anda lihat.
3. Pilih tab AWS akun.
4. Di bawah AWS akun, lihat daftar koneksi akun untuk ruang tersebut, termasuk ID akun dan status untuk setiap koneksi.

Untuk melihat detail koneksi akun

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda. Pilih Pengaturan, lalu pilih Akun AWS.
3. Di nama CodeCatalyst tampilan Amazon, pilih nama koneksi. Pada halaman Detail, lihat daftar peran IAM yang terkait dengan koneksi bersama dengan detail lainnya.

Menghapus akun dari spasi (in CodeCatalyst)

Anda dapat menghapus koneksi akun yang tidak lagi Anda perlukan. Untuk prosedur ini, Anda akan menggunakan CodeCatalyst untuk menghapus koneksi akun yang sebelumnya telah Anda tambahkan ke ruang Anda. Ini menghapus koneksi akun dari ruang Anda, asalkan akun tersebut bukan akun penagihan untuk ruang tersebut.

> **Important**
>
> Setelah koneksi akun dihapus, Anda tidak dapat menghubungkannya kembali. Anda harus membuat koneksi akun baru dan kemudian mengaitkan peran dan lingkungan IAM, atau mengatur penagihan, sesuai kebutuhan.

Akun penagihan harus ditunjuk untuk CodeCatalyst ruang Anda, meskipun penggunaan untuk ruang tersebut tidak akan melebihi tingkat Gratis. Sebelum Anda dapat menghapus spasi untuk akun yang merupakan akun penagihan yang ditunjuk, Anda perlu menambahkan akun lain untuk ruang Anda. Lihat Mengelola penagihan di Panduan CodeCatalyst Administrator Amazon.

> **Important**
>
> Meskipun Anda dapat menggunakan langkah-langkah ini untuk menghapus akun, ini tidak disarankan. Akun juga dapat diatur untuk mendukung alur kerja di CodeCatalyst.

Untuk mengelola koneksi akun untuk ruang Anda, Anda harus memiliki peran administrator Space atau Power user.

Untuk menghapus koneksi akun

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.

2. Arahkan ke CodeCatalyst ruang Anda. Pilih Pengaturan, lalu pilih Akun AWS.
3. Di bawah nama CodeCatalyst tampilan Amazon, pilih pemilih di sebelah koneksi akun yang ingin Anda hapus.
4. Pilih Hapus Akun AWS. Konfirmasikan penghapusan dengan memasukkan nama di bidang, lalu pilih Hapus.

   Spanduk sukses ditampilkan, dan koneksi akun dihapus dari daftar koneksi.

Mengkonfigurasi akun penagihan untuk spasi

Akun penagihan harus ditunjuk untuk CodeCatalyst ruang Anda, meskipun penggunaan untuk ruang tersebut tidak akan melebihi tingkat Gratis.

Untuk mengonfigurasi akun penagihan, lihat Penagihan di Panduan CodeCatalyst Administrator. Anda dapat menggunakan halaman untuk CodeCatalyst in AWS untuk menghapus akun yang telah ditambahkan ke spasi. Untuk prosedur ini, menggunakan izin administratif untuk akun tertentu yang Anda kelola, Anda masuk ke halaman Amazon CodeCatalyst Spaces di AWS Management Console untuk menghapus Akun AWS dari ruang Anda. Untuk menghapus akun yang merupakan akun penagihan yang ditunjuk untuk CodeCatalyst ruang Anda, pastikan untuk terlebih dahulu menentukan akun penagihan baru.

Akun yang telah dihapus dapat ditambahkan lagi nanti, tetapi Anda harus membuat koneksi baru antara akun dan ruang. Anda harus mengaitkan kembali peran IAM apa pun ke akun yang ditambahkan.

Mengkonfigurasi peran IAM untuk akun yang terhubung

Anda membuat peran di AWS Identity and Access Management (IAM) untuk akun yang ingin Anda tambahkan. CodeCatalyst Jika Anda menambahkan akun penagihan, Anda tidak perlu membuat peran.

Di Anda Akun AWS, Anda harus memiliki izin untuk membuat peran untuk yang ingin Akun AWS Anda tambahkan ke ruang Anda. Untuk informasi selengkapnya tentang peran dan kebijakan IAM, termasuk referensi IAM dan kebijakan contoh, lihat. Identity and Access Management dan Amazon CodeCatalyst Untuk informasi selengkapnya tentang kebijakan kepercayaan dan prinsip layanan yang digunakan, lihat. CodeCatalyst Memahami model CodeCatalyst kepercayaan

Masuk CodeCatalyst, Anda harus masuk dengan peran administrator Space untuk menyelesaikan langkah-langkah menambahkan akun (dan peran, jika berlaku) ke ruang Anda.

Anda dapat menambahkan peran ke koneksi akun Anda dengan menggunakan salah satu metode berikut.

- Untuk membuat peran layanan yang berisi kebijakan izin dan kebijakan kepercayaan untuk CodeCatalystWorkflowDevelopmentRole-***spaceName***peran tersebut, lihatPeran CodeCatalystWorkflowDevelopmentRole-***spaceName***.
- Untuk contoh membuat peran dan menambahkan kebijakan untuk membuat proyek dari cetak biru, lihat. Membuat peran IAM dan menggunakan kebijakan CodeCatalyst kepercayaan
- Untuk daftar contoh kebijakan peran yang akan digunakan saat membuat peran IAM Anda, lihatBerikan akses ke AWS sumber daya proyek dengan peran IAM.
- Untuk langkah-langkah mendetail untuk membuat peran untuk tindakan alur kerja, lihat tutorial alur kerja untuk tindakan tersebut sebagai berikut:
  - Tutorial: Unggah artefak ke Amazon S3
  - Tutorial: Menyebarkan aplikasi tanpa server menggunakan AWS CloudFormation
  - Tutorial: Menyebarkan aplikasi ke Amazon ECS
  - Tutorial: Kode lint menggunakan GitHub Action dalam alur kerja

Topik

Peran CodeCatalystWorkflowDevelopmentRole-***spaceName***

Anda membuat peran pengembang sebagai peran 1-klik di IAM. Anda harus memiliki peran administrator Space atau Power user di ruang tempat Anda ingin menambahkan akun. Anda juga harus memiliki izin administratif untuk yang ingin Akun AWS Anda tambahkan.

Sebelum memulai prosedur di bawah ini, Anda harus masuk ke akun yang sama AWS Management Console dengan yang ingin Anda tambahkan ke CodeCatalyst ruang Anda. Jika tidak, konsol akan mengembalikan kesalahan akun yang tidak dikenal.

Untuk membuat dan menambahkan CodeCatalyst CodeCatalystWorkflowDevelopmentRole-*spaceName*

1. Sebelum Anda mulai di CodeCatalyst konsol, buka AWS Management Console, dan kemudian pastikan Anda masuk dengan yang sama Akun AWS untuk ruang Anda.
2. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
3. Arahkan ke CodeCatalyst ruang Anda. Pilih Pengaturan, lalu pilih Akun AWS.
4. Pilih tautan untuk Akun AWS tempat Anda ingin membuat peran. Halaman Akun AWS detail ditampilkan.
5. Pilih Kelola peran dari AWS Management Console.

   Peran Tambahkan IAM ke halaman CodeCatalyst ruang Amazon terbuka di. AWS Management Console Ini adalah halaman CodeCatalyst spasi Amazon. Anda mungkin perlu masuk untuk mengakses halaman.
6. Pilih Buat peran administrator CodeCatalyst pengembangan di IAM. Opsi ini membuat peran layanan yang berisi kebijakan izin dan kebijakan kepercayaan untuk peran pengembangan. Peran akan memiliki nama`CodeCatalystWorkflowDevelopmentRole-`*`spaceName`*. Untuk informasi selengkapnya tentang kebijakan peran dan peran, lihatMemahami peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan.

   > **Note**
   >
   > Peran ini hanya disarankan untuk digunakan dengan akun pengembang dan menggunakan kebijakan `AdministratorAccess` AWS terkelola, memberikan akses penuh untuk membuat kebijakan dan sumber daya baru dalam hal ini Akun AWS.
7. Pilih Buat peran pengembangan.
8. Pada halaman koneksi, di bawah peran IAM yang tersedia untuk CodeCatalyst, lihat `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran dalam daftar peran IAM yang ditambahkan ke akun Anda.
9. Untuk kembali ke ruang Anda, pilih Buka Amazon CodeCatalyst.

Peran AWSRoleForCodeCatalystSupport

Anda membuat peran dukungan sebagai peran 1-klik di IAM. Anda harus memiliki peran administrator Space atau Power user di ruang tempat Anda ingin menambahkan akun. Anda juga harus memiliki izin administratif untuk yang ingin Akun AWS Anda tambahkan.

Sebelum memulai prosedur di bawah ini, Anda harus masuk ke akun yang sama AWS Management Console dengan yang ingin Anda tambahkan ke CodeCatalyst ruang Anda. Jika tidak, konsol akan mengembalikan kesalahan akun yang tidak dikenal.

Untuk membuat dan menambahkan CodeCatalyst AWSRoleForCodeCatalystSupport

1. Sebelum Anda mulai di CodeCatalyst konsol, buka AWS Management Console, dan kemudian pastikan Anda masuk dengan yang sama Akun AWS untuk ruang Anda.
2. Arahkan ke CodeCatalyst ruang Anda. Pilih Pengaturan, lalu pilih Akun AWS.
3. Pilih tautan untuk Akun AWS tempat Anda ingin membuat peran. Halaman Akun AWS detail ditampilkan.
4. Pilih Kelola peran dari AWS Management Console.

   Peran Tambahkan IAM ke halaman CodeCatalyst ruang Amazon terbuka di. AWS Management Console Ini adalah halaman Amazon CodeCatalyst Spaces. Anda mungkin perlu masuk untuk mengakses halaman.
5. Di bawah detail CodeCatalyst spasi, pilih peran Tambah CodeCatalyst Dukungan. Opsi ini membuat peran layanan yang berisi kebijakan izin dan kebijakan kepercayaan untuk peran pengembangan pratinjau. Peran akan memiliki nama AWSRoleForCodeCatalystSupportdengan pengenal unik ditambahkan. Untuk informasi selengkapnya tentang kebijakan peran dan peran, lihatMemahami peran AWSRoleForCodeCatalystSupportlayanan.
6. Pada halaman Add role for CodeCatalyst Support, biarkan default dipilih, lalu pilih Create role.
7. Di bawah peran IAM yang tersedia CodeCatalyst, lihat `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran dalam daftar peran IAM yang ditambahkan ke akun Anda.
8. Untuk kembali ke ruang Anda, pilih Buka Amazon CodeCatalyst.

Membuat peran IAM dan menggunakan kebijakan CodeCatalyst kepercayaan

Peran IAM yang akan digunakan CodeCatalyst dengan Akun AWS koneksi harus dikonfigurasi untuk menggunakan kebijakan kepercayaan yang disediakan di sini. Gunakan langkah-langkah ini untuk membuat peran IAM dan lampirkan kebijakan yang memungkinkan Anda membuat proyek dari cetak biru. CodeCatalyst

Sebagai alternatif, Anda dapat membuat peran layanan yang berisi kebijakan izin dan kebijakan kepercayaan untuk `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut. Untuk informasi selengkapnya, lihat Menambahkan peran IAM ke koneksi akun.

1. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Peran, lalu pilih Buat peran.
3. Pilih Kebijakan kepercayaan khusus.
4. Di bawah formulir Kebijakan kepercayaan khusus, tempel kebijakan kepercayaan berikut.

```
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
             "Principal": {
                "Service": [
                    "codecatalyst-runner.amazonaws.com",
                    "codecatalyst.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:codecatalyst:::space/spaceId/project/
*"
                }
            }
        }
    ]
```

5. Pilih Selanjutnya.
6. Di bawah Tambahkan izin, cari dan pilih kebijakan khusus yang telah Anda buat di IAM.
7. Pilih Selanjutnya.
8. Untuk nama Peran, masukkan nama untuk peran, misalnya: `codecatalyst-project-role`
9. Pilih Buat peran.
10. Salin peran Nama Sumber Daya Amazon (ARN). Anda harus memberikan informasi ini saat menambahkan peran ke koneksi atau lingkungan akun Anda.

Memberikan izin ruang kepada pengguna

Anda dapat mengelola anggota untuk ruang dengan melihat, menambahkan, menghapus, atau mengubah peran bagi pengguna yang bergabung dengan ruang tersebut.

Informasi dalam panduan ini disediakan untuk mengundang dan mengelola pengguna di ruang CodeCatalyst yang mendukung pengguna AWS Builder ID. Untuk mempelajari selengkapnya tentang langkah-langkah menyiapkan dan mengelola ruang yang mendukung federasi identitas, lihat Pengaturan dan administrasi untuk CodeCatalyst spasi di Panduan CodeCatalyst Administrator Amazon.

Melihat anggota di ruang

Anda dapat melihat pengguna di ruang Anda, termasuk informasi tentang nama tampilan, alias, dan peran yang mereka miliki untuk ruang tersebut. Ada tiga peran untuk anggota dalam sebuah ruang:

- Administrator ruang — Peran ini memiliki semua izin CodeCatalyst, termasuk membuat proyek. Hanya tetapkan peran ini kepada pengguna yang perlu mengelola setiap aspek ruang, seperti mengakses semua proyek di ruang tersebut.

  Anda tidak dapat mengubah peran ini nanti tanpa menghapus pengguna terlebih dahulu. Untuk informasi selengkapnya, lihat Peran administrator ruang.
- Pengguna daya - Peran ini adalah peran paling kuat kedua di CodeCatalyst ruang Amazon, tetapi tidak memiliki akses ke proyek di suatu ruang. Ini dirancang untuk pengguna yang harus dapat membuat proyek di ruang dan membantu mengelola pengguna dan sumber daya untuk ruang tersebut. Untuk informasi selengkapnya, lihat Peran pengguna daya.
- Akses terbatas - Peran ini ditetapkan secara default untuk pengguna yang bergabung dengan ruang dengan menerima undangan ke proyek di ruang tersebut. Anggota proyek diberi peran

dalam sebuah proyek. Untuk informasi tentang mengelola anggota proyek, lihatMemberikan izin proyek kepada pengguna.

Tabel administrator Space menunjukkan pengguna dengan peran administrator Space. Pengguna ini tidak ditampilkan di anggota Space karena mereka secara otomatis (implisit) ditugaskan ke semua proyek di ruang dan tidak memiliki peran dalam proyek.

Tabel anggota Space menunjukkan semua anggota dalam ruang yang memiliki peran dalam proyek sementara tidak memiliki peran administrator Space.

Pengguna ditampilkan berdasarkan apakah pengguna memiliki peran administrator Space CodeCatalyst sebagai berikut:

- Pengguna dengan peran administrator Space yang kemudian menerima undangan dan peran proyek tidak akan ditampilkan dalam tabel Anggota Spasi di bawah spasi atau pada tabel Anggota proyek di bawah proyek. Mereka akan terus ditampilkan di tabel administrator Space di kedua tempat. Dalam setiap proyek, semua pengguna dengan peran administrator Space ditampilkan dalam tabel administrator Space proyek untuk proyek tersebut.
- Pengguna yang menerima undangan proyek untuk bergabung dengan peran proyek ditambahkan ke ruang dengan peran akses terbatas. Jika peran pengguna nanti berubah ke peran administrator Space, tetapi juga akan berpindah dari tabel anggota Spasi ke tabel administrator Space. Di bawah proyek, pengguna akan berpindah dari tabel anggota Project ke tabel administrator Space.

Untuk melihat pengguna dan peran di ruang Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda.

   > **Tip**
   >
   > Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Pilih Pengaturan, lalu pilih Anggota.

   Pengguna yang merupakan anggota ruang ditampilkan di tabel anggota Space.

> **Tip**
>
> Jika Anda memiliki peran administrator Space, Anda dapat melihat proyek mana yang telah langsung diundang. Arahkan ke Pengaturan proyek untuk proyek, lalu pilih Proyek saya.

Di kolom Status, berikut ini adalah nilai yang valid:

- Diundang - CodeCatalyst mengirim undangan tetapi pengguna belum menerima atau menolak.
- Anggota — Pengguna menerima undangan.

Mengundang pengguna langsung ke spasi

Anda dapat mengundang pengguna langsung ke CodeCatalyst ruang Anda. Ini berguna ketika Anda ingin mengundang pengguna tersebut untuk membantu Anda mengelola ruang dengan menetapkan peran administrator Space atau Power user. Menetapkan salah satu peran tersebut kepada pengguna lain dapat membantu Anda mendistribusikan tanggung jawab mengelola ruang di lebih banyak orang tanpa harus mengundang pengguna ini ke proyek apa pun.

> **Note**
>
> Anda harus memiliki peran administrator Space atau Power user untuk mengundang anggota.

Tabel administrator Space menunjukkan pengguna dengan peran administrator Space. Pengguna ini tidak ditampilkan dalam tabel anggota Space karena mereka secara otomatis (implisit) ditugaskan ke semua proyek di ruang dan tidak memiliki peran dalam proyek.

Anggota yang menerima undangan proyek ditambahkan ke ruang secara default. Tabel anggota Proyek menunjukkan semua anggota dalam ruang yang memiliki peran dalam proyek.

Untuk informasi selengkapnya tentang cara menerima undangan dan masuk untuk pertama kalinya, lihatSiapkan dan masuk ke CodeCatalyst.

Untuk mengundang pengguna ke ruang Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda.
3. Pilih Pengaturan, lalu pilih Anggota.
4. Pilih Undang.
5. Masukkan email orang yang ingin Anda undang untuk bergabung dengan ruang Anda. Di Peran, pilih peran yang ingin Anda tetapkan kepada pengguna tersebut di ruang.
6. Pilih Undang

Membatalkan undangan untuk ruang

Jika Anda ingin membatalkan undangan untuk bergabung dengan ruang yang Anda kirim baru-baru ini, dan belum diterima, Anda dapat membatalkannya.

Untuk mengelola undangan ruang, Anda harus memiliki peran administrator Space atau Power user.

Untuk membatalkan undangan anggota ruang

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda.

> **Tip**
>
> Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Pilih Pengaturan, lalu pilih Anggota.
4. Verifikasi bahwa anggota memiliki status Diundang.

> **Note**
>
> Anda hanya dapat membatalkan undangan yang belum diterima.

5. Pilih opsi di sebelah baris dengan anggota yang diundang, lalu pilih Batalkan undangan.
6. Jendela konfirmasi ditampilkan. Pilih Batalkan undangan untuk mengonfirmasi.

Mengubah peran anggota luar angkasa

Anda dapat mengubah peran yang ditetapkan untuk anggota ruang Anda. Anda harus memiliki peran administrator Space untuk mengubah peran pengguna di ruang.

Tabel administrator Space menunjukkan pengguna dengan peran administrator Space. Pengguna ini tidak ditampilkan dalam tabel anggota Space karena mereka secara otomatis (implisit) ditugaskan ke semua proyek di ruang.

Untuk mengubah peran pengguna di ruang Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda.

> **Tip**
>
> Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Pilih Pengaturan, lalu pilih Anggota.
4. Di tabel anggota Spasi, pilih pengguna yang perannya ingin Anda ubah. Pilih Ubah peran.

Menghapus anggota spasi

Anda dapat menghapus anggota ruang Anda ketika mereka tidak perlu mengakses sumber daya ruang apa pun. Anda harus memiliki peran administrator Space untuk menghapus anggota dari spasi.

Tabel administrator Space menunjukkan pengguna dengan peran administrator Space. Pengguna ini tidak ditampilkan dalam tabel anggota Space karena mereka secara otomatis (implisit) ditugaskan ke semua proyek di ruang dan tidak memiliki peran dalam proyek. Anda hanya dapat langsung menghapus anggota ruang Anda di tabel ini.

Untuk menghapus pengguna dari tabel anggota Project

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda.

> **Tip**
>
> Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Pilih Pengaturan, lalu pilih Anggota.
4. Pilih pengguna di tabel anggota Project. Pilih Hapus.

> **Note**
>
> Menghapus anggota dari ruang akan menghapus pengguna dari semua proyek di ruang, bersama dengan izin yang terkait dengan sumber daya dalam proyek tersebut.

Menghapus atau mengubah peran untuk pengguna dengan peran administrator Space

Anda dapat menghapus atau mengubah peran untuk pengguna dengan peran administrator Space untuk ruang Anda.

Anda harus memiliki peran administrator Space untuk menghapus pengguna dengan peran administrator Space dari spasi. Mengubah peran untuk pengguna dengan peran administrator Space pada dasarnya menghapus pengguna dari tabel administrator Space. Jika pengguna tersebut tidak memiliki peran proyek dalam proyek apa pun di ruang, menghapus peran administrator Space dari pengguna akan menghapus pengguna dari ruang.

> **Note**
>
> Sebagai pengguna dengan peran administrator Space, Anda tidak dapat menghapus sendiri. Hubungi pengguna lain dengan peran administrator Space.

Untuk menghapus pengguna dengan peran administrator Space dari tabel anggota Space

> **Note**
>
> Untuk pengguna yang belum ditambahkan secara eksplisit ke proyek, mereka tidak memiliki peran proyek apa pun (Administrator proyek atau Kontributor). Jika peran administrator

Space adalah satu-satunya peran pengguna, maka pengguna akan dihapus dari ruang seluruhnya.

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang tempat Anda ingin menghapus atau mengubah peran untuk pengguna dengan peran administrator Space.
3. Pilih Pengaturan, lalu pilih Anggota.
4. Lihat status undangan untuk daftar anggota, dan pastikan daftar tersebut tidak berisi undangan tertunda yang tidak sah ke ruang (status Diundang).

   Important

   Sebelum menghapus pengguna dengan peran administrator Space, Anda harus memverifikasi bahwa tidak ada undangan yang tertunda telah dimulai.

5. Pilih tab Anggota. Di tabel Administrator ruang, pilih pengguna, lalu pilih Hapus.

   Pada kotak dialog Hapus anggota, lakukan salah satu hal berikut.

   - Pilih opsi untuk menghapus hanya peran administrator Space pengguna. Pilih Hapus.

     Important

     Jika pengguna tidak memiliki peran lain yang ditetapkan, maka mengubah peran dari administrator Space akan menghapus pengguna dari ruang.

   - Pilih opsi untuk menghapus pengguna dengan peran administrator Space dari ruang dan semua proyeknya. Pilih Hapus.
6. Segarkan tab Anggota. Pengguna secara otomatis ditambahkan ke daftar anggota proyek di setiap proyek di mana pengguna memiliki keanggotaan melalui peran proyek. Jika peran administrator Space adalah satu-satunya peran pengguna, maka pengguna akan dihapus dari ruang seluruhnya.

Mengizinkan akses ruang menggunakan tim

Setelah Anda membuat spasi, Anda dapat menambahkan tim. Tim memungkinkan Anda mengelompokkan pengguna sehingga mereka dapat berbagi izin dan mengelola proyek, pelacakan masalah, peran, dan sumber daya di dalamnya CodeCatalyst.

Anda harus memiliki peran administrator Space untuk mengelola tim.

Tim juga dikelola di tingkat proyek/ruang di. CodeCatalyst Untuk mempelajari lebih lanjut tentang tim di ruang/proyek, lihat. Mengizinkan akses ruang menggunakan tim

Topik

Membuat tim

Sebuah tim dapat memiliki izin peran, seperti Power user, di ruang. Tim juga dapat memiliki izin proyek, seperti Administrator proyek, dalam proyek. Tim dapat dikaitkan dengan banyak proyek dengan peran yang berbeda untuk setiap proyek. Anda dapat mengelola tim di mana anggota tim adalah pengguna individu untuk ruang ID AWS Pembangun atau grup SSO untuk ruang yang mendukung federasi identitas.

Pada halaman anggota untuk pengguna ruang dan proyek, pengguna dapat memiliki banyak peran. Pengguna dengan beberapa peran akan menampilkan indikator ketika mereka memiliki beberapa peran, dan mereka akan ditampilkan dengan peran dengan izin terbanyak terlebih dahulu.

> **Note**
>
> Jika ruang Anda mendukung federasi identitas, Anda harus sudah menyiapkan pengguna SSO atau grup SSO Anda di Pusat Identitas IAM.

Cara Anda mengelola anggota tim tergantung pada bagaimana Anda akan menambah dan menghapus pengguna. Ada dua opsi untuk mengelola anggota tim:

- Menambahkan pengguna secara langsung — Anda menambah atau menghapus pengguna satu per satu. Misalnya, Anda menambahkan pengguna ke tim dengan memilih pengguna AWS Builder ID atau pengguna SSO yang sudah disiapkan di IAM Identity Center. Ketika Anda memilih untuk mengelola anggota tim dengan menambahkan pengguna AWS Builder ID atau pengguna SSO secara langsung, opsi untuk menggunakan grup SSO tidak akan lagi tersedia.
- Gunakan grup SSO — Anda mengelola anggota tim melalui grup SSO yang sudah disiapkan di Pusat Identitas IAM. Ketika Anda memilih untuk mengelola anggota tim dengan menggunakan grup SSO, opsi untuk menambahkan pengguna secara langsung tidak lagi tersedia.

Anda harus memiliki peran administrator Space untuk mengelola tim.

Untuk membuat tim

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda. Pilih Pengaturan, lalu pilih Tim.
3. Pilih Buat tim.
4. Dalam nama Tim, masukkan nama deskriptif untuk tim Anda.

   > **Note**
   >
   > Nama tim harus unik di ruang Anda.

   (Opsional) Dalam deskripsi Tim, masukkan deskripsi untuk tim Anda.
5. Di bawah Peran ruang, pilih peran dari daftar peran ruang yang tersedia CodeCatalyst yang ingin Anda tetapkan ke tim. Peran tersebut akan diwarisi oleh semua anggota tim.

   - Administrator ruang - Untuk detailnya, lihatPeran administrator ruang.

- Akses terbatas - Untuk detailnya, lihatPeran akses terbatas.
- Pengguna daya - Untuk detailnya, lihatPeran pengguna daya.

6. Dalam keanggotaan Tim, pilih salah satu dari berikut ini untuk memilih metode penambahan anggota ke tim.

   - Pilih Tambahkan anggota secara langsung untuk mengelola pengguna secara individual. Ini termasuk menambahkan pengguna AWS Builder ID untuk spasi atau menambahkan pengguna SSO untuk ruang yang mendukung federasi identitas.
   - Pilih Gunakan Grup SSO untuk memilih grup SSO yang telah Anda atur di Pusat Identitas IAM.

     Di Grup SSO, pilih kotak di samping grup yang ingin Anda tambahkan. Anda dapat menambahkan hingga lima grup SSO.

   > **Note**
   >
   > Anda tidak dapat mengubah ini nanti. Ketika Anda memilih untuk mengelola anggota tim dengan menambahkan pengguna AWS Builder ID atau pengguna SSO secara langsung, opsi untuk menggunakan grup SSO tidak akan lagi tersedia. Ketika Anda memilih untuk mengelola anggota tim dengan menggunakan grup SSO, opsi untuk menambahkan pengguna secara langsung tidak lagi tersedia.

7. Pilih Buat.

   > **Note**
   >
   > Ketika Anda memilih untuk menggunakan grup SSO, perhatikan bahwa pengguna dalam grup SSO tidak ditarik pada pembuatan tim. Pengguna harus masuk CodeCatalyst sebelum terlihat dalam daftar.

Melihat tim

Di CodeCatalyst, Anda dapat melihat proyek dan peran untuk tim Anda. Pada halaman anggota, Anda dapat melihat peran proyek dan daftar pengguna. Untuk tim tipe grup SSO, Anda juga akan dapat melihat daftar grup SSO yang terkait dengan tim.

Untuk melihat tim

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda. Pilih Pengaturan, lalu pilih Tim.
3. Dalam peran Space, lihat peran yang ditetapkan ke tim untuk ruang ini.
4. Pada tab Peran proyek, lihat peran proyek dan proyek yang ditetapkan ke tim untuk setiap CodeCatalyst proyek di ruang di mana tim telah ditambahkan sebagai anggota (hanya untuk ruang AWS Builder ID).
5. Pada tab Anggota, lihat daftar anggota yang ditetapkan ke tim.
6. Pada tab Grup SSO, lihat daftar grup SSO yang ditetapkan ke tim (untuk spasi yang hanya mendukung federasi identitas).

Memberikan peran ruang untuk tim

Tim adalah cara untuk mengelompokkan pengguna sehingga Anda dapat memberikan dan mengelola akses tim ke proyek CodeCatalyst. Misalnya, Anda dapat menggunakan tim untuk mengelola peran dan izin pengguna dengan cepat dengan memberi tim kemampuan untuk mengelola ruang bagi pengguna.

Sebuah tim dapat memiliki izin peran, seperti Power user, di ruang. Anda dapat mengubah peran ruang untuk tim, tetapi perhatikan bahwa semua anggota tim akan mewarisi izin tersebut.

Anda harus memiliki peran administrator Space untuk mengelola tim.

Mengubah peran ruang untuk tim

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda. Pilih Pengaturan, lalu pilih Tim.
3. Di Tindakan, pilih Ubah peran ruang. Anda dapat mengubah peran ruang ke salah satu dari berikut ini. Ini mengubah peran untuk semua anggota tim.
   - Administrator ruang - Untuk detailnya, lihatPeran administrator ruang.
   - Akses terbatas - Untuk detailnya, lihatPeran akses terbatas.
   - Pengguna daya - Untuk detailnya, lihatPeran pengguna daya.
4. Pilih Simpan.

Memberikan peran proyek untuk tim di tingkat ruang angkasa

Sebuah tim CodeCatalyst mirip dengan pengguna di mana anggota tim dapat memiliki izin peran, seperti administrator Proyek, dalam proyek. Perubahan peran akan diterapkan ke tim, dan semua anggota tim akan mewarisi izin tersebut. Anda dapat memilih satu peran untuk setiap proyek yang akan secara otomatis diberikan kepada tim.

Anda harus memiliki peran administrator Space untuk mengelola tim.

Untuk menambah atau mengubah peran proyek

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda. Pilih Pengaturan, lalu pilih Tim.
3. Pilih tab Peran proyek.
4. Untuk mengubah peran, pilih pemilih di samping proyek dalam daftar ini, lalu pilih Ubah peran. Untuk menambahkan peran, pilih Tambahkan peran proyek. Di Proyek, pilih proyek yang ingin Anda tambahkan dan di Peran, pilih peran. Pilih salah satu peran proyek yang tersedia:
   - Administrator proyek - Untuk detailnya, lihatPeran administrator proyek.
   - Kontributor - Untuk detailnya, lihatPeran kontributor.
   - Peninjau - Untuk detailnya, lihatPeran pengulas.
   - Hanya baca - Untuk detailnya, lihatBaca saja peran.
5. Pilih Simpan.

Untuk menghapus peran proyek

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda. Pilih Pengaturan, lalu pilih Tim.
3. Pilih tab Peran proyek.
4. Pilih peran yang ingin Anda hapus.

> **Important**
>
> Menghapus peran dari tim akan menghapus izin terkait untuk semua pengguna di tim.

5. Pilih Simpan.

Menambahkan pengguna ke tim secara langsung

Anda dapat menambahkan anggota tim ke tim Anda. Saat Anda menambahkan pengguna, pengguna baru akan mewarisi izin dari semua peran yang ada di tim.

Apakah ruang Anda disiapkan untuk dukungan pengguna AWS Builder ID atau federasi identitas, Anda dapat mengatur ruang Anda untuk menambahkan pengguna secara langsung.

> **Note**
>
> Ketika ruang Anda diatur untuk mengelola anggota tim menggunakan grup SSO, opsi untuk menggunakan Tambah pengguna secara langsung tidak tersedia. Untuk menggunakan grup SSO, lihatMenambahkan grup SSO ke tim.

Anda harus memiliki peran administrator Space untuk mengelola tim.

Untuk menambahkan pengguna secara langsung

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda. Pilih Pengaturan, lalu pilih Tim.
3. Pilih tab Anggota.
4. Pilih Tambah anggota.

   > **Note**
   >
   > Pengguna yang ditambahkan ke tim harus sudah menjadi anggota ruang. Anda tidak dapat menambahkan atau mengundang anggota tim yang bukan anggota ruang.

5. Pilih pengguna di bidang drop-down, lalu pilih Simpan. Pilih pengguna AWS Builder ID atau pengguna SSO yang sudah diatur di IAM Identity Center.

Menghapus pengguna dari tim secara langsung

Anda dapat menghapus anggota tim dari tim Anda. Semua izin tidak akan lagi diwarisi oleh pengguna. Anda dapat menambahkan pengguna kembali ke tim nanti.

> **Note**
>
> Saat Anda menghapus anggota tim, izin terkait akan dihapus untuk pengguna dari semua proyek dan sumber daya di ruang tersebut.

Anda harus memiliki peran administrator Space untuk mengelola tim.

Untuk menghapus anggota tim

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda. Pilih Pengaturan, lalu pilih Tim.
3. Pilih tab Anggota.
4. Pilih pemilih di sebelah pengguna yang ingin Anda hapus, lalu pilih Hapus.
5. Masukkan hapus di bidang input, lalu pilih Hapus.

Menambahkan grup SSO ke tim

Jika ruang Anda dikonfigurasi sebagai ruang dengan pengguna dan grup SSO yang dikelola di Pusat Identitas IAM, Anda dapat menambahkan grup SSO yang akan bergabung dengan ruang sebagai tim terpisah.

> **Note**
>
> Bila Anda memilih untuk mengelola anggota tim dengan menambahkan pengguna AWS Builder ID atau pengguna SSO secara langsung, opsi untuk menggunakan grup SSO tidak tersedia. Untuk menambahkan pengguna secara langsung, lihatMenambahkan pengguna ke tim secara langsung.

Anda harus memiliki peran administrator Space untuk mengelola tim.

Untuk menambahkan grup SSO sebagai tim

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pada halaman untuk ruang Anda, pilih Tim. Pilih tab grup SSO.
3. Pilih grup SSO yang ingin Anda tambahkan. Anda dapat menambahkan hingga lima grup SSO.

Menghapus tim

Anda dapat menghapus tim yang tidak lagi Anda butuhkan.

> **Note**
>
> Saat Anda menghapus tim, izin terkait akan dihapus untuk semua anggota tim dari semua proyek dan sumber daya di ruang tersebut.

Anda harus memiliki peran administrator Space untuk mengelola tim.

Menghapus tim

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda. Pilih Pengaturan, lalu pilih Tim.
3. Di Tindakan, pilih Hapus tim. Ini mengubah peran untuk seluruh tim.
4. Pilih Hapus.

Memungkinkan akses ruang untuk sumber daya mesin

Sumber daya mesin adalah sumber daya khusus CodeCatalyst yang diberikan izin untuk proyek atau spasi di CodeCatalyst.

> **Note**
>
> Istilah sumber daya mesin tidak mengacu pada infrastruktur cloud seperti instans Amazon EC2, tetapi sebaliknya dimaksudkan untuk merujuk ke cetak biru atau sumber daya alur kerja dengan izin untuk ruang atau proyek.

Sumber daya mesin mewakili identitas Anda dari sumber daya resmi Anda saat mengakses CodeCatalyst melalui SSO. Sumber daya mesin digunakan untuk memberikan izin ke sumber daya di ruang, seperti cetak biru dan alur kerja. Anda dapat melihat sumber daya mesin di ruang Anda, dan Anda dapat memilih untuk mengaktifkan atau menonaktifkan sumber daya mesin untuk ruang Anda. Misalnya, Anda mungkin ingin menonaktifkan sumber daya mesin untuk mengelola akses dan kemudian mengaktifkannya kembali nanti.

Operasi ini tersedia untuk sumber daya mesin dalam kasus di mana sumber daya mesin perlu dicabut atau dinonaktifkan. Misalnya, jika Anda mencurigai kredenal mungkin telah disusupi, Anda dapat menonaktifkan sumber daya mesin. Umumnya, operasi ini tidak perlu digunakan.

Anda harus memiliki peran administrator Space untuk melihat halaman ini dan mengelola sumber daya mesin di tingkat ruang.

Sumber daya mesin juga dikelola di tingkat proyek di CodeCatalyst. Untuk mempelajari lebih lanjut tentang tim dalam proyek, lihatMemungkinkan akses ruang untuk sumber daya mesin .

Topik

Melihat akses ruang untuk sumber daya mesin

Anda dapat melihat daftar sumber daya mesin yang digunakan di ruang Anda.

Anda harus memiliki peran administrator Space untuk mengelola sumber daya mesin.

Untuk melihat sumber daya mesin

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda, lalu pilih Pengaturan. Pilih Sumber daya mesin.
3. Di drop-down, pilih Tindakan alur kerja untuk hanya melihat sumber daya mesin untuk alur kerja. Pilih Blueprint untuk hanya melihat sumber daya mesin untuk cetak biru.

   Anda juga dapat memfilter nama menggunakan bidang Filter.

Menonaktifkan akses ruang untuk sumber daya mesin

Anda dapat memilih untuk menonaktifkan sumber daya mesin yang digunakan di ruang Anda.

> **Important**
>
> Menonaktifkan sumber daya mesin akan menghapus semua izin ke semua cetak biru atau alur kerja terkait di ruang tersebut.

Anda harus memiliki peran administrator Space untuk mengelola sumber daya mesin.

Untuk menonaktifkan sumber daya mesin

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda, lalu pilih Pengaturan. Pilih Sumber daya mesin.
3. Pilih salah satu dari berikut ini.

> **Important**
>
> Menonaktifkan sumber daya mesin akan menghapus semua izin ke semua cetak biru atau alur kerja terkait di ruang tersebut.

   - Untuk menonaktifkan satu per satu, pilih pemilih di sebelah satu atau beberapa sumber daya mesin yang ingin Anda nonaktifkan. Pilih Nonaktifkan, lalu pilih Sumber daya ini.
   - Untuk menonaktifkan semua sumber daya, pilih Nonaktifkan, lalu pilih Semua sumber daya.
   - Untuk menonaktifkan semua tindakan alur kerja, pilih Nonaktifkan, lalu pilih Semua tindakan alur kerja.
   - Untuk menonaktifkan semua cetak biru, pilih Nonaktifkan, lalu pilih Semua cetak biru.

Mengaktifkan akses ruang untuk sumber daya mesin

Anda dapat memilih untuk mengaktifkan sumber daya mesin yang digunakan di ruang Anda dan yang telah dinonaktifkan.

Anda harus memiliki peran administrator Space untuk mengelola sumber daya mesin.

Untuk mengaktifkan sumber daya mesin

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda, lalu pilih Pengaturan. Pilih Sumber daya mesin.
3. Pilih salah satu dari berikut ini.

   - Untuk mengaktifkan satu per satu, pilih pemilih di sebelah satu atau beberapa sumber daya mesin yang ingin Anda aktifkan. Pilih Aktifkan, lalu pilih Sumber daya ini.
   - Untuk mengaktifkan semua sumber daya, pilih Aktifkan, lalu pilih Semua sumber daya.

- Untuk mengaktifkan semua tindakan alur kerja, pilih Aktifkan, lalu pilih Semua tindakan alur kerja.
- Untuk mengaktifkan semua cetak biru, pilih Aktifkan, lalu pilih Semua cetak biru.

Mengelola Lingkungan Pengembang untuk suatu ruang

Semua Lingkungan Dev dibuat sebagai bagian dari proyek dalam suatu ruang. Anggota luar angkasa dapat membuat Lingkungan Dev mereka sendiri dalam sebuah proyek di tingkat repositori sumber. Administrator ruang angkasa kemudian dapat menggunakan CodeCatalyst konsol Amazon untuk melihat, mengedit, menghapus, dan menghentikan Lingkungan Pengembang atas nama anggota ruang angkasa. Singkatnya, administrator ruang mempertahankan Lingkungan Dev di tingkat ruang.

Pertimbangan untuk mengelola Lingkungan Dev

- Anda harus memiliki peran administrator Space untuk melihat halaman Lingkungan Dev di bawah Pengaturan dan untuk mengelola Lingkungan Dev di tingkat ruang.
- Anggota luar angkasa mengelola Lingkungan Pengembang yang mereka buat dalam proyek melalui CodeCatalyst akun mereka. Saat mengelola Lingkungan Dev sebagai administrator ruang, Anda memelihara sumber daya ini atas nama anggota ruang angkasa.
- Dev Environments default ke konfigurasi komputasi dan penyimpanan tertentu. Untuk informasi tentang penagihan dan tarif untuk meningkatkan konfigurasi Anda, lihat halaman CodeCatalyst harga Amazon.

Untuk pertimbangan lain tentang Lingkungan Dev, termasuk menghentikan menjalankan instance, konfigurasi komputasi default, memutakhirkan komputasi, mengeluarkan biaya, dan mengonfigurasi batas waktu, lihat. Menulis dan memodifikasi kode dengan Dev Environments di CodeCatalyst

Topik

- Melihat Lingkungan Pengembang untuk ruang Anda
- Mengedit Lingkungan Pengembang untuk ruang Anda
- Menghentikan Lingkungan Pengembang untuk ruang Anda
- Menghapus Lingkungan Pengembang untuk ruang Anda

Melihat Lingkungan Pengembang untuk ruang Anda

Anda dapat melihat jenis, status, dan detail untuk semua Lingkungan Pengembang di ruang Anda. Untuk informasi selengkapnya tentang membuat dan menjalankan Lingkungan Dev, lihatMembuat Lingkungan Dev.

Anda harus memiliki peran administrator Space untuk melihat halaman ini dan mengelola Lingkungan Dev di tingkat ruang.

Untuk melihat Lingkungan Pengembang di ruang Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.

   > **Tip**
   >
   > Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Pilih Pengaturan, lalu pilih Lingkungan Pengembang.

   Halaman ini mencantumkan semua Lingkungan Pengembang di ruang Anda. Anda dapat melihat nama Resource, alias resource jika berlaku, jenis IDE, Compute dan Storage default atau yang dikonfigurasi, dan Timeout yang dikonfigurasi untuk setiap Lingkungan Dev.

Mengedit Lingkungan Pengembang untuk ruang Anda

Anda dapat mengedit konfigurasi untuk Lingkungan Dev, seperti panjang batas waktu yang dikonfigurasi, jika ada, agar Lingkungan Dev yang tidak aktif berhenti berjalan. Untuk informasi selengkapnya tentang mengedit Lingkungan Pengembang, lihatMengedit Lingkungan Pengembang.

Anda harus memiliki peran administrator Space untuk melihat halaman ini dan mengelola Lingkungan Dev di tingkat ruang.

Untuk mengedit Lingkungan Dev di ruang Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.

> **Tip**
>
> Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Pilih Pengaturan, lalu pilih Lingkungan Pengembang.
4. Pilih pemilih di sebelah Lingkungan Dev yang ingin Anda kelola. Pilih Edit.
5. Buat perubahan yang Anda inginkan pada batas waktu komputasi atau ketidakaktifan untuk Lingkungan Pengembang.
6. Pilih Simpan.

Menghentikan Lingkungan Pengembang untuk ruang Anda

Anda dapat menghentikan Lingkungan Dev yang sedang berjalan sebelum menjadi idle jika Lingkungan Dev dikonfigurasi untuk memiliki batas waktu. Jika tidak, Lingkungan Pengembang dengan batas waktu yang telah berlalu sudah akan dihentikan. Untuk informasi lebih lanjut tentang menghentikan Lingkungan Pengembang, lihatMenghentikan Lingkungan Pengembang.

Anda harus memiliki peran administrator Space untuk melihat halaman ini dan mengelola Lingkungan Dev di tingkat ruang.

Untuk menghentikan Lingkungan Dev di ruang Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.

> **Tip**
>
> Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Pilih Pengaturan, lalu pilih Lingkungan Pengembang.
4. Pilih pemilih di sebelah Lingkungan Dev yang ingin Anda kelola. Pilih Berhenti.

Menghapus Lingkungan Pengembang untuk ruang Anda

Anda dapat menghapus Lingkungan Pengembang yang tidak lagi diperlukan atau yang tidak lagi memiliki pemilik. Untuk informasi selengkapnya tentang pertimbangan untuk menghapus Lingkungan Pengembang, lihat. Menghapus Lingkungan Dev

Anda harus memiliki peran administrator Space untuk melihat halaman ini dan mengelola Lingkungan Dev di tingkat ruang.

Untuk menghapus Lingkungan Dev di ruang Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.

> **Tip**
>
> Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Pilih Pengaturan, lalu pilih Lingkungan Pengembang.
4. Pilih pemilih di sebelah Lingkungan Dev yang ingin Anda kelola. Pilih Hapus. Untuk mengonfirmasi, ketik`delete`, lalu pilih Hapus.

Kuota untuk spasi

Tabel berikut menjelaskan kuota dan batas untuk spasi di Amazon CodeCatalyst. Untuk informasi selengkapnya tentang kuota di Amazon CodeCatalyst, lihatKuota untuk CodeCatalyst.

Jumlah maksimum saluran Slack untuk suatu ruang	500
Jumlah maksimum undangan untuk alamat email	25
Jumlah maksimum undangan untuk pengguna	500
Jumlah maksimum ruang aktif per pengguna per Wilayah AWS	5

Jumlah maksimum kreasi ruang per Wilayah per bulan per pengguna	5
Jumlah maksimum grup SSO untuk sebuah tim	5
Jumlah maksimum tim untuk spasi	100
Jumlah maksimum pengguna untuk sebuah tim	1000
Deskripsi ruang	Deskripsi ruang bersifat opsional. Jika ditentukan, panjangnya harus antara 0 dan 200 karakter. Mereka dapat berisi kombinasi huruf, angka, spasi, titik, garis bawah, koma, tanda hubung, dan karakter khusus berikut: `? & $ % + = / \ ; : \n \t \r`
Nama ruang	Nama ruang harus unik di seberang CodeCatalyst. Anda tidak dapat menggunakan kembali nama spasi yang dihapus. Nama spasi harus antara 3 dan 63 karakter panjangnya. Mereka juga harus mulai dengan karakter alfanumerik. Nama spasi dapat berisi kombinasi huruf, angka, titik, garis bawah, dan tanda hubung. Mereka tidak dapat berisi salah satu karakter berikut: `! ? @ # $ % ^ & * ( ) + = { } [ ] | /\ > < ~ ` ' " ; :`

Mengatur pekerjaan dengan proyek-proyek di CodeCatalyst

Anda menggunakan proyek di Amazon CodeCatalyst untuk membangun ruang kolaborasi di mana tim pengembangan dapat melakukan tugas pengembangan dengan alur kerja dan repositori shared continuous integration/continuous delivery (CI/CD). Saat membuat proyek, Anda dapat menambahkan, memperbarui, atau menghapus sumber daya. Anda juga dapat memantau kemajuan kerja tim Anda. Anda dapat memiliki beberapa proyek dalam satu ruang.

Ruang CodeCatalyst di terdiri dari proyek. Anda dapat melihat setiap proyek di dalam ruang Anda, tetapi Anda hanya dapat menggunakan proyek yang menjadi anggotanya. Saat Anda membuat proyek, peran default untuk proyek Anda akan dibuat, yang Anda tetapkan kepada pengguna yang Anda undang ke proyek Anda.

- Siapa pun yang ditugaskan ke proyek dengan peran proyek, seperti peran Kontributor, dapat mengakses sumber daya proyek, seperti repositori sumber.
- Siapa pun dengan administrator Space Administrator atau peran proyek dapat mengirim undangan untuk bergabung dengan proyek.
- Pengguna dengan peran administrator Project dapat melacak aktivitas, status, dan setelan lainnya di seluruh sumber daya bersama.
- Pengguna dengan peran akses terbatas dapat mengelola tugas proyek untuk fitur, perbaikan kode, dan pengujian sebagai bagian dari alur kerja CI/CD.

  Alur kerja digunakan untuk membangun, menguji, dan merilis atau memperbarui aplikasi sebagai pipa CI/CD. Anda dapat merakit alur kerja dengan menambahkan tindakan yang mentransfer dan bekerja pada artefak sumber Anda. Saat Anda menjalankan tindakan, sumber daya cloud proyek Anda digunakan untuk menyediakan kemampuan komputasi sesuai permintaan untuk tindakan alur kerja Anda. Anda dapat mengonfigurasi lebih banyak alur kerja CI/CD berdasarkan aktivitas dan output yang ingin Anda atur. Misalnya, Anda dapat membuat alur kerja hanya untuk tindakan build dan pengujian, di mana Anda dapat melihat hasil pengujian dan menyelesaikan alur kerja tanpa penerapan saat memperbaiki bug. Kemudian, Anda dapat membuat alur kerja lain untuk membangun dan menerapkan aplikasi Anda ke lingkungan pementasan.

Saat Anda membuat proyek, Anda dapat menggunakan cetak biru untuk membuat proyek yang berisi kode sampel dan membuat sumber daya, atau Anda dapat memulai dengan proyek kosong. Jika Anda membuat proyek menggunakan cetak biru, cetak biru yang Anda pilih menentukan sumber daya yang ditambahkan ke proyek Anda dan alat yang CodeCatalyst membuat atau

mengonfigurasi sehingga Anda dapat melacak dan menggunakan sumber daya proyek Anda. Anda dapat menambahkan atau menghapus sumber daya secara manual setelah Anda membuat proyek.

Setiap proyek melacak aktivitas proyek sebagai daftar peristiwa oleh pengguna, seperti saat proyek dibuat atau sumber daya dimodifikasi. Aktivitas proyek dipantau dan dikumpulkan di tingkat ruang. Untuk informasi selengkapnya tentang bekerja dengan data aktivitas, lihatMelihat semua proyek dalam satu ruang.

Jika proyek Anda menggunakan AWS sumber daya, Anda dapat menghubungkan CodeCatalyst akun Anda ke AWS akun yang memiliki izin administratif untuk mengintegrasikan sumber daya untuk proyek Anda.

Anda dapat menambahkan repositori sumber, masalah, dan sumber daya lainnya ke proyek Anda setelah Anda membuatnya. Anda harus memiliki peran administrator Space untuk membuat proyek.

Membuat proyek

Dengan CodeCatalyst proyek, Anda dapat melakukan tugas pengembangan dengan alur kerja dan repositori shared continuous integration/continuous delivery (CI/CD), mengelola sumber daya, melacak masalah, dan menambahkan pengguna.

Sebelum Anda membuat proyek, Anda harus memiliki peran administrator Space atau Power user.

Topik

- Membuat proyek dengan cetak biru
- Membuat proyek kosong di Amazon CodeCatalyst
- Membuat proyek dengan repositori pihak ketiga yang ditautkan
- Menambahkan sumber daya dan tugas ke proyek yang dibuat

Membuat proyek dengan cetak biru

Anda dapat menyediakan semua sumber daya proyek dan kode sampel Anda dengan cetak biru proyek. Untuk informasi tentang cetak biru, lihat. Membuat proyek yang komprehensif dengan CodeCatalyst cetak biru

Untuk membuat proyek dengan cetak biru

1. Di CodeCatalyst konsol, arahkan ke ruang tempat Anda ingin membuat proyek.

2. Di dasbor ruang, pilih Buat proyek.
3. Pilih Mulai dengan cetak biru.
4. Dari cetak biru atau tab CodeCatalyst Blueprints Space, pilih cetak biru, lalu pilih Berikutnya.
5. Di bawah Nama proyek Anda, masukkan nama yang ingin Anda tetapkan ke proyek Anda dan nama sumber daya yang terkait. Nama harus unik di dalam ruang Anda.
6. (Opsional) Secara default, kode sumber yang dibuat oleh cetak biru disimpan dalam repositori. CodeCatalyst Atau, Anda dapat memilih untuk menyimpan kode sumber cetak biru di repositori pihak ketiga. Untuk informasi selengkapnya, lihat Tambahkan fungsionalitas ke proyek dengan ekstensi di CodeCatalyst.

   Lakukan salah satu hal berikut tergantung pada penyedia repositori pihak ketiga yang ingin Anda gunakan:

   - GitHub repositori: Hubungkan akun. GitHub

     Pilih menu tarik-turun Advanced, pilih GitHub sebagai penyedia repositori, lalu pilih GitHub akun tempat Anda ingin menyimpan kode sumber yang dibuat oleh cetak biru.

     > **Note**
     >
     > Jika Anda menghubungkan GitHub akun, Anda harus membuat koneksi pribadi untuk membuat pemetaan identitas antara CodeCatalyst identitas dan GitHub identitas Anda. Untuk informasi selengkapnya, lihat Koneksi pribadi dan Mengakses GitHub sumber daya dengan koneksi pribadi.

   - Repositori Bitbucket: Hubungkan ruang kerja Bitbucket.

     Pilih menu tarik-turun Advanced, pilih Bitbucket sebagai penyedia repositori, lalu pilih ruang kerja Bitbucket tempat Anda ingin menyimpan kode sumber yang dibuat oleh cetak biru.
7. Di bawah sumber daya Proyek, konfigurasikan parameter cetak biru. Tergantung pada cetak biru, Anda mungkin memiliki opsi untuk memberi nama nama repositori sumber.
8. (Opsional) Untuk melihat file definisi dengan pembaruan berdasarkan pilihan parameter proyek yang Anda buat, pilih Lihat kode atau Lihat alur kerja dari Hasilkan pratinjau proyek.
9. (Opsional) Pilih Lihat detail dari kartu cetak biru untuk melihat detail spesifik tentang cetak biru, seperti ikhtisar arsitektur cetak biru, koneksi dan izin yang diperlukan, dan jenis sumber daya yang dibuat cetak biru.
10. Pilih Buat proyek.

Membuat proyek kosong di Amazon CodeCatalyst

Anda dapat membuat proyek kosong tanpa sumber daya dan secara manual menambahkan sumber daya yang Anda inginkan di lain waktu.

Sebelum Anda membuat proyek, Anda harus memiliki peran administrator Space atau Power user.

Untuk membuat proyek kosong

1. Arahkan ke ruang tempat Anda ingin membuat proyek.
2. Di dasbor ruang, pilih Buat proyek.
3. Pilih Mulai dari awal.
4. Di bawah Berikan nama untuk proyek Anda, masukkan nama yang ingin Anda tetapkan ke proyek Anda. Nama harus unik di dalam ruang Anda.
5. Pilih Buat proyek.

Membuat proyek dengan repositori pihak ketiga yang ditautkan

Anda dapat menyimpan kode sumber proyek Anda di penyedia pihak ketiga pilihan dan tetap menggunakan semua CodeCatalyst fitur seperti cetak biru, manajemen siklus hidup, alur kerja, dan banyak lagi. Untuk melakukan ini, Anda dapat membuat CodeCatalyst proyek baru yang menautkan ke GitHub repositori atau Bitbucket. Anda kemudian dapat menggunakan repositori sumber tertaut di proyek Anda CodeCatalyst .

Sebelum Anda membuat CodeCatalyst proyek, Anda harus memiliki peran administrator Space atau Power user. Untuk informasi selengkapnya, lihat Menciptakan ruang dan Mengundang pengguna langsung ke spasi .

Untuk membuat proyek di tautan CodeCatalyst tersebut ke repositori sumber di GitHub akun atau ruang kerja Bitbucket, Anda harus menyelesaikan tiga tugas berikut:

1. Instal GitHub repositori atau ekstensi Bitbucket. Anda diminta di situs eksternal untuk terhubung dan menyediakan CodeCatalyst akses ke repositori pihak ketiga Anda, yang dilakukan sebagai bagian dari langkah berikutnya.

> **Important**
>
> Untuk menginstal GitHub repositori atau ekstensi repositori Bitbucket ke CodeCatalyst ruang Anda, Anda harus masuk dengan akun yang memiliki peran administrator Space di ruang tersebut.

2. Hubungkan GitHub akun Anda atau ruang kerja Bitbucket ke. CodeCatalyst

> **Important**
>
> Untuk menghubungkan GitHub akun atau ruang kerja Bitbucket ke CodeCatalyst ruang Anda, Anda harus menjadi administrator sumber pihak ketiga dan administrator CodeCatalyst Space.

> **Important**
>
> Setelah Anda menginstal ekstensi repositori, repositori apa pun yang Anda tautkan CodeCatalyst akan memiliki kode mereka diindeks dan disimpan. CodeCatalyst Ini akan membuat kode dapat dicari di. CodeCatalyst Untuk lebih memahami perlindungan data untuk kode Anda saat menggunakan repositori tertaut di CodeCatalyst, lihat Perlindungan data di CodeCatalyst Panduan Pengguna Amazon.

> **Note**
>
> Jika Anda menggunakan koneksi ke GitHub akun, Anda harus membuat koneksi pribadi untuk membuat pemetaan identitas antara CodeCatalyst identitas dan GitHub identitas Anda. Untuk informasi selengkapnya, lihat Koneksi pribadi dan Mengakses GitHub sumber daya dengan koneksi pribadi.

3. Buat CodeCatalyst proyek yang ditautkan ke repositori atau GitHub repositori Bitbucket Anda.

> **Important**
>
> Meskipun Anda dapat menautkan repositori GitHub atau Bitbucket sebagai Kontributor, Anda hanya dapat memutuskan tautan repositori pihak ketiga sebagai administrator Space

atau administrator Project. Untuk informasi selengkapnya, lihat Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.

> **Note**
>
> - A hanya GitHub dapat dihubungkan ke satu CodeCatalyst proyek dalam satu ruang.
> - Anda tidak dapat menggunakan GitHub repositori kosong atau diarsipkan dengan proyek. CodeCatalyst
> - Anda tidak dapat menautkan GitHub epositori yang memiliki nama yang sama dengan repositori dalam proyek. CodeCatalyst
> - Ekstensi GitHub repositori tidak kompatibel dengan repositori GitHub Enterprise Server.

Untuk informasi selengkapnya, lihat Tambahkan fungsionalitas ke proyek dengan ekstensi di CodeCatalyst.

Untuk menginstal ekstensi pihak ketiga

1. Arahkan ke ruang tempat Anda ingin membuat proyek.
2. Di dasbor ruang, pilih Buat proyek.
3. Pilih Bawa kode Anda sendiri.
4. Di bawah Tautkan repositori yang ada, pilih GitHub repositori atau repositori Bitbucket tergantung pada penyedia repositori pihak ketiga yang ingin Anda gunakan. Anda diminta untuk menghubungkan GitHub akun atau ruang kerja Bitbucket. Jika ekstensi pihak ketiga belum diinstal, prompt instalasi akan ditampilkan.
5. Jika diminta, pilih Instal. Tinjau izin yang diperlukan oleh ekstensi, dan jika Anda ingin melanjutkan, pilih Instal lagi.

Setelah Anda menginstal ekstensi pihak ketiga, langkah selanjutnya adalah menghubungkan GitHub akun atau ruang kerja Bitbucket ke ruang Anda CodeCatalyst .

Untuk menghubungkan penyedia GitHub repositori Bitbucket Anda ke CodeCatalyst

Lakukan salah satu hal berikut tergantung pada ekstensi pihak ketiga yang Anda pilih untuk dikonfigurasi:

- GitHub repositori: Connect ke akun. GitHub

  1. Pilih Connect GitHub account untuk pergi ke situs eksternal untuk GitHub.
  2. Masuk ke GitHub akun Anda menggunakan GitHub kredensi Anda, lalu pilih akun tempat Anda ingin menginstal Amazon. CodeCatalyst

     > **Tip**
     >
     > Jika sebelumnya Anda telah menghubungkan GitHub akun ke ruang tersebut, Anda tidak akan diminta untuk mengotorisasi ulang. Anda malah akan melihat kotak dialog yang menanyakan di mana Anda ingin menginstal ekstensi jika Anda adalah anggota atau kolaborator di lebih dari satu GitHub ruang, atau halaman konfigurasi untuk CodeCatalyst aplikasi Amazon jika Anda hanya memiliki satu GitHub ruang. Konfigurasikan aplikasi untuk akses repositori yang ingin Anda izinkan, lalu pilih Simpan. Jika tombol Simpan tidak aktif, buat perubahan pada konfigurasi, lalu coba lagi.

  3. Pilih apakah Anda ingin mengizinkan CodeCatalyst untuk mengakses semua repositori saat ini dan masa depan, atau pilih GitHub repositori tertentu yang ingin Anda gunakan. CodeCatalyst Opsi default adalah memasukkan semua GitHub repositori di GitHub akun, termasuk repositori future yang akan diakses oleh. CodeCatalyst
  4. Tinjau izin yang diberikan CodeCatalyst, lalu pilih Instal.

  Setelah menghubungkan GitHub akun Anda CodeCatalyst, Anda akan dibawa ke halaman detail ekstensi GitHub repositori, di mana Anda dapat melihat dan mengelola GitHub akun yang terhubung dan repositori tertaut GitHub .

- Repositori Bitbucket: Connect ke ruang kerja Bitbucket.

  1. Pilih Connect Bitbucket workspace untuk pergi ke situs eksternal untuk Bitbucket.
  2. Masuk ke ruang kerja Bitbucket Anda menggunakan kredensi Bitbucket Anda.
  3. Dari menu tarik-turun Otorisasi untuk ruang kerja, pilih ruang kerja Bitbucket yang ingin Anda akses, lalu pilih Grant CodeCatalyst access.

     > **Tip**
     >
     > Jika sebelumnya Anda telah menghubungkan ruang kerja Bitbucket ke ruang tersebut, Anda tidak akan diminta untuk mengotorisasi ulang. Sebagai gantinya, Anda akan

melihat dialog yang menanyakan di mana Anda ingin menginstal ekstensi jika Anda anggota atau kolaborator di lebih dari satu ruang kerja Bitbucket, atau halaman konfigurasi untuk CodeCatalyst aplikasi Amazon jika Anda hanya memiliki satu ruang kerja Bitbucket. Konfigurasikan aplikasi untuk akses ruang kerja yang ingin Anda izinkan, lalu pilih Grant access. Jika tombol akses Grant tidak aktif, buat perubahan pada konfigurasi, lalu coba lagi.

Setelah menghubungkan ruang kerja Bitbucket CodeCatalyst, Anda akan dibawa ke halaman detail ekstensi repositori Bitbucket, tempat Anda dapat melihat dan mengelola ruang kerja Bitbucket yang terhubung dan repositori Bitbucket yang ditautkan.

Setelah menghubungkan penyedia repositori pihak ketiga Anda CodeCatalyst, Anda dapat menautkan repositori pihak ketiga ke proyek Anda. CodeCatalyst

Untuk membuat proyek Anda

1. Pada halaman Buat proyek, pilih GitHub akun atau ruang kerja Bitbucket yang Anda sambungkan.
2. Bergantung pada penyedia repositori pihak ketiga yang Anda sambungkan, pilih repositori atau menu tarik-turun GitHub repositori repositori Bitbucket untuk melihat repositori pihak ketiga, lalu pilih repositori yang ingin Anda tautkan ke proyek Anda.
3. Di kolom input teks Nama proyek Anda, masukkan nama yang ingin Anda tetapkan ke proyek Anda. Nama harus unik di dalam ruang Anda.
4. Pilih Buat proyek.

Setelah menginstal GitHub repositori atau ekstensi repositori Bitbucket, menghubungkan penyedia sumber daya Anda, dan menautkan repositori pihak ketiga Anda ke CodeCatalyst proyek Anda, Anda dapat menggunakannya dalam alur kerja dan Lingkungan Dev. CodeCatalyst Anda juga dapat membuat repositori pihak ketiga di GitHub akun yang terhubung atau ruang kerja Bitbucket dengan kode yang dihasilkan dari cetak biru. Untuk informasi selengkapnya, lihat Memulai alur kerja secara otomatis setelah peristiwa repositori pihak ketiga dan Membuat Lingkungan Dev.

Menambahkan sumber daya dan tugas ke proyek yang dibuat

Setelah proyek Anda siap, Anda dapat menambahkan sumber daya dan tugas.

- Untuk mempelajari alur kerja CI/CD yang dibuat dengan proyek Anda, lihat. Memulai dengan alur kerja
- Untuk bekerja dengan tindakan build yang serupa dengan yang ada di proyek baru Anda yang menerapkan artefak build ke bucket Amazon S3, lihat dan. Membangun dengan alur kerja Tutorial: Unggah artefak ke Amazon S3
- Untuk memulai dengan proyek kosong dan bekerja dengan menerapkan aplikasi tanpa server serupa dengan penerapan AWS CloudFormation tumpukan, lihat. Tutorial: Menyebarkan aplikasi tanpa server menggunakan AWS CloudFormation
- Untuk menambahkan papan perencanaan masalah, lihatLacak dan atur pekerjaan dengan masalah di CodeCatalyst.
- Untuk melihat ikhtisar proyek, status proyek, aktivitas tim terbaru, dan pekerjaan yang ditetapkan, lihatMendapatkan daftar proyek.
- Untuk melihat kode sumber atau membuat permintaan tarik, lihatSimpan dan berkolaborasi pada kode dengan repositori sumber di CodeCatalyst.
- Untuk mengatur notifikasi yang mengirim peringatan status agar alur kerja berhasil atau gagal berjalan, lihat. Mengelola notifikasi di Amazon CodeCatalyst
- Untuk mengundang anggota ke proyek Anda, lihatMemberikan izin proyek kepada pengguna.
- Untuk mengatur Lingkungan Dev, lihatMenulis dan memodifikasi kode dengan Dev Environments di CodeCatalyst.

Mendapatkan daftar proyek

Dari CodeCatalyst ruang Anda, Anda dapat melihat detail setiap proyek di mana Anda memiliki izin proyek.

Untuk melihat proyek, Anda harus menjadi anggota proyek atau memiliki peran administrator Space untuk ruang tersebut.

Jika Anda belum membuat proyek, lihatMembuat proyek. Anda harus memiliki peran administrator Space untuk ruang tempat Anda ingin membuat proyek.

- Pada ikhtisar proyek, Anda dapat melihat anggota proyek, repositori sumber, alur kerja berjalan, permintaan tarik terbuka, Lingkungan Pengembang proyek, dan masalah.
- Di bawah pengaturan proyek, Anda dapat melihat dan mengelola detail proyek, menghapus proyek, mengundang anggota baru ke proyek, mengelola anggota proyek, dan mengonfigurasi pemberitahuan.

Melihat tugas proyek dan Lingkungan Pengembang

Untuk melihat ringkasan tugas proyek, seperti masalah terbuka dan permintaan tarik yang ditetapkan untuk Anda atau dibuat oleh Anda, dan Lingkungan Dev terkait proyek, gunakan konsol.

Untuk melihat proyek, Anda harus menjadi anggota proyek atau memiliki peran administrator Space untuk ruang tersebut.

Untuk melihat repositori sumber Anda, alur kerja berjalan, masalah, permintaan tarik, Lingkungan Pengembang, dan masalah

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang dengan proyek yang ingin Anda lihat. Di bawah Proyek, pilih proyek Anda.
3. Di panel navigasi, pilih Ikhtisar.
4. Lihat tugas proyek yang diberikan kepada Anda dan dibuat oleh Anda.
   - Lihat Member+Lihat semua daftar untuk melihat daftar anggota proyek.
   - Lihat kartu Repositori untuk melihat repositori sumber yang terkait dengan proyek.
   - Lihat kartu Workflow running untuk melihat alur kerja yang terkait dengan proyek.
   - Lihat kartu Open pull requests untuk melihat ringkasan status repositori kode, selain permintaan tarik yang ditetapkan untuk Anda dan dibuat oleh Anda.
   - Lihat kartu My Dev Environments untuk melihat ringkasan Lingkungan Pengembang yang terkait dengan proyek.
   - Lihat kartu Masalah untuk melihat ringkasan tugas atau tugas yang ditetapkan yang Anda buat.

Melihat semua proyek dalam satu ruang

Dalam daftar Proyek untuk ruang Anda, Anda dapat melihat semua proyek di mana Anda memiliki izin.

Untuk melihat ringkasan tugas proyek, seperti masalah terbuka dan permintaan tarik yang ditetapkan untuk Anda atau dibuat oleh Anda, dan Lingkungan Dev terkait proyek, gunakan konsol.

Untuk melihat proyek, Anda harus menjadi anggota proyek atau memiliki peran administrator Space untuk ruang tersebut.

Untuk melihat repositori sumber Anda, alur kerja berjalan, masalah, permintaan tarik, Lingkungan Pengembang, dan masalah

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang dengan proyek yang ingin Anda lihat. Di bawah Proyek, pilih proyek Anda.
3. Di panel navigasi, pilih Pengaturan proyek.
4. Lihat nama proyek, jalur, ID proyek, dan deskripsi.

Melihat pengaturan proyek

Dalam pengaturan Proyek, Anda dapat melihat anggota proyek, repositori sumber, alur kerja berjalan, permintaan tarik terbuka, Lingkungan Pengembang proyek, dan masalah.

Untuk melihat ringkasan tugas proyek, seperti masalah terbuka dan permintaan tarik yang ditetapkan untuk Anda atau dibuat oleh Anda, dan Lingkungan Dev terkait proyek, gunakan konsol.

Untuk melihat repositori sumber Anda, alur kerja berjalan, masalah, permintaan tarik, Lingkungan Pengembang, dan masalah

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang dengan proyek yang ingin Anda lihat. Di bawah Proyek, pilih proyek Anda.
3. Di panel navigasi, pilih Pengaturan proyek.
4. Lihat nama proyek, jalur, ID proyek, dan deskripsi.

Mengubah ke proyek yang berbeda di CodeCatalyst

Untuk mengubah ke proyek lain, gunakan konsol untuk memilih dari daftar proyek yang dapat Anda akses.

Untuk mengubah ke proyek yang berbeda

1. Di CodeCatalyst konsol, pilih pemilih proyek di bagian atas.
2. Perluas drop-down dan pilih proyek yang ingin Anda navigasikan.

Menghapus proyek

Anda dapat menghapus proyek untuk menghapus semua akses ke sumber daya proyek. Anda harus memiliki peran administrator Space atau administrator Proyek untuk menghapus proyek. Setelah Anda menghapus proyek, anggota proyek tidak akan dapat mengakses sumber daya proyek, dan alur kerja apa pun yang diminta oleh repositori sumber pihak ketiga akan dihentikan.

Untuk menghapus proyek Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang dengan proyek yang ingin Anda lihat. Di bawah Proyek, pilih proyek Anda.
3. Di panel navigasi, pilih Pengaturan proyek.
4. Pilih Hapus proyek.
5. Masukkan **`delete`** untuk mengonfirmasi penghapusan.
6. Pilih Hapus proyek.

Memberikan izin proyek kepada pengguna

Anda dapat mengelola anggota dalam proyek Anda menggunakan CodeCatalyst konsol Amazon. Anda dapat menambah atau menghapus pengguna, mengelola peran anggota saat ini, mengirim undangan untuk bergabung dengan proyek Anda, dan membatalkan undangan yang belum diterima.

Pada halaman anggota untuk pengguna ruang dan proyek, pengguna dapat memiliki banyak peran. Pengguna dengan beberapa peran akan menampilkan indikator ketika mereka memiliki beberapa peran, dan mereka akan ditampilkan dengan peran dengan izin terbanyak terlebih dahulu.

Mendapatkan daftar anggota dan peran proyek mereka

Saat menambahkan pengguna ke project, Anda menetapkan peran yang memberikan izin proyek sebagai berikut:

- Peran administrator Proyek memiliki semua izin dalam proyek. Hanya tetapkan peran ini kepada pengguna yang perlu mengelola setiap aspek proyek, termasuk mengedit pengaturan proyek, mengelola izin proyek, dan menghapus proyek. Untuk informasi selengkapnya, lihat Peran administrator proyek.

- Peran Kontributor memiliki izin yang diperlukan untuk bekerja dalam proyek. Tetapkan peran ini kepada pengguna yang perlu bekerja dengan kode, alur kerja, masalah, dan tindakan dalam proyek. Untuk informasi selengkapnya, lihat Peran kontributor.
- Peran Peninjau memiliki izin peninjauan. Lihat perinciannya di Peran pengulas.
- Peran Baca saja memiliki izin baca. Lihat perinciannya di Baca saja peran.

Anda tidak perlu mengundang pengguna dengan peran administrator Space ke proyek Anda karena mereka sudah memiliki akses implisit ke semua proyek di ruang tersebut.

Saat Anda mengundang pengguna ke proyek Anda (tanpa menetapkan peran administrator Space), pengguna akan ditampilkan dalam tabel Anggota proyek di bawah proyek dan dalam tabel anggota Proyek di bawah spasi.

Untuk melihat pengguna dan peran dalam ruang

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang dengan proyek yang ingin Anda lihat. Di bawah Proyek, pilih proyek Anda.
3. Di panel navigasi, pilih Pengaturan proyek.
4. Pilih tab Anggota.

   Tabel anggota Proyek menunjukkan semua anggota yang memiliki peran dalam proyek.

   > **Tip**
   >
   > Jika Anda memiliki peran administrator Space, Anda dapat melihat proyek mana yang telah langsung diundang. Arahkan ke Pengaturan proyek untuk proyek, lalu pilih Proyek saya.

   Tabel administrator Space menunjukkan pengguna dengan peran administrator Space. Pengguna ini secara otomatis (implisitas) ditugaskan ke semua proyek di ruang dan tidak memiliki peran dalam proyek.

   Di kolom Status, berikut ini adalah nilai yang valid:

   - Diundang - CodeCatalyst mengirim undangan tetapi pengguna belum menerima atau menolak.

- Anggota — Pengguna menerima undangan.

Topik

Mengundang pengguna ke proyek

Anda dapat menggunakan konsol untuk mengundang pengguna ke proyek Anda. Anda dapat mengundang anggota ruang Anda atau menambahkan nama dari luar ruang Anda.

Untuk mengundang pengguna ke proyek Anda, Anda harus masuk dengan peran Administrator proyek atau administrator Space.

Anda tidak perlu mengundang pengguna dengan peran administrator Space ke proyek Anda karena mereka sudah memiliki akses implisit ke semua proyek di ruang tersebut.

Saat Anda mengundang pengguna ke proyek Anda (tanpa menetapkan peran administrator Space), pengguna akan ditampilkan dalam tabel Anggota proyek di bawah proyek dan dalam tabel anggota Proyek di bawah spasi.

Untuk mengundang anggota ke proyek Anda dari tab Pengaturan proyek

1. Arahkan ke proyek Anda.

   > **Tip**
   >
   > Anda dapat memilih proyek mana yang akan dilihat di bilah navigasi atas.

2. Di panel navigasi, pilih Pengaturan proyek.
3. Pilih tab Anggota.
4. Di anggota Project, pilih Undang anggota baru.
5. Ketik alamat email anggota baru, pilih peran untuk anggota ini, lalu pilih Undang. Untuk informasi lebih lanjut tentang peran, lihat Memberikan akses dengan peran pengguna.

Untuk mengundang anggota ke proyek Anda dari halaman ikhtisar Proyek

1. Arahkan ke proyek Anda.

   > **Tip**
   >
   > Anda dapat memilih proyek mana yang akan dilihat di bilah navigasi atas.

2. Pilih tombol Anggota +.
3. Ketik alamat email anggota baru, pilih peran untuk anggota ini, lalu pilih Undang. Untuk informasi lebih lanjut tentang peran, lihat Memberikan akses dengan peran pengguna.

Membatalkan undangan

Jika Anda baru saja mengirim undangan, Anda dapat membatalkannya selama undangan belum diterima.

Untuk mengelola undangan proyek, Anda harus memiliki administrator Proyek atau peran administrator Space.

Untuk membatalkan undangan anggota proyek

1. Arahkan ke proyek tempat Anda mengirim undangan yang ingin Anda batalkan.
2. Di panel navigasi, pilih Pengaturan proyek.
3. Lihat tab Anggota dan verifikasi bahwa anggota memiliki status Diundang.

   > **Note**
   >
   > Anda hanya dapat membatalkan undangan yang belum diterima.

4. Pilih opsi di sebelah baris dengan anggota yang diundang, lalu pilih Batalkan undangan.
5. Jendela konfirmasi ditampilkan. Pilih Batalkan undangan untuk mengonfirmasi.

Menghapus pengguna dari proyek Anda

Anda dapat menggunakan konsol untuk menghapus pengguna dari proyek Anda.

Untuk menghapus pengguna dari proyek Anda, Anda harus masuk dengan peran Administrator proyek atau administrator Space.

> **Note**
>
> Menghapus pengguna dari semua proyek dalam ruang secara otomatis menghapus pengguna dari ruang itu.

Untuk menghapus pengguna dari proyek

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang dengan proyek yang ingin Anda lihat. Di bawah Proyek, pilih proyek Anda.
3. Di panel navigasi, pilih Pengaturan proyek.
4. Pilih tab Anggota.
5. Pilih pemilih di sebelah profil yang ingin Anda hapus, lalu pilih Hapus.
6. Konfirmasikan bahwa Anda ingin menghapus pengguna, lalu pilih Hapus.

Menerima atau menolak undangan untuk suatu proyek

Anda mungkin menerima undangan email untuk bergabung dengan CodeCatalyst proyek Amazon. Anda dapat menerima atau menolak undangan.

Untuk menerima atau menolak undangan

1. Buka email undangan.
2. Pilih tautan proyek di email.
3. Pilih Terima atau Tolak.

   Jika Anda memilih Tolak, email dikirim ke akun manajemen proyek yang memberi tahu mereka bahwa Anda menolak undangan.

Mengizinkan akses proyek menggunakan tim

Setelah Anda membuat proyek, Anda dapat menambahkan tim. Tim memungkinkan Anda mengelompokkan pengguna sehingga mereka dapat berbagi izin dan mengelola proyek, pelacakan masalah, peran, dan sumber daya CodeCatalyst sebagai anggota proyek dan ruang angkasa.

Anda harus memiliki peran administrator Proyek untuk mengelola tim untuk proyek Anda.

Tim juga dikelola di tingkat luar angkasa di CodeCatalyst. Untuk mempelajari lebih lanjut tentang tim di ruang angkasa, lihatMengizinkan akses ruang menggunakan tim.

Topik

Menambahkan tim ke proyek

Anda dapat mengelola tim tempat anggota tim dapat mengakses sumber daya dalam proyek Anda.

Pada halaman anggota untuk pengguna ruang dan proyek, pengguna dapat memiliki banyak peran. Pengguna dengan beberapa peran akan menampilkan indikator ketika mereka memiliki beberapa peran, dan mereka akan ditampilkan dengan peran dengan izin terbanyak terlebih dahulu.

Untuk menambahkan tim

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda. Pilih Pengaturan proyek, lalu pilih Tim.
3. Pilih Tambah tim.
4. Di Tim, pilih tim dari daftar tim yang tersedia.
5. Di bawah Peran proyek, pilih peran dari daftar peran proyek yang tersedia di CodeCatalyst.
   - Administrator proyek — Untuk detailnya, lihatPeran administrator proyek.
   - Kontributor — Untuk detailnya, lihatPeran kontributor.
   - Peninjau — Untuk detailnya, lihatPeran pengulas.
   - Baca saja — Untuk detailnya, lihatBaca saja peran.

6. Pilih Tambah tim.

Memberikan peran proyek untuk tim

Sebuah tim dapat memiliki izin peran, seperti Power user, di ruang. Anda dapat mengubah peran ruang untuk tim, tetapi perhatikan bahwa semua anggota tim akan mewarisi izin tersebut.

Untuk menambah atau mengubah peran proyek

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda. Pilih Pengaturan proyek, lalu pilih Teams.
3. Untuk mengubah peran, pilih pemilih di sebelah tim dalam daftar ini, lalu pilih Ubah peran. Untuk menambahkan peran, pilih Tambahkan peran proyek. Di Proyek, pilih proyek yang ingin Anda tambahkan dan di Peran, pilih peran. Pilih salah satu peran proyek yang tersedia:
   - Administrator proyek - Untuk detailnya, lihatPeran administrator proyek.
   - Kontributor - Untuk detailnya, lihatPeran kontributor.
   - Peninjau - Untuk detailnya, lihatPeran pengulas.
   - Hanya baca - Untuk detailnya, lihatBaca saja peran.
4. Pilih Simpan.

Menghapus peran proyek untuk tim

Di CodeCatalyst, Anda dapat melihat peran proyek untuk tim Anda. Anda juga dapat melihat anggota dalam tim. Anda dapat menghapus peran proyek untuk tim.

Untuk menghapus peran proyek

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang Anda. Pilih Pengaturan proyek, lalu pilih Tim.
3. Pilih tab Peran proyek.
4. Pilih peran yang ingin Anda hapus.

> **Important**
>
> Menghapus peran dari tim akan menghapus izin terkait untuk semua pengguna di tim.

5. Pilih Simpan.

Mengizinkan akses proyek untuk sumber daya mesin

Sumber daya mesin adalah sumber daya khusus CodeCatalystthat yang diberikan izin untuk proyek atau spasi di CodeCatalyst.

> **Note**
>
> Istilah sumber daya mesin tidak mengacu pada infrastruktur cloud seperti instans EC2, tetapi sebaliknya dimaksudkan untuk merujuk ke cetak biru atau sumber daya alur kerja dengan izin untuk ruang atau proyek.

Contoh bekerja dengan sumber daya mesin dalam proyek termasuk mengaktifkan sumber daya cetak biru untuk mengakses proyek atas nama Anda.

Sumber daya mesin mewakili identitas Anda dari sumber daya resmi Anda saat mengakses CodeCatalyst melalui SSO. Sumber daya mesin digunakan untuk memberikan izin ke sumber daya dalam proyek Anda, seperti cetak biru dan alur kerja. Anda dapat melihat sumber daya mesin dalam proyek Anda, dan Anda dapat memilih untuk mengaktifkan atau menonaktifkan sumber daya mesin untuk proyek Anda. Misalnya, Anda mungkin ingin menonaktifkan sumber daya mesin untuk mengelola akses dan kemudian mengaktifkannya kembali nanti.

Operasi ini tersedia untuk sumber daya mesin dalam kasus di mana sumber daya mesin perlu dicabut atau dinonaktifkan. Misalnya, jika Anda mencurigai kredenal mungkin telah disusupi, Anda dapat menonaktifkan sumber daya mesin. Umumnya, operasi ini tidak perlu digunakan.

Anda harus memiliki peran administrator Space atau peran administrator Proyek untuk melihat halaman ini dan mengelola sumber daya mesin di tingkat proyek.

Sumber daya mesin juga dikelola di tingkat ruang di CodeCatalyst. Untuk mempelajari lebih lanjut tentang tim di ruang/proyek, lihat. Memungkinkan akses ruang untuk sumber daya mesin

Topik

- Melihat akses proyek untuk sumber daya mesin
- Menonaktifkan akses proyek untuk sumber daya mesin
- Mengaktifkan akses proyek untuk sumber daya mesin

Melihat akses proyek untuk sumber daya mesin

Anda dapat melihat daftar sumber daya mesin yang digunakan dalam proyek Anda.

Anda harus memiliki peran administrator Space atau peran administrator Proyek.

Untuk melihat sumber daya mesin

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda, lalu pilih Pengaturan proyek. Pilih Sumber daya mesin.
3. Di drop-down, pilih Tindakan alur kerja untuk hanya melihat sumber daya mesin untuk alur kerja. Pilih Blueprint untuk hanya melihat sumber daya mesin untuk cetak biru.

   Anda juga dapat memfilter nama menggunakan bidang Filter.

Menonaktifkan akses proyek untuk sumber daya mesin

Anda dapat memilih untuk menonaktifkan sumber daya mesin yang sedang digunakan dalam proyek Anda.

> **Important**
>
> Menonaktifkan sumber daya mesin akan menghapus semua izin ke semua cetak biru atau alur kerja terkait di ruang tersebut.

Anda harus memiliki peran administrator Space atau peran administrator Proyek.

Untuk menonaktifkan sumber daya mesin

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda, lalu pilih Pengaturan proyek. Pilih Sumber daya mesin.
3. Pilih salah satu dari berikut ini.

   > **Important**
   >
   > Menonaktifkan sumber daya mesin akan menghapus semua izin ke semua cetak biru atau alur kerja terkait di ruang tersebut.

- Untuk menonaktifkan satu per satu, pilih pemilih di sebelah satu atau beberapa sumber daya mesin yang ingin Anda nonaktifkan. Pilih Nonaktifkan, lalu pilih Sumber daya ini.
- Untuk menonaktifkan semua sumber daya, pilih Nonaktifkan, lalu pilih Semua sumber daya.
- Untuk menonaktifkan semua tindakan alur kerja, pilih Nonaktifkan, lalu pilih Semua tindakan alur kerja.
- Untuk menonaktifkan semua cetak biru, pilih Nonaktifkan, lalu pilih Semua cetak biru.

Mengaktifkan akses proyek untuk sumber daya mesin

Anda dapat memilih untuk mengaktifkan sumber daya mesin yang digunakan dalam proyek Anda dan yang telah dinonaktifkan.

Anda harus memiliki peran administrator Space atau peran administrator Proyek.

Untuk mengaktifkan sumber daya mesin

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda, lalu pilih Pengaturan proyek. Pilih Sumber daya mesin.
3. Pilih salah satu dari berikut ini.

   - Untuk mengaktifkan satu per satu, pilih pemilih di sebelah satu atau beberapa sumber daya mesin yang ingin Anda aktifkan. Pilih Aktifkan, lalu pilih Sumber daya ini.
   - Untuk mengaktifkan semua sumber daya, pilih Aktifkan, lalu pilih Semua sumber daya.
   - Untuk mengaktifkan semua tindakan alur kerja, pilih Aktifkan, lalu pilih Semua tindakan alur kerja.
   - Untuk mengaktifkan semua cetak biru, pilih Aktifkan, lalu pilih Semua cetak biru.

Kuota untuk proyek

Tabel berikut menjelaskan kuota dan batasan untuk proyek di Amazon CodeCatalyst. Untuk informasi selengkapnya tentang kuota di Amazon CodeCatalyst, lihatKuota untuk CodeCatalyst.

Jumlah maksimum proyek per ruang	100

Jumlah maksimum proyek yang dapat dimiliki pengguna	1.000
Jumlah maksimum anggota yang dapat menjadi bagian dari suatu proyek.	10.000
Nama proyek	Nama proyek harus unik dalam ruang. Nama harus antara 3 dan 63 karakter. Nilai peka huruf besar/kecil. Nama proyek harus dimulai dengan karakter alfanumerik. Karakter yang valid: A-Z, a-z, 0-9, spasi, dan., _ (garis bawah) - (tanda hubung) Nama proyek tidak dapat berisi salah satu karakter berikut: `! ? @ # $ % ^ & * ( ) + = { } [ ] | \ / > < ~ ` ' " ; :`
Deskripsi proyek	Deskripsi proyek bisa sampai 200 karakter. Karakter yang valid: A-Z, a-z, 0-9, spasi, dan., _ (garis bawah) - (tanda hubung). Deskripsi proyek bersifat opsional.

Bekerja dengan notifikasi di CodeCatalyst

Anda dapat mengatur notifikasi untuk memantau proyek dan sumber daya Anda CodeCatalyst. Pengguna dapat memilih acara proyek yang ingin mereka terima email di proyek mana pun di mana mereka menjadi anggota. Anda juga dapat memilih untuk mengonfigurasi notifikasi yang dikirim ke seluruh tim dalam aplikasi perpesanan tim, seperti Slack, dengan mengonfigurasi akses antara CodeCatalyst ruang dan ruang kerja Slack, dan kemudian mengonfigurasi pemberitahuan untuk proyek yang akan dikirim ke satu atau beberapa saluran di ruang kerja Slack itu. Setelah Anda mengonfigurasi akses antara CodeCatalyst ruang dan ruang kerja Slack, anggota proyek juga akan memiliki opsi untuk menambahkan ID anggota Slack mereka sendiri sehingga mereka dapat diberitahu secara langsung tentang CodeCatalyst peristiwa di ruang kerja dan saluran Slack yang terhubung.

> **Note**
>
> Kumpulan peristiwa proyek yang dapat dikirim ke Slack bukanlah kumpulan peristiwa yang sama yang dapat dipilih pengguna untuk diberitahukan melalui email.

Topik

Bagaimana notifikasi bekerja?

Anda dapat mengatur proyek Anda untuk memberikan pemberitahuan ke aplikasi pesan tim Anda, seperti Slack.

Izin apa yang diperlukan untuk notifikasi?

Setiap anggota proyek dapat mengonfigurasi, melihat, memperbarui, atau menghapus setelan notifikasi untuk saluran CodeCatalyst. Namun, hanya pengguna dengan peran administrator Space yang dapat menambah atau menghapus ruang kerja Slack. Semua pengguna dapat mengonfigurasi acara proyek apa yang ingin mereka terima email untuk proyek tempat mereka berada CodeCatalyst.

CodeCatalyst Acara apa yang dapat saya konfigurasikan notifikasi?

Anda dapat mengonfigurasi CodeCatalyst untuk mengirimkan pemberitahuan ke satu atau beberapa saluran Slack tentang peristiwa alur kerja. Setelah pemberitahuan telah dikonfigurasi antara CodeCatalyst proyek dan Slack, pengguna proyek dapat memilih untuk menambahkan ID anggota Slack mereka sendiri untuk menerima pesan langsung di saluran Slack tentang peristiwa. CodeCatalyst Pengguna yang menambahkan ID anggota Slack mereka akan menerima sebutan langsung ke ID mereka di saluran Slack yang dikonfigurasi untuk proyek mereka, membantu meningkatkan kesadaran tentang peristiwa yang mereka pedulikan.

Anda juga dapat memilih acara apa yang ingin Anda terima email. Email ini dikirim ke alamat email yang dikonfigurasi untuk AWS Builder ID Anda.

Bagaimana notifikasi muncul?

Anda dapat mengonfigurasi CodeCatalyst untuk mengirimkan pemberitahuan ke satu atau beberapa saluran Slack. Anda perlu mengotorisasi CodeCatalyst untuk memberikan izin untuk mengakses ruang kerja Slack Anda. Setelah otorisasi diberikan, CodeCatalyst dapat mengirimkan pemberitahuan ke saluran Slack yang Anda konfigurasi. Jika anggota proyek memilih untuk menambahkan ID anggota Slack mereka, mereka dapat menerima sebutan tentang CodeCatalyst peristiwa di saluran Slack yang dikonfigurasi untuk proyek tersebut.

Bagaimana cara mengatur notifikasi?

Pemberitahuan email dikonfigurasi sebagai bagian dari CodeCatalyst. Pengguna proyek dapat memilih acara apa yang ingin mereka terima email di halaman Pengaturan Saya.

Untuk mengatur pemberitahuan Slack untuk sumber daya proyek Anda, Anda harus menyelesaikan tugas tingkat tinggi berikut.

Untuk mengatur notifikasi (tugas tingkat tinggi)

1. Di CodeCatalyst, Anda mengatur koneksi antara CodeCatalyst dan klien perpesanan, seperti Slack. Setelah ruang kerja Slack terhubung, itu akan tersedia untuk semua proyek di ruang tersebut.

   > **Note**
   >
   > Hanya pengguna dengan peran administrator Space yang dapat menambah atau menghapus ruang kerja Slack.

2. Dalam proyek Anda CodeCatalyst, tambahkan saluran tempat Anda ingin tim Anda menerima pemberitahuan.
3. Di CodeCatalyst, Anda mengaktifkan notifikasi untuk berbagai acara, seperti kegagalan menjalankan alur kerja, dan menentukan saluran tempat Anda ingin mengirimnya.

Untuk langkah terperinci, lihat Memulai dengan notifikasi Slack.

Setelah pemberitahuan dikonfigurasi antara CodeCatalyst spasi dan Slack, pengguna dapat memilih untuk menambahkan ID anggota Slack mereka sendiri untuk menerima pesan langsung tentang CodeCatalyst peristiwa di saluran Slack yang dikonfigurasi untuk proyek mereka,

Memulai dengan notifikasi Slack

Setelah membuat proyek, Anda dapat mengatur pemberitahuan Slack yang membantu tim Anda memantau sumber daya proyek.

Langkah-langkah ini memandu Anda melalui pengaturan pemberitahuan Slack untuk pertama kalinya. CodeCatalyst Jika Anda telah mengonfigurasi notifikasi, lihatMengelola notifikasi di Amazon CodeCatalyst.

> **Note**
>
> Kumpulan peristiwa proyek yang dapat dikirim ke saluran notifikasi bukanlah rangkaian peristiwa yang sama yang dapat dipilih pengguna untuk diberitahukan melalui email. Untuk informasi selengkapnya, lihat Mengelola notifikasi di Amazon CodeCatalyst.

Topik

Prasyarat

Sebelum memulai, Anda perlu melakukan hal berikut:

- Sebuah CodeCatalyst ruang. Untuk informasi tentang membuat CodeCatalyst spasi dan masuk untuk pertama kalinya, lihatSiapkan dan masuk ke CodeCatalyst.
- Sebuah CodeCatalyst proyek. Untuk informasi selengkapnya, lihat Membuat proyek.
- CodeCatalyst Akun dengan administrator Proyek atau peran administrator Space. Untuk informasi selengkapnya, lihat Memberikan akses dengan peran pengguna.
- Akun Slack dan ruang kerja Slack yang dapat diakses oleh. CodeCatalyst
- Saluran Slack tempat CodeCatalyst akan mengirim notifikasi. Saluran dapat bersifat publik atau pribadi.

Langkah 1: Connect CodeCatalyst ke ruang kerja Slack Anda

Hanya pengguna dengan peran administrator Space yang dapat menambah atau menghapus ruang kerja Slack. Menambahkan atau menghapus ruang kerja Slack memengaruhi semua proyek di ruang tersebut. Untuk membuat koneksi antara CodeCatalyst dan Slack, CodeCatalyst lakukan jabat tangan otentikasi OAuth yang aman dengan ruang kerja Slack Anda.

Gunakan petunjuk berikut untuk terhubung CodeCatalyst ke ruang kerja Slack Anda.

> **Note**
>
> Ini hanya perlu dilakukan sekali untuk setiap ruang kerja Slack. Anda kemudian dapat mengatur notifikasi oleh saluran Slack.

Untuk terhubung CodeCatalyst ke ruang kerja Slack

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda.
3. Di panel navigasi, pilih Pengaturan proyek.
4. Pilih tab Pemberitahuan.
5. Pilih Konfigurasikan notifikasi.
6. Pilih Connect to Slack workspace.
7. Baca isi kotak dialog, lalu pilih Connect to Slack workspace.
8. Pada pesan AWSChatbot:

   a. Di kanan atas, pilih ruang kerja Slack yang berisi saluran Anda.

   b. Pilih Izinkan.

   Anda dikembalikan ke CodeCatalyst konsol.
9. Lanjutkan ke Langkah 2: Tambahkan saluran Slack Anda CodeCatalyst.

Langkah 2: Tambahkan saluran Slack Anda CodeCatalyst

Anda memerlukan ID saluran Slack untuk menambahkan saluran Anda. CodeCatalyst

Untuk mendapatkan ID saluran Slack Anda

1. Masuk ke Slack. Untuk informasi selengkapnya, lihat Masuk ke Slack.
2. Buka ruang kerja Slack yang berisi saluran tempat Anda ingin pemberitahuan pergi. Untuk informasi selengkapnya, lihat Beralih di antara ruang kerja Slack atau Masuk ke ruang kerja Slack tambahan.
3. Di panel navigasi, buka menu konteks (klik kanan) untuk saluran tempat Anda ingin pemberitahuan pergi, dan pilih Buka detail saluran.

   ID saluran ditampilkan di bagian bawah kotak dialog.
4. Salin nilai ID Saluran. Anda akan membutuhkannya di langkah berikutnya.

Dengan menggunakan ID saluran yang baru saja Anda salin, Anda sekarang dapat menghubungkan saluran Slack Anda. CodeCatalyst

Untuk menambahkan saluran Slack Anda CodeCatalyst

1. Sebelum memulai, jika saluran Slack Anda bersifat pribadi, tambahkan aplikasi AWS Chatbot ke saluran sebagai berikut:

   a. Di kotak pesan saluran Slack Anda, masukkan **`@aws`** dan pilih aws app dari kotak dialog.

   b. Tekan Enter.

      Pesan Slackbot muncul, menunjukkan bahwa Chatbot AWS tidak ada di saluran pribadi.

   c. Pilih Undang Mereka untuk mengundang AWS Chatbot ke saluran.
2. Di CodeCatalyst konsol, pilih Berikutnya.
3. Di Channel ID, rekatkan ID saluran Slack yang Anda peroleh sebelumnya.
4. Di Nama saluran, masukkan nama. Sebaiknya gunakan nama saluran Slack.
5. Pilih Selanjutnya.
6. Di Pilih acara pemberitahuan, pilih jenis acara yang ingin Anda terima notifikasi.
7. Pilih Selesai.

Langkah 3: Uji pemberitahuan dari CodeCatalyst ke Slack

Setelah proyek Anda dikonfigurasi untuk mengirim pemberitahuan untuk status alur kerja, Anda dapat melihat notifikasi Anda di Slack.

Untuk melihat notifikasi Anda di Slack

1. Dalam CodeCatalyst proyek Anda, mulai alur kerja secara manual untuk menyelesaikan alur kerja dan menerima pemberitahuan status saat proses selesai.
2. Di Slack, lihat saluran yang Anda atur untuk notifikasi. Pemberitahuan Anda menampilkan status terbaru dari setiap alur kerja yang dijalankan, dan apakah itu gagal atau berhasil.

Langkah 4: Langkah selanjutnya

Setelah ruang kerja Slack dikonfigurasi untuk CodeCatalyst ruang Anda, Anda dapat menambahkan saluran Slack tambahan CodeCatalyst proyek yang ada, dan menambahkannya untuk proyek baru setelah Anda membuatnya. Anda juga dapat memberi tahu pengguna proyek bahwa mereka dapat mengonfigurasi pemberitahuan Slack pribadi untuk ID anggota Slack mereka, dan mengonfigurasi acara yang akan mereka terima emailnya. Untuk informasi selengkapnya, lihat Mengelola notifikasi di Amazon CodeCatalyst.

Mengelola notifikasi di Amazon CodeCatalyst

Anda dapat mengonfigurasi CodeCatalyst untuk mengirim pemberitahuan tentang peristiwa di proyek Anda. Anda dapat mengirim notifikasi ke klien perpesanan seperti saluran Slack. Pengguna proyek dapat memilih acara proyek apa yang akan mereka beri tahu melalui email yang dikirim ke alamat email yang dikonfigurasi untuk profil mereka.

> **Note**
>
> Kumpulan peristiwa proyek yang dapat dikirim ke saluran notifikasi bukanlah rangkaian peristiwa yang sama yang dapat dipilih pengguna untuk diberitahukan melalui email.

Topik

- Mengelola notifikasi yang dikirimkan langsung kepada Anda
- Mengelola notifikasi yang dikirim ke saluran

Mengelola notifikasi yang dikirimkan langsung kepada Anda

Anda dapat memilih untuk memiliki pemberitahuan email yang dikirimkan kepada Anda tentang peristiwa di proyek mana pun di mana Anda menjadi anggota. Email ini akan dikirim ke alamat email yang dikonfigurasi di AWS Builder ID Anda. Secara default, Anda akan menerima email tentang semua acara proyek yang emailnya dapat dikirim.

Jika proyek telah dikonfigurasi untuk mengirim pemberitahuan ke saluran Slack, Anda dapat menambahkan ID anggota Slack Anda untuk menerima sebutan langsung tentang CodeCatalyst peristiwa di saluran Slack tersebut. Ini dapat membantu meningkatkan kesadaran Anda tentang peristiwa yang terjadi dalam proyek di mana Anda memiliki peran.

Untuk mengonfigurasi pemberitahuan email untuk acara proyek

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di bilah menu atas, pilih lencana profil Anda, lalu pilih Pengaturan saya. Halaman Pengaturan CodeCatalyst saya terbuka.

   > **Tip**
   >
   > Anda juga dapat menemukan profil pengguna Anda dengan membuka halaman anggota untuk proyek atau ruang dan memilih nama Anda dari daftar anggota.

3. Di Pemberitahuan email, temukan proyek dalam daftar tempat Anda ingin mengonfigurasi notifikasi email, lalu pilih Edit.
4. Pilih acara yang ingin Anda terima emailnya, lalu pilih Simpan.

Untuk mengkonfigurasi pemberitahuan Slack pribadi

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di bilah menu atas, pilih lencana profil Anda, lalu pilih Pengaturan saya. Halaman Pengaturan CodeCatalyst saya terbuka.

   > **Tip**
   >
   > Anda juga dapat menemukan profil pengguna Anda dengan membuka halaman anggota untuk proyek atau ruang dan memilih nama Anda dari daftar anggota.

3. Di notifikasi Personal Slack, pilih Connect Slack ID, lalu pilih Connect to Slack workspace. Jendela terpisah akan terbuka.

> **Tip**
>
> Opsi ini tidak dapat dikonfigurasi kecuali pengguna dengan peran administrator Space telah menambahkan ruang kerja Slack untuk ruang Anda. CodeCatalyst Untuk informasi selengkapnya, lihat Memulai dengan notifikasi Slack dan Mengelola notifikasi yang dikirim ke saluran.

4. Di jendela permintaan izin, pastikan bahwa nama ruang kerja cocok dengan ruang kerja Slack yang dikonfigurasi untuk ruang tersebut. CodeCatalyst Pilih Izinkan untuk mengizinkan AWS Chatbot akses ke ruang kerja. Jendela akan ditutup, dan ruang kerja Slack akan menampilkan status Connnection sebagai Connected.

> **Tip**
>
> Jika status koneksi tidak berubah, periksa untuk melihat apakah terjadi kesalahan saat menghubungkan ruang kerja Slack. Anda mungkin harus menggulir ke atas untuk melihat kesalahannya.

5. Untuk berhenti menerima notifikasi Slack pribadi, pilih ruang kerja Slack yang terhubung, lalu pilih Putuskan Slack ID.

Mengelola notifikasi yang dikirim ke saluran

Anda dapat memilih untuk menambahkan dan mengelola pemberitahuan tentang peristiwa proyek yang CodeCatalyst dikirim ke sumber daya tim, seperti saluran Slack tim. Dengan melakukan ini, Anda dapat membantu memastikan bahwa seluruh tim Anda mengetahui peristiwa penting, seperti ketika alur kerja gagal dijalankan.

> **Note**
>
> Setiap anggota proyek dapat mengelola notifikasi yang dikirim ke saluran untuk proyek tersebut. Namun, hanya pengguna dengan peran administrator Space yang dapat menambah atau menghapus ruang kerja Slack.

Menambahkan saluran notifikasi untuk proyek

Anda dapat menambahkan saluran tempat Anda ingin menerima notifikasi, seperti saluran Slack tim Anda.

Untuk menambahkan saluran Slack untuk notifikasi

1. Jika Anda menambahkan saluran Slack pertama Anda, lihat sebagai gantinyaMemulai dengan notifikasi Slack.

   Setelah menyiapkan saluran pertama Anda, kembali ke prosedur ini untuk mengatur saluran tambahan.
2. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
3. Arahkan ke proyek Anda.
4. Di panel navigasi, pilih Pengaturan proyek.
5. Pilih tab Pemberitahuan.
6. Pilih Tambah saluran.
7. Pilih Pilih ruang kerja, lalu pilih ruang kerja Slack yang berisi saluran tempat Anda ingin mengirim notifikasi.

   Jika ruang kerja Slack Anda tidak ada dalam daftar, Anda dapat menambahkannya dengan mengikuti petunjuk di. Memulai dengan notifikasi Slack
8. Sebelum memasukkan ID Saluran, jika saluran Slack yang ingin Anda tambahkan bersifat pribadi, selesaikan langkah-langkah berikut:

   a. Di kotak pesan saluran Slack Anda, masukkan **`@aws`** dan pilih aplikasi aws dari pop-up.
   b. Tekan Enter.

      Pesan Slackbot muncul, menunjukkan bahwa Chatbot AWS tidak ada di saluran pribadi.
   c. Pilih Undang Mereka untuk mengundang AWS Chatbot ke saluran.
9. Di CodeCatalyst bidang ID Saluran, masukkan ID saluran Slack. Untuk menemukan ID, buka Slack, dan di panel navigasi, klik kanan saluran dan pilih Buka detail saluran.

   ID saluran ditampilkan di bagian bawah kotak dialog.
10. Di Nama saluran, masukkan nama. Sebaiknya gunakan nama saluran Slack.
11. Di Pilih acara pemberitahuan, pilih jenis acara yang ingin Anda terima notifikasi.
12. Pilih Tambahkan.

Mengedit notifikasi untuk saluran notifikasi

Anda dapat mengubah notifikasi saluran mana yang masuk, dan Anda dapat mematikan notifikasi tertentu sama sekali.

Untuk mengedit notifikasi

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda.
3. Di panel navigasi, pilih Pengaturan proyek.
4. Pilih tab Pemberitahuan.
5. Pilih Edit notifikasi.
6. Lakukan salah satu dari berikut:
   - Untuk mengirim pemberitahuan ke saluran tertentu, pilih saluran dari daftar drop-down.
   - Untuk mematikan notifikasi secara global, pilih sakelar di sebelah notifikasi.
   - Untuk berhenti mengirim pemberitahuan ke saluran tertentu, pilih X pada saluran.
7. Pilih Save (Simpan).

Menghapus saluran

Untuk menghapus saluran

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda. Di panel navigasi, pilih Pengaturan proyek.
3. Pada halaman Pengaturan proyek, pilih tab Pemberitahuan.
4. Pilih indikator di sebelah saluran yang ingin Anda hapus lalu pilih Hapus saluran. Saat diminta, pilih Ok di jendela konfirmasi.

Siapkan CodeCatalyst proyek dengan cetak biru

Blueprints adalah generator kode arbitrer yang mewakili komponen arsitektur proyek. CodeCatalyst Komponen dapat terdiri dari apa saja dari alur kerja dalam satu file ke seluruh proyek lengkap dengan kode sampel. Cetak biru mengambil serangkaian opsi yang sewenang-wenang dan menggunakannya untuk menghasilkan sekumpulan kode output arbitrer yang diteruskan ke proyek. Saat cetak biru diperbarui dengan praktik terbaik terbaru atau opsi baru, cetak biru dapat meregenerasi bagian yang relevan dari basis kode Anda dalam proyek yang berisi cetak biru itu.

Anda dapat menggunakan CodeCatalyst cetak biru Amazon untuk membuat proyek lengkap dengan repositori sumber, contoh kode sumber, alur kerja CI/CD, laporan pembuatan dan pengujian, serta alat pelacakan masalah terintegrasi. CodeCatalyst Cetak biru menghasilkan sumber daya dan kode sumber berdasarkan parameter konfigurasi yang ditetapkan. Saat menggunakan cetak biru yang CodeCatalyst dikelola, cetak biru yang Anda pilih menentukan sumber daya yang ditambahkan ke proyek Anda, serta alat yang CodeCatalyst membuat atau mengonfigurasi, sehingga Anda dapat melacak dan menggunakan sumber daya proyek Anda. Sebagai pengguna cetak biru, Anda dapat membuat proyek dengan cetak biru atau menerapkannya ke proyek yang ada. CodeCatalyst Anda dapat menerapkan beberapa cetak biru dalam proyek Anda, dan masing-masing dapat diterapkan sebagai komponen independen. Misalnya, Anda dapat memiliki proyek yang dibuat dengan cetak biru aplikasi web, dan kemudian Anda menerapkan cetak biru keamanan di lain waktu. Ketika salah satu cetak biru diperbarui, Anda dapat menggabungkan perubahan atau perbaikan dalam proyek Anda melalui manajemen siklus hidup. Untuk informasi selengkapnya, lihat Membuat proyek yang komprehensif dengan CodeCatalyst cetak biru dan Bekerja dengan manajemen siklus hidup sebagai pengguna cetak biru.

Sebagai penulis cetak biru, Anda juga dapat membuat dan menerbitkan cetak biru khusus untuk anggota CodeCatalyst ruang Anda untuk menggunakan sumber daya proyek Anda. Cetak biru khusus dapat dikembangkan untuk memenuhi kebutuhan tertentu untuk proyek ruang Anda. Setelah menambahkan cetak biru khusus ke katalog cetak biru ruang Anda, Anda dapat mengelola cetak biru dan terus melakukan pembaruan sehingga proyek ruang Anda tetap up to date dengan praktik terbaik terbaru. Untuk informasi selengkapnya, lihat Standarisasi proyek cetak biru kustom di CodeCatalyst. Untuk melihat cetak biru SDK dan contoh cetak biru, lihat repositori sumber terbuka. GitHub

Topik

- Memutuskan cetak biru dari proyek untuk menghentikan pembaruan
- Mengubah versi cetak biru dalam sebuah proyek
- Mengedit desciption untuk cetak biru dalam sebuah proyek
- Bekerja dengan manajemen siklus hidup sebagai pengguna cetak biru
- Membuat proyek yang komprehensif dengan CodeCatalyst cetak biru
- Standarisasi proyek cetak biru kustom di CodeCatalyst
- Kuota untuk cetak biru di CodeCatalyst

Membuat proyek dengan cetak biru

Anda dapat dengan cepat membuat proyek menggunakan cetak biru dari katalog cetak CodeCatalyst biru Amazon atau katalog ruang tim Anda dengan cetak biru khusus. Tergantung pada cetak biru, proyek Anda dibuat dengan sumber daya tertentu. Untuk informasi selengkapnya, lihat Membuat proyek dengan cetak biru dan Membuat proyek yang komprehensif dengan CodeCatalyst cetak biru.

Setelah membuat proyek, Anda dapat menerapkan cetak biru tambahan ke CodeCatalyst proyek Anda dari CodeCatalyst katalog atau katalog ruang Anda dengan cetak biru khusus. Cetak biru mewakili komponen arsitektur, sehingga beberapa cetak biru dapat digunakan bersama dalam proyek Anda untuk menggabungkan praktik terbaik tim Anda. Ini juga memberi Anda kemampuan untuk memastikan proyek Anda mutakhir dengan perubahan terbaru pada komponen yang berkembang. Untuk mempelajari selengkapnya tentang bekerja dengan cetak biru dalam proyek Anda, lihat. Bekerja dengan manajemen siklus hidup sebagai pengguna cetak biru

Menerapkan cetak biru dalam proyek untuk menambahkan sumber daya

Anda dapat menerapkan beberapa cetak biru dalam sebuah proyek untuk menggabungkan komponen fungsional, sumber daya, dan tata kelola. Proyek Anda dapat mendukung berbagai elemen yang dikelola secara independen dalam cetak biru terpisah. Menerapkan cetak biru ke proyek mengurangi kebutuhan untuk secara manual membuat sumber daya dan membuat komponen perangkat lunak berfungsi. Proyek Anda juga dapat tetap terkini seiring dengan berkembangnya persyaratan. Untuk mempelajari selengkapnya tentang menerapkan cetak biru dalam proyek Anda, lihat. Bekerja dengan manajemen siklus hidup sebagai pengguna cetak biru

Saat mengonfigurasi detail cetak biru, Anda juga dapat memilih untuk menyimpan kode sumber cetak biru di repositori pihak ketiga pilihan, di mana Anda masih dapat mengelola cetak biru dan

memanfaatkan kemampuan manajemen siklus hidup untuk menjaga proyek Anda tetap up to date. Untuk informasi selengkapnya, lihat Tambahkan fungsionalitas ke proyek dengan ekstensi di CodeCatalyst dan Bekerja dengan manajemen siklus hidup sebagai pengguna cetak biru.

Untuk menerapkan cetak biru untuk proyek Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, navigasikan ke ruang, lalu pilih proyek tempat Anda ingin menerapkan cetak biru.
3. Di panel navigasi, pilih Blueprints, lalu pilih Apply blueprint.
4. Pilih cetak biru dari tab cetak biru atau cetak biru khusus dari tab CodeCatalyst Blueprints Space, lalu pilih Berikutnya.
5. Di bawah Rincian cetak biru, pilih versi cetak biru dari menu tarik-turun versi Target. Versi katalog terbaru dipilih secara otomatis.
6. (Opsional) Secara default, kode sumber yang dibuat oleh cetak biru disimpan dalam repositori. CodeCatalyst Atau, Anda dapat memilih untuk menyimpan kode sumber cetak biru di repositori pihak ketiga. Untuk informasi selengkapnya, lihat Tambahkan fungsionalitas ke proyek dengan ekstensi di CodeCatalyst.

   Lakukan salah satu hal berikut tergantung pada penyedia repositori pihak ketiga yang ingin Anda gunakan:

   - GitHub repositori: Hubungkan akun. GitHub

     Pilih menu tarik-turun Advanced, pilih GitHub sebagai penyedia repositori, lalu pilih GitHub akun tempat Anda ingin menyimpan kode sumber yang dibuat oleh cetak biru.

     > **Note**
     >
     > Jika Anda menggunakan koneksi ke GitHub akun, Anda harus membuat koneksi pribadi untuk membuat pemetaan identitas antara CodeCatalyst identitas dan GitHub identitas Anda. Untuk informasi selengkapnya, lihat Koneksi pribadi dan Mengakses GitHub sumber daya dengan koneksi pribadi.

   - Bitbucket: Hubungkan ruang kerja Bitbucket.

     Pilih menu tarik-turun Advanced, pilih Bitbucket sebagai penyedia repositori, lalu pilih ruang kerja Bitbucket tempat Anda ingin menyimpan kode sumber yang dibuat oleh cetak biru.

7. Di bawah Konfigurasi cetak biru, konfigurasikan parameter cetak biru. Tergantung pada cetak biru, Anda mungkin memiliki opsi untuk memberi nama repositori sumber.
8. Tinjau perbedaan antara versi cetak biru saat ini dan versi terbaru Anda. Perbedaan yang ditampilkan dalam permintaan tarik menunjukkan perubahan antara versi saat ini dan versi terbaru, yang merupakan versi yang diinginkan pada saat permintaan tarik dibuat. Jika tidak ada perubahan yang ditampilkan, versi mungkin identik, atau Anda mungkin telah memilih versi yang sama untuk versi saat ini dan versi yang diinginkan.
9. Jika Anda puas bahwa permintaan tarik berisi kode dan perubahan yang ingin Anda tinjau, pilih Terapkan cetak biru. Setelah Anda membuat permintaan tarik, Anda dapat menambahkan komentar. Komentar dapat ditambahkan ke permintaan tarik atau ke baris individual dalam file serta permintaan tarik keseluruhan. Anda dapat menambahkan tautan ke sumber dayaseperti filesdengan menggunakan @ tanda, diikuti dengan nama file.

> **Note**
>
> Cetak biru tidak akan diterapkan sampai permintaan tarik disetujui dan digabungkan. Untuk informasi selengkapnya, lihat Meninjau permintaan tarik dan Menggabungkan permintaan tarik.

Penulis cetak biru juga dapat menerapkan cetak biru khusus untuk proyek di ruang tertentu yang tidak memiliki cetak biru yang tersedia untuk membuat proyek baru atau berlaku untuk proyek yang ada. Untuk informasi selengkapnya, lihat Menerbitkan dan menerapkan cetak biru khusus di ruang dan proyek tertentu.

Jika Anda tidak lagi ingin menerima pembaruan untuk cetak biru, Anda dapat memisahkan cetak biru dari proyek Anda. Untuk informasi selengkapnya, lihat Memutuskan cetak biru dari proyek untuk menghentikan pembaruan.

Memutuskan cetak biru dari proyek untuk menghentikan pembaruan

Jika Anda tidak ingin pembaruan baru dari cetak biru, Anda dapat memisahkan cetak biru dari proyek Anda. Sumber daya dan komponen perangkat lunak fungsional yang ditambahkan ke proyek Anda dari cetak biru akan tetap ada di proyek Anda.

Untuk memisahkan cetak biru dari proyek Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, navigasikan ke ruang, lalu pilih proyek tempat Anda ingin memisahkan cetak biru.
3. Di panel navigasi, pilih Cetak biru.
4. Pilih cetak biru dengan sumber daya yang ingin Anda pisahkan, pilih menu tarik-turun Tindakan, lalu pilih Putuskan cetak biru.
5. Masukkan `confirm` untuk mengkonfirmasi disasosiasi.
6. Pilih Konfirmasi.

Mengubah versi cetak biru dalam sebuah proyek

Jika Anda membuat proyek dengan cetak biru atau menerapkan cetak biru ke proyek yang sudah ada, maka Anda akan diberi tahu tentang versi baru cetak biru tersebut. Sebelum versi cetak biru diperbarui melalui permintaan tarik yang disetujui, Anda dapat melihat perubahan kode dan lingkungan yang terpengaruh. Manajemen siklus hidup memungkinkan Anda mengubah versi dari satu atau beberapa cetak biru yang diterapkan dalam proyek Anda, sehingga setiap versi cetak biru dapat diubah tanpa memengaruhi area lain dari proyek Anda. Anda juga dapat mengganti pembaruan cetak biru. Untuk informasi selengkapnya, lihat Bekerja dengan manajemen siklus hidup sebagai pengguna cetak biru.

Untuk memperbarui cetak biru ke versi terbaru

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, navigasikan ke ruang tempat Anda ingin memperbarui versi cetak biru.
3. Di dasbor ruang, pilih proyek dengan cetak biru yang ingin Anda perbarui.
4. Di panel navigasi, pilih Cetak biru.
5. Di kolom Status, pilih tautan untuk mengubah versi katalog (misalnya, Ubah katalog versi 0.3.109).
6. Pilih menu tarik-turun Tindakan, lalu pilih Perbarui versi. Versi terbaru dipilih secara otomatis.
7. (Opsional) Di bawah Konfigurasi cetak biru, konfigurasikan parameter cetak biru.
8. (Opsional) Di tab Perubahan kode, tinjau perbedaan antara versi cetak biru saat ini dan versi yang diperbarui. Perbedaan yang ditampilkan dalam permintaan tarik adalah perubahan antara versi saat ini dan versi terbaru, yang merupakan versi yang diinginkan pada saat permintaan

tarik dibuat. Jika tidak ada perubahan yang ditampilkan, versi mungkin identik, atau Anda mungkin telah memilih versi yang sama untuk versi saat ini dan versi yang diinginkan.

9. Jika Anda puas bahwa permintaan tarik berisi kode dan perubahan yang ingin Anda tinjau, pilih Terapkan pembaruan. Setelah Anda membuat permintaan tarik, Anda dapat menambahkan komentar. Komentar dapat ditambahkan ke permintaan tarik atau ke baris individual dalam file dan ke permintaan tarik keseluruhan. Anda dapat menambahkan tautan ke sumber daya seperti file dengan menggunakan @ tanda, diikuti dengan nama file.

> **Note**
>
> Cetak biru tidak akan diperbarui hingga permintaan tarik disetujui dan digabungkan. Untuk informasi selengkapnya, lihat Meninjau permintaan tarik dan Menggabungkan permintaan tarik.

> **Note**
>
> Jika Anda memiliki permintaan tarik yang terbuka untuk memperbarui versi cetak biru, tutup permintaan tarik sebelumnya sebelum membuat yang baru. Ketika Anda memilih versi Perbarui, Anda akan diarahkan ke daftar permintaan tarik tertunda untuk cetak biru. Anda juga dapat melihat permintaan tarik tertunda dari tab Blueprints di Pengaturan proyek dan halaman ringkasan proyek. Untuk informasi selengkapnya, lihat Melihat permintaan tarik.

Untuk mengubah versi cetak biru

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, navigasikan ke ruang tempat Anda ingin memperbarui versi cetak biru.
3. Di dasbor ruang, pilih proyek dengan cetak biru yang ingin Anda perbarui.
4. Di panel navigasi, pilih Blueprints, lalu pilih tombol radio untuk cetak biru yang ingin Anda perbarui.
5. Pilih menu dropdown Actions, lalu pilih Configure blueprint.
6. Dari menu tarik-turun versi Target, pilih versi yang ingin Anda gunakan. Versi terbaru dipilih secara otomatis.
7. (Opsional) Di bawah Konfigurasi cetak biru, konfigurasikan parameter cetak biru.

8. (Opsional) Di tab Perubahan kode, tinjau perbedaan antara versi cetak biru saat ini dan versi yang diperbarui. Perbedaan yang ditampilkan dalam permintaan tarik adalah perubahan antara versi saat ini dan versi terbaru, yang merupakan versi yang diinginkan pada saat permintaan tarik dibuat. Jika tidak ada perubahan yang ditampilkan, versi mungkin identik, atau Anda mungkin telah memilih versi yang sama untuk versi saat ini dan versi yang diinginkan.
9. Jika Anda puas bahwa permintaan tarik berisi kode dan perubahan yang ingin Anda tinjau, pilih Terapkan pembaruan. Setelah Anda membuat permintaan tarik, Anda dapat menambahkan komentar. Komentar dapat ditambahkan ke permintaan tarik atau ke baris individual dalam file dan ke permintaan tarik keseluruhan. Anda dapat menambahkan tautan ke sumber daya seperti file dengan menggunakan @ tanda, diikuti dengan nama file.

> **Note**
>
> Cetak biru tidak akan diperbarui hingga permintaan tarik disetujui dan digabungkan. Untuk informasi selengkapnya, lihat Meninjau permintaan tarik dan Menggabungkan permintaan tarik.

> **Note**
>
> Jika Anda memiliki permintaan tarik yang terbuka untuk memperbarui versi cetak biru, tutup permintaan tarik sebelumnya sebelum membuat yang baru. Ketika Anda memilih versi Perbarui, Anda akan diarahkan ke daftar permintaan tarik tertunda untuk cetak biru. Anda juga dapat melihat permintaan tarik tertunda dari tab Blueprints di Pengaturan proyek dan halaman ringkasan proyek. Untuk informasi selengkapnya, lihat Melihat permintaan tarik.

Mengedit desciption untuk cetak biru dalam sebuah proyek

Anda dapat mengedit deskripsi cetak biru yang Anda gunakan untuk membuat proyek atau diterapkan setelah proyek dibuat. Cetak biru dapat digunakan lebih dari sekali dalam sebuah proyek. Untuk membedakan tujuan cetak biru dalam proyek Anda, Anda dapat menggunakan deskripsi untuk cetak biru tersebut. Deskripsi juga dapat digunakan untuk mengidentifikasi komponen yang Anda terapkan dari cetak biru tertentu.

Untuk mengedit deskripsi cetak biru dalam proyek Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, navigasikan ke ruang Anda, lalu pilih proyek dengan pengaturan cetak biru yang ingin Anda perbarui.
3. Di panel navigasi, pilih Cetak biru.
4. Pilih cetak biru dengan deskripsi yang ingin Anda perbarui, pilih menu tarik-turun Tindakan, lalu pilih Edit deskripsi.
5. Di kolom input teks deskripsi Blueprint, masukkan deskripsi untuk mengidentifikasi cetak biru dalam proyek Anda.
6. Pilih Simpan.

Bekerja dengan manajemen siklus hidup sebagai pengguna cetak biru

Manajemen siklus hidup adalah kemampuan untuk membuat ulang basis kode dari opsi atau versi cetak biru yang diperbarui. Hal ini memungkinkan penulis cetak biru untuk secara terpusat mengelola siklus hidup pengembangan perangkat lunak dari setiap proyek yang berisi cetak biru tertentu. Misalnya, mendorong perbaikan keamanan ke cetak biru aplikasi web akan memungkinkan setiap proyek yang berisi atau dibuat dari cetak biru aplikasi web untuk mengambil perbaikan itu secara otomatis. Kerangka kerja manajemen yang sama ini juga memungkinkan Anda sebagai pengguna cetak biru untuk mengubah opsi cetak biru setelah dipilih.

Topik

Menggunakan manajemen siklus hidup pada proyek yang ada

Anda dapat menggunakan manajemen siklus hidup untuk proyek yang dibuat dari cetak biru atau pada proyek yang ada yang tidak terkait dengan cetak biru apa pun. Misalnya, Anda dapat menambahkan cetak biru praktik keamanan standar ke dalam aplikasi five-year-old Java yang tidak pernah dibuat dari cetak biru. Cetak biru menghasilkan alur kerja pemindaian keamanan dan

kode terkait lainnya. Bagian basis kode dalam aplikasi Java sekarang akan terus diperbarui secara otomatis dengan praktik terbaik tim Anda setiap kali perubahan dilakukan pada cetak biru.

Menggunakan manajemen siklus hidup pada beberapa cetak biru dalam sebuah proyek

Karena cetak biru mewakili komponen arsitektur, beberapa cetak biru sering dapat digunakan bersama dalam proyek yang sama. Misalnya, sebuah proyek dapat terdiri dari cetak biru API web pusat yang dibangun oleh insinyur platform perusahaan, bersama dengan cetak biru pemeriksaan rilis yang dibangun oleh tim keamanan aplikasi. Masing-masing cetak biru tersebut dapat diperbarui secara independen dan akan mengingat resolusi gabungan yang diterapkan padanya di masa lalu.

> **Note**
>
> Sebagai komponen arsitektur yang sewenang-wenang, tidak semua cetak biru masuk akal bersama atau secara logis akan bekerja sama, meskipun mereka masih akan mencoba untuk bergabung satu sama lain.

Bekerja dengan konflik dalam permintaan tarik siklus hidup

Terkadang, permintaan tarik siklus hidup dapat menghasilkan konflik gabungan. Ini dapat diselesaikan secara manual. Resolusi diingat pada pembaruan cetak biru berikutnya.

Memilih keluar dari perubahan manajemen siklus hidup

Pengguna dapat menghapus cetak biru dari proyek untuk memisahkan semua referensi ke cetak biru dan memilih keluar dari pembaruan siklus hidup. Untuk alasan keamanan, ini tidak menghapus atau memengaruhi kode atau sumber daya proyek apa pun, termasuk apa yang ditambahkan dari cetak biru. Untuk informasi selengkapnya, lihat Memutuskan cetak biru dari proyek untuk menghentikan pembaruan.

Mengesampingkan manajemen siklus hidup cetak biru dalam sebuah proyek

Jika Anda ingin mengganti pembaruan cetak biru ke file tertentu dalam proyek Anda, Anda dapat menyertakan file kepemilikan di repositori Anda. GitLabSpesifikasi Pemilik Kode adalah pedoman

yang direkomendasikan. Cetak biru selalu menghormati file pemilik kode di atas yang lainnya dan dapat menghasilkan contoh seperti berikut:

```
new BlueprintOwnershipFile(sourceRepo, {
      resynthesis: {
        strategies: [
          {
            identifier: 'dont-override-sample-code',
            description: 'This strategy is applied accross all sample code. The
 blueprint will create sample code, but skip attempting to update it.',
            strategy: MergeStrategies.neverUpdate,
            globs: [
              '**/src/**',
              '**/css/**',
            ],
          },
        ],
      },
    });
```

Ini menghasilkan a `.ownership-file` dengan konten berikut:

```
[dont-override-sample-code] @amazon-codecatalyst/blueprints.import-from-git
# This strategy is applied accross all sample code. The blueprint will create sample
 code, but skip attempting to update it.
# Internal merge strategy: neverUpdate
**/src/**
**/css/**
```

Membuat proyek yang komprehensif dengan CodeCatalyst cetak biru

Saat Anda membuat proyek menggunakan cetak biru, buat proyek lengkap dengan repositori sumber, CodeCatalyst contoh kode sumber, alur kerja CI/CD, laporan pembuatan dan pengujian, dan alat pelacakan masalah terintegrasi. Cetak biru proyek menggunakan kode untuk menyediakan infrastruktur cloud, sumber daya, dan artefak sumber sampel untuk berbagai jenis aplikasi dan kerangka kerja.

Untuk informasi selengkapnya, lihat Membuat proyek. Anda harus menjadi administrator Space untuk membuat proyek.

Topik

Cetak biru yang tersedia

Nama cetak biru	Deskripsi cetak biru
API web ASP.NET Core	Cetak biru ini membuat aplikasi API web .NET 6 ASP.NET Core. Cetak biru menggunak an alat AWS Deployment untuk.NET dan menyediakan opsi untuk mengonfigurasi Amazon Elastic Container Service, AWS App Runner, atau AWS Elastic Beanstalk sebagai target penyebaran.
AWS Glue ETL	Cetak biru ini membuat implementasi referensi sample extract transform load (ETL) menggunakan CDK, AWS Glue AWS , Lambda AWS , dan Amazon Athena untuk mengonver si nilai yang dipisahkan koma (CSV) ke Apache Parquet.
DevOps pipa penyebaran	Cetak biru ini membuat pipeline penerapan menggunakan Arsitektur Referensi Pipeline AWS Deployment yang menyebarkan aplikasi referensi ke berbagai tahap. AWS
Java API dengan AWS Fargate	Cetak biru ini membuat proyek layanan web kontainer. Proyek ini menggunakan AWS Copilot CLI untuk membangun dan menerapka n layanan web Spring Boot Java dalam wadah yang didukung oleh Amazon DynamoDB di Amazon ECS. Proyek ini menerapkan aplikasi kontainer ke cluster Amazon ECS pada komputasi tanpa server. AWS Fargate Aplikasi

Nama cetak biru	Deskripsi cetak biru
	menyimpan data dalam tabel DynamoDB. Setelah alur kerja Anda berjalan dengan sukses, layanan web sampel tersedia untuk umum melalui Application Load Balancer.
Aplikasi web tiga tingkat modern	Cetak biru ini menghasilkan kode dalam Python untuk lapisan aplikasi dan kerangka kerja front-end Vue untuk membangun dan menyebark an aplikasi web modern 3-tier yang dirancang dengan baik.
.NET aplikasi tanpa server	Cetak biru ini membuat AWS Lambda fungsi menggunakan alat-alat.NET CLI Lambda. Cetak biru menyediakan opsi untuk AWS Lambda fungsi, termasuk pilihan C # atau F #.
Node.js API dengan AWS Fargate	Cetak biru ini membuat proyek layanan web kontainer. Proyek ini menggunakan AWS Copilot CLI untuk membangun dan menerapkan layanan web Express/Node.js dalam kontainer di Amazon Elastic Container Service. Proyek ini menerapkan aplikasi kontainer ke cluster Amazon ECS pada komputasi tanpa server. AWS Fargate Setelah alur kerja Anda berjalan dengan sukses, layanan web sampel tersedia untuk umum melalui Application Load Balancer.
Model Aplikasi Tanpa Server (SAM)	Cetak biru ini membuat proyek yang menggunakan model aplikasi tanpa server (SAM) untuk membuat dan menyebarkan API. Anda dapat memilih SDK for Java TypeScrip t,, atau SDK untuk Python sebagai bahasa pemrograman.

Nama cetak biru	Deskripsi cetak biru
Penangan gambar tanpa server	Cetak biru ini membuat aplikasi untuk pemrosesan gambar berkecepatan tinggi tanpa mengurangi kualitas gambar.
Layanan mikro RESTful tanpa server	Cetak biru ini membuat REST API yang menggunakan AWS Lambda dan Amazon API Gateway dengan referensi layanan To Do. Anda dapat memilih SDK for Java TypeScrip t,, atau SDK untuk Python sebagai bahasa pemrograman.
Aplikasi satu halaman	Cetak biru ini membuat aplikasi satu halaman (SPA) yang menggunakan kerangka kerja React, Vue, dan Angular. Untuk hosting, pilih dari AWS Amplify Hosting atau Amazon CloudFront dan Amazon S3.
Situs web statis	Cetak biru ini membuat situs web statis menggunakan generator situs statis Hugo atau Jekyll. Generator situs statis menggunakan file input teks (seperti Markdown) untuk menghasil kan halaman web statis. Mereka ideal untuk konten informatif yang jarang berubah, seperti halaman produk, dokumentasi, dan blog. Cetak biru menggunakan AWS CDK untuk menyebark an halaman web statis ke salah satu atau Amazon AWS Amplify S3 +. CloudFront
Untuk Melakukan aplikasi web	Cetak biru ini menciptakan aplikasi web To Do tanpa server dengan komponen frontend dan backend. Anda dapat memilih SDK for Java TypeScript,, atau SDK untuk Python sebagai bahasa pemrograman.

Nama cetak biru	Deskripsi cetak biru
V layanan ideo-on-demand web	Cetak biru ini menciptakan video-on-demand layanan yang menyediakan kemampuan untuk menerima, mentranskode, dan mengirimkan konten. Cetak biru menggunakan, Amazon AWS Lambda S3,, dan. Amazon CloudWatch AWS Elemental MediaConvert
Berlangganan cetak biru eksternal	Cetak biru ini menciptakan alur kerja untuk setiap paket yang diimpor. Alur kerja ini berjalan sekali sehari untuk memeriksa NPM untuk versi baru dari paket. Jika ada versi baru, alur kerja mencoba menambahkannya ke CodeCatalyst ruang Anda sebagai cetak biru khusus. Tindakan akan gagal jika paket tidak dapat ditemukan atau bukan cetak biru. Paket target harus di NPM, dan paket harus cetak biru. Ruang harus berlangganan pada tingkat yang mendukung cetak biru khusus.
Batuan dasar GenAI chatbot	Cetak biru ini menciptakan chatbot AI generatif dengan Amazon Bedrock dan Claude Anthropic. Dengan cetak biru ini, Anda dapat membangun dan menyebarkan taman bermain LLM Anda sendiri yang aman dan dilindung i login yang dapat disesuaikan dengan data Anda. Untuk informasi selengkapnya, lihat dokumentasi Bedrock GenAI Chatbot.

Nama cetak biru	Deskripsi cetak biru
Cetak biru AWS Project Development Kit (AWS PDK)	Cetak biru PDK ini dapat disusun bersama untuk membuat aplikasi yang terdiri dari situs web React, Smithy API, dan infrastruktur CDK pendukung untuk menerapkannya ke AWS. AWS PDK menyediakan blok bangunan untuk pola umum bersama dengan alat pengemban gan untuk mengelola dan membangun proyek Anda. Untuk informasi selengkapnya, lihat repositori GitHub sumber AWS PDK dan. Tutorial: Membuat aplikasi full-stack dengan cetak biru PDK yang dapat dikomposisi

Menemukan informasi cetak biru proyek

Beberapa cetak biru proyek tersedia di. CodeCatalyst Untuk setiap cetak biru, ada ringkasan dan file README yang menyertainya. Ringkasan menjelaskan sumber daya yang diinstal oleh cetak biru, sedangkan file README menjelaskan cetak biru secara rinci dan memberikan instruksi tentang cara menggunakannya.

Standarisasi proyek cetak biru kustom di CodeCatalyst

Anda dapat membakukan pengembangan dan praktik terbaik untuk proyek CodeCatalyst ruang Anda dengan cetak biru khusus. Cetak biru khusus dapat digunakan untuk mendefinisikan berbagai aspek CodeCatalyst proyek, seperti definisi alur kerja dan kode aplikasi. Setelah cetak biru khusus digunakan untuk membuat proyek baru atau diterapkan pada proyek yang ada, setiap perubahan pada cetak biru tersedia untuk proyek tersebut sebagai pembaruan permintaan tarik. Sebagai penulis cetak biru, Anda dapat melihat detail tentang proyek mana yang menggunakan cetak biru Anda di seluruh ruang Anda, sehingga Anda dapat melihat bagaimana standar diterapkan di seluruh proyek. Pengelolaan siklus hidup cetak biru memungkinkan Anda mengelola siklus hidup pengembangan perangkat lunak setiap proyek secara terpusat, memberi Anda kemampuan untuk memastikan proyek di ruang Anda terus mengikuti praktik terbaik dengan perubahan atau perbaikan terbaru. Untuk informasi selengkapnya, lihat Bekerja dengan manajemen siklus hidup sebagai penulis cetak biru.

Cetak biru khusus memberikan kemampuan untuk memperbarui versi cetak biru terhadap proyek sebelumnya melalui resynthesis. Resintesis adalah proses menjalankan ulang sintesis cetak biru

dengan versi yang diperbarui atau kemampuan untuk menggabungkan perbaikan dan perubahan ke dalam proyek yang ada. Untuk informasi selengkapnya, lihat Konsep cetak biru kustom.

Untuk melihat cetak biru SDK dan contoh cetak biru, lihat repositori sumber terbuka. GitHub

Topik

Konsep cetak biru kustom

Berikut adalah beberapa konsep dan istilah yang harus Anda ketahui saat bekerja dengan cetak biru khusus di. CodeCatalyst

Topik

- Opsi sebagian
- Projen

Proyek cetak biru

Proyek cetak biru memberi Anda kemampuan untuk mengembangkan dan mempublikasikan cetak biru ke ruang Anda. Repositori sumber dibuat selama proses pembuatan proyek, dan nama repositori adalah yang Anda pilihsaat memasukkan detail sumber daya Proyek. Selama proses pembuatan cetak biru, jika Anda memilih untuk menghasilkan rilis alur kerja, alur kerja penerbitan dibuat dalam cetak biru Anda dengan cetak biru Blueprint Builder. Alur kerja secara otomatis menerbitkan versi terbaru Anda.

Cetak biru luar angkasa

Anda dapat melihat dan mengelola semua cetak biru dari tabel cetak biru Space saat Anda menavigasi ke bagian Cetak Biru di ruang Anda. Setelah cetak biru dipublikasikan ke ruang Anda, mereka tersedia sebagai cetak biru ruang untuk ditambahkan dan dihapus dari katalog cetak biru ruang Anda. Anda juga dapat mengelola izin penerbitan dan menghapus cetak biru dari bagian Cetak Biru di ruang Anda. Untuk informasi selengkapnya, lihat Melihat detail, versi, dan proyek cetak biru kustom.

Katalog cetak biru luar angkasa

Anda dapat melihat semua cetak biru kustom yang ditambahkan dari katalog cetak biru ruang. Di sinilah anggota ruang dapat memilih cetak biru kustom Anda untuk membuat proyek baru. Katalog ini berbeda dengan CodeCatalyst katalog, yang sudah memiliki cetak biru yang tersedia untuk semua anggota ruang angkasa. Untuk informasi selengkapnya, lihat Membuat proyek yang komprehensif dengan CodeCatalyst cetak biru.

Sintesis

Sintesis adalah proses menghasilkan bundel CodeCatalyst proyek yang mewakili kode sumber, konfigurasi, dan sumber daya dalam sebuah proyek. Bundel ini kemudian digunakan oleh operasi API CodeCatalyst penerapan untuk menyebarkan ke dalam proyek. Prosesnya dapat dijalankan secara lokal sambil mengembangkan cetak biru khusus Anda untuk meniru pembuatan proyek tanpa harus membuat proyek. CodeCatalyst Perintah berikut dapat digunakan untuk melakukan sintesis:

```
yarn blueprint:synth            # fast mode
```

```
yarn blueprint:synth --cache     # wizard emulation mode
```

Cetak biru dimulai dengan sendirinya dengan memanggil `blueprint.ts` kelas utama dengan opsi yang digabungkan. `defaults.json` Bundel proyek baru dihasilkan di bawah `synth/synth.[options-name]/proposed-bundle/` folder. Outputnya mencakup bundel proyek yang dihasilkan cetak biru kustom, mengingat opsi yang Anda tetapkan, termasuk opsi sebagian yang mungkin telah Anda konfigurasikan.

Resintesis

Resintesis adalah proses regenerasi cetak biru dengan opsi cetak biru yang berbeda atau versi cetak biru dari proyek yang ada. Sebagai penulis cetak biru, Anda dapat menentukan strategi penggabungan kustom dalam kode cetak biru kustom. Anda juga dapat menentukan batas kepemilikan dalam `.ownership-file` untuk menentukan di bagian mana dari basis kode cetak biru diizinkan untuk diperbarui. Sementara cetak biru kustom dapat mengusulkan pembaruan ke`.ownership-file`, pengembang proyek yang menggunakan cetak biru khusus dapat menentukan batas kepemilikan untuk proyek mereka. Anda dapat menjalankan resynthesis secara lokal, dan menguji serta memperbarui sebelum menerbitkan cetak biru kustom Anda. Gunakan perintah berikut untuk melakukan resintesis:

```
yarn blueprint:resynth             # fast mode
yarn blueprint:resynth --cache     # wizard emulation mode
```

Cetak biru dimulai dengan sendirinya dengan memanggil `blueprint.ts` kelas utama dengan opsi yang digabungkan. `defaults.json` Bundel proyek baru dihasilkan di bawah `synth/resynth.[options-name]/` folder. Outputnya mencakup bundel proyek yang dihasilkan cetak biru kustom, mengingat opsi yang Anda tetapkan, termasuk opsi sebagian yang mungkin telah Anda konfigurasikan.

Isi berikut dibuat setelah proses sintesis dan resintesis:

- proposed-bundle - Output sintesis ketika dijalankan dengan opsi baru untuk versi cetak biru target.
- existing-bundle - Sebuah tiruan dari proyek Anda yang ada. Jika tidak ada apa pun di folder ini, itu dihasilkan dengan output yang sama seperti dia`proposed-bundle`.
- ancestor-bundle - Sebuah tiruan dari apa yang akan dihasilkan cetak biru Anda saat dijalankan dengan versi sebelumnya, opsi sebelumnya, atau kombinasi. Jika tidak ada apa pun di folder ini, itu dihasilkan dengan output yang sama dengan file`proposed-bundle`.

- resolved-bundle - Bundel selalu dibuat ulang dan default menjadi penggabungan tiga arah antara,, dan. `proposed-bundle` `existing-bundle` `ancestor-bundle` Bundel ini memberikan emulasi tentang apa yang akan dihasilkan resintesis secara lokal.

Untuk mempelajari lebih lanjut tentang bundel keluaran cetak biru, lihat. Menghasilkan file dengan resynthesis

Opsi sebagian

Anda dapat menambahkan variasi opsi di bawah `src/wizard-configuration/` yang tidak harus menghitung keseluruhan `Options` antarmuka, dan opsi digabungkan di atas file. `defaults.json` Itu memungkinkan Anda untuk menyesuaikan kasus uji di seluruh opsi tertentu.

Contoh:

`Options`Antarmuka:

```
{
  language: "Python" | "Java" | "Typescript",
  repositoryName: string
  ...
}
```

`defaults.json`berkas:

```
{
  language: "Python",
  repositoryName: "Myrepo"
  ...
}
```

Tes konfigurasi tambahan:

- ```
  #wizard-config-typescript-test.json
  {
    language: "Typescript",
  }
  ```
- ```
  #wizard-config-java-test.json
  ```

```
{
  language: "Java",
}
```

Projen

Projen adalah alat sumber terbuka yang digunakan cetak biru khusus untuk menjaga diri mereka diperbarui dan konsisten. Cetak biru datang sebagai paket Projen karena kerangka kerja ini memberi Anda kemampuan untuk membangun, menggabungkan, dan menerbitkan proyek, dan Anda dapat menggunakan antarmuka untuk mengelola konfigurasi dan pengaturan proyek.

Anda dapat menggunakan Projen untuk memperbarui cetak biru dalam skala besar, bahkan setelah mereka dibuat. Alat Projen adalah teknologi yang mendasari di balik sintesis cetak biru yang menghasilkan bundel proyek. Projen memiliki konfigurasi untuk sebuah proyek, dan seharusnya tidak memengaruhi Anda sebagai penulis cetak biru. Anda dapat menjalankan `yarn projen` untuk membuat ulang konfigurasi proyek Anda setelah menambahkan dependensi, atau Anda dapat mengubah opsi dalam file. `projenrc.ts` Projen juga merupakan alat generasi yang mendasari cetak biru khusus untuk mensintesis proyek. Untuk informasi lebih lanjut, lihat GitHub halaman projen. Untuk mempelajari lebih lanjut tentang bekerja dengan Projen, lihat dokumentasi Projen dan Cara menyederhanakan penyiapan proyek dengan Projen.

Memulai dengan cetak biru khusus

Selama proses pembuatan cetak biru, Anda dapat mengonfigurasi cetak biru dan menghasilkan pratinjau sumber daya proyek. Setiap cetak biru kustom dikelola oleh CodeCatalyst proyek, yang berisi alur kerja secara default untuk dipublikasikan ke katalog cetak biru ruang.

Saat mengonfigurasi detail cetak biru kustom Anda, Anda juga dapat memilih untuk menyimpan kode sumber cetak biru Anda di repositori pihak ketiga, di mana Anda masih dapat mengelola cetak biru khusus dan memanfaatkan kemampuan manajemen siklus hidup untuk menjaga proyek ruang Anda tetap disinkronkan saat cetak biru kustom dimodifikasi. Untuk informasi selengkapnya, lihat Tambahkan fungsionalitas ke proyek dengan ekstensi di CodeCatalyst dan Bekerja dengan manajemen siklus hidup sebagai penulis cetak biru.

Topik

- Prasyarat
- Langkah 1: Buat cetak biru khusus di CodeCatalyst

- Langkah 2: Kembangkan cetak biru khusus dengan komponen
- Langkah 3: Pratinjau cetak biru khusus
- (Opsional) Langkah 4: Publikasikan versi pratinjau cetak biru kustom

Prasyarat

Sebelum membuat cetak biru khusus, pertimbangkan persyaratan berikut:

- CodeCatalyst Ruang Anda harus tingkat Enterprise. Untuk informasi selengkapnya, lihat Mengelola penagihan di Panduan CodeCatalyst Administrator Amazon.
- Anda harus memiliki administrator Space atau peran pengguna Power untuk membuat cetak biru kustom. Untuk informasi selengkapnya, lihat Memberikan akses dengan peran pengguna.

Langkah 1: Buat cetak biru khusus di CodeCatalyst

Saat Anda membuat cetak biru khusus dari pengaturan ruang Anda, repositori dibuat untuk Anda. Repositori mencakup semua sumber daya yang diperlukan yang harus Anda miliki untuk mengembangkan cetak biru Anda sebelum menerbitkannya ke katalog cetak biru ruang.

Untuk membuat cetak biru kustom

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, navigasikan ke ruang tempat Anda ingin membuat cetak biru khusus.
3. Di dasbor ruang, pilih tab Pengaturan, lalu pilih Blueprints.
4. Pilih Buat cetak biru.
5. Di bawah Nama cetak biru Anda, masukkan nama yang ingin Anda tetapkan ke proyek Anda dan nama sumber daya yang terkait. Nama harus unik di dalam ruang Anda.
6. (Opsional) Secara default, kode sumber yang dibuat oleh cetak biru disimpan dalam repositori. CodeCatalyst Atau, Anda dapat memilih untuk menyimpan kode sumber cetak biru di repositori pihak ketiga. Untuk informasi selengkapnya, lihat Tambahkan fungsionalitas ke proyek dengan ekstensi di CodeCatalyst.

   Lakukan salah satu hal berikut tergantung pada penyedia repositori pihak ketiga yang ingin Anda gunakan:

   - GitHub repositori: Hubungkan akun. GitHub

Pilih menu tarik-turun Advanced, pilih GitHub sebagai penyedia repositori, lalu pilih GitHub akun tempat Anda ingin menyimpan kode sumber yang dibuat oleh cetak biru.

> **Note**
>
> Jika Anda menggunakan koneksi ke GitHub akun, Anda harus membuat koneksi pribadi untuk membuat pemetaan identitas antara CodeCatalyst identitas dan GitHub identitas Anda. Untuk informasi selengkapnya, lihat Koneksi pribadi dan Mengakses GitHub sumber daya dengan koneksi pribadi.

- Bitbucket: Hubungkan ruang kerja Bitbucket.

  Pilih menu tarik-turun Advanced, pilih Bitbucket sebagai penyedia repositori, lalu pilih ruang kerja Bitbucket tempat Anda ingin menyimpan kode sumber yang dibuat oleh cetak biru.

7. Di bawah detail Blueprint, lakukan hal berikut:

   a. Di kolom input teks nama tampilan Blueprint, masukkan nama yang akan muncul di katalog cetak biru ruang Anda.

   b. Di bidang masukan teks Deskripsi, masukkan deskripsi untuk cetak biru kustom Anda.

   c. Di kolom masukan teks nama penulis, masukkan nama penulis untuk cetak biru kustom Anda.

   d. (Opsional) Pilih Pengaturan lanjutan.

      i. Pilih + Tambahkan untuk menambahkan tag yang ditambahkan ke `package.json` file.

      ii. Pilih menu tarik-turun Lisensi, lalu pilih lisensi untuk cetak biru kustom Anda.

      iii. Di kolom input teks nama paket Blueprint, masukkan nama untuk mengidentifikasi paket cetak biru Anda.

      iv. Secara default, alur kerja rilis dihasilkan menggunakan cetak biru penerbitan dalam proyek Anda yang disebut Blueprint Builder. Alur kerja menerbitkan versi cetak biru terbaru ke ruang Anda saat Anda mendorong perubahan karena izin penerbitan diaktifkan oleh alur kerja rilis. Untuk mematikan pembuatan alur kerja, hapus centang pada kotak centang Rilis alur kerja.

8. (Opsional) Proyek cetak biru dilengkapi dengan kode yang telah ditentukan untuk mendukung penerbitan cetak biru ke katalog ruang. Untuk melihat file definisi dengan pembaruan berdasarkan pilihan parameter proyek yang Anda buat, pilih Lihat kode atau Lihat alur kerja dari Buat pratinjau cetak biru.

9. Pilih Buat cetak biru.

Jika Anda tidak menonaktifkan pembuatan alur kerja untuk cetak biru kustom Anda, alur kerja secara otomatis mulai berjalan saat cetak biru Anda dibuat. Saat alur kerja selesai, cetak biru kustom Anda tersedia untuk ditambahkan ke katalog cetak biru ruang Anda secara default. Anda dapat menonaktifkan izin penerbitan jika Anda tidak ingin versi cetak biru terbaru dipublikasikan secara otomatis ke ruang Anda. Untuk informasi selengkapnya, lihat Menyetel izin penerbitan untuk cetak biru kustom dan Menjalankan alur kerja.

Karena alur kerja penerbitan yang `blueprint-release` disebut dibuat menggunakan cetak biru, cetak biru dapat ditemukan sebagai cetak biru yang diterapkan dalam proyek Anda. Untuk informasi selengkapnya, lihat Menerapkan cetak biru dalam proyek untuk menambahkan sumber daya dan Memutuskan cetak biru dari proyek untuk menghentikan pembaruan.

Langkah 2: Kembangkan cetak biru khusus dengan komponen

Wizard cetak biru dihasilkan saat Anda membuat cetak biru khusus, dan dapat dimodifikasi dengan komponen saat mengembangkan cetak biru kustom. Anda dapat memperbarui `src/defaults.json` file `src/blueprints.js` dan untuk memodifikasi wizard.

> **Important**
>
> Jika Anda ingin menggunakan paket cetak biru dari sumber eksternal, pertimbangkan risiko yang mungkin datang dengan paket tersebut. Anda bertanggung jawab atas cetak biru khusus yang Anda tambahkan ke ruang Anda dan kode yang dihasilkannya.

Buat Lingkungan Pengembang di CodeCatalyst proyek Anda dengan lingkungan pengembangan terintegrasi (IDE) yang didukung sebelum mengonfigurasi kode cetak biru Anda. Lingkungan Pengembang diperlukan untuk bekerja dengan alat dan paket yang diperlukan.

Untuk membuat Lingkungan Dev

1. Di panel navigasi, lakukan salah satu hal berikut:
   a. Pilih Ikhtisar, lalu arahkan ke bagian My Dev Environments.
   b. Pilih Kode, lalu pilih Dev Environments.
   c. Pilih Kode, pilih repositori Sumber, dan pilih repositori yang Anda buat saat membuat cetak biru Anda.

2. Pilih Buat Lingkungan Pengembang.
3. Pilih IDE yang didukung dari menu tarik-turun. Lihat Lingkungan pengembangan terintegrasi yang didukung untuk Lingkungan Pengembang untuk informasi selengkapnya.
4. Pilih Bekerja di cabang yang ada, dan dari menu dropdown cabang yang ada, pilih cabang fitur yang Anda buat.
5. (Opsional) Di kolom Alias - input teks opsional, masukkan alias untuk mengidentifikasi Lingkungan Pengembang.
6. Pilih Buat. Saat Dev Environment Anda sedang dibuat, kolom status Dev Environment menampilkan Mulai, dan kolom status menampilkan Berjalan saat Lingkungan Dev telah dibuat.

Untuk informasi selengkapnya, lihat Menulis dan memodifikasi kode dengan Dev Environments di CodeCatalyst.

Untuk mengembangkan cetak biru kustom Anda

1. Di terminal yang berfungsi, gunakan `yarn` perintah berikut untuk menginstal dependensi:

   ```
   yarn
   ```

   Alat dan paket yang diperlukan tersedia melalui CodeCatalyst Dev Environment, termasuk Yarn. Jika Anda mengerjakan cetak biru khusus tanpa Dev Environment, instal Yarn ke sistem Anda terlebih dahulu. Untuk informasi selengkapnya, lihat dokumentasi instalasi Yarn.
2. Kembangkan cetak biru khusus Anda sehingga dikonfigurasi sesuai preferensi Anda. Anda dapat memodifikasi wizard cetak biru Anda dengan menambahkan komponen. Lihat informasi selengkapnya di Mengembangkan cetak biru khusus untuk memenuhi persyaratan proyek, Memodifikasi fitur cetak biru dengan wizard front-end, dan Menerbitkan cetak biru khusus ke spasi.

Langkah 3: Pratinjau cetak biru khusus

Setelah menyiapkan dan mengembangkan cetak biru kustom Anda, Anda dapat melihat pratinjau dan mempublikasikan versi pratinjau cetak biru Anda ke ruang Anda. Versi pratinjau memberi Anda kemampuan untuk memeriksa apakah cetak biru adalah apa yang Anda inginkan sebelum digunakan untuk membuat proyek baru atau diterapkan pada proyek yang ada.

Untuk melihat pratinjau cetak biru kustom

1. Di terminal yang berfungsi, gunakan `yarn` perintah berikut:

```
yarn blueprint:preview
```

2. Arahkan ke `See this blueprint at:` tautan yang disediakan untuk melihat pratinjau cetak biru kustom Anda.
3. Periksa apakah UI, termasuk teks, muncul seperti yang Anda harapkan berdasarkan konfigurasi Anda. Jika Anda ingin mengubah cetak biru kustom Anda, Anda dapat mengedit `blueprint.ts` file, mensintesis ulang cetak biru, dan kemudian mempublikasikan versi pratinjau lagi. Untuk informasi selengkapnya, lihat Resintesis.

(Opsional) Langkah 4: Publikasikan versi pratinjau cetak biru kustom

Anda dapat mempublikasikan versi pratinjau cetak biru kustom Anda ke ruang Anda jika Anda ingin menambahkannya ke katalog cetak biru ruang Anda. Ini memungkinkan Anda untuk melihat cetak biru sebagai pengguna sebelum menambahkan versi non-pratinjau ke katalog. Versi pratinjau memungkinkan Anda untuk mempublikasikan tanpa mengambil versi sebenarnya. Misalnya, jika Anda mengerjakan `0.0.1` versi, Anda dapat mempublikasikan dan menambahkan versi pratinjau, sehingga pembaruan baru untuk versi kedua dapat dipublikasikan dan ditambahkan sebagai`0.0.2`.

Untuk mempublikasikan versi pratinjau cetak biru kustom

Arahkan ke `Enable version` *`[version number]`* `at:` tautan yang disediakan untuk mengaktifkan cetak biru kustom Anda. Tautan ini disediakan saat menjalankan `yarn` perintah diLangkah 3: Pratinjau cetak biru khusus.

Setelah membuat, mengembangkan, melihat pratinjau, dan menerbitkan cetak biru kustom Anda, Anda dapat mempublikasikan dan menambahkan versi cetak biru akhir ke katalog cetak biru ruang Anda. Lihat informasi yang lebih lengkap di Menerbitkan cetak biru khusus ke spasi dan Menambahkan cetak biru khusus ke katalog ruang.

Tutorial: Membuat dan memperbarui aplikasi React

Sebagai penulis cetak biru, Anda dapat mengembangkan dan menambahkan cetak biru khusus ke katalog cetak biru ruang Anda. Cetak biru ini kemudian dapat digunakan oleh anggota ruang angkasa untuk membuat proyek baru atau menerapkannya pada proyek yang ada. Anda dapat terus membuat perubahan pada cetak biru Anda yang kemudian tersedia sebagai pembaruan melalui permintaan tarik.

Tutorial ini memberikan panduan dari perspektif penulis cetak biru dan perspektif pengguna cetak biru. Tutorial ini menunjukkan cara membuat cetak biru aplikasi web satu halaman React. Cetak biru tersebut kemudian digunakan untuk membuat proyek baru. Ketika cetak biru diperbarui dengan perubahan, proyek yang dibuat dari cetak biru menggabungkan perubahan tersebut melalui permintaan tarik.

Topik

Prasyarat

Untuk membuat dan memperbarui cetak biru kustom, Anda harus telah menyelesaikan tugas sebagai berikut: Siapkan dan masuk ke CodeCatalyst

- Memiliki ID AWS Pembangun untuk masuk CodeCatalyst.
- Milik ruang dan memiliki administrator Space atau peran pengguna Power yang ditetapkan untuk Anda di ruang itu. Lihat informasi selengkapnya di Menciptakan ruang, Memberikan izin ruang kepada pengguna, dan Peran administrator ruang.

Langkah 1: Buat cetak biru khusus

Saat Anda membuat cetak biru kustom, sebuah CodeCatalyst proyek dibuat yang berisi kode sumber cetak biru dan alat pengembangan serta sumber daya Anda. Proyek Anda adalah tempat Anda akan mengembangkan, menguji, dan menerbitkan cetak biru.

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, arahkan ke ruang tempat Anda ingin membuat cetak biru.

3. Pilih Pengaturan untuk menavigasi ke pengaturan ruang.
4. Di tab Pengaturan ruang, pilih Blueprints dan pilih Create blueprint.
5. Perbarui bidang di wizard pembuatan cetak biru dengan nilai berikut:

   - Dalam nama Blueprint, masukkan. `react-app-blueprint`
   - Di Nama Tampilan Blueprint, masukkan. `react-app-blueprint`
6. Secara opsional, pilih Lihat kode untuk melihat pratinjau kode sumber cetak biru untuk cetak biru Anda. Demikian juga, pilih Lihat alur kerja untuk melihat alur kerja yang akan dibuat dalam proyek yang membangun dan menerbitkan cetak biru.
7. Pilih Buat cetak biru.
8. Setelah cetak biru Anda dibuat, Anda akan dibawa ke proyek cetak biru. Proyek ini berisi kode sumber cetak biru, bersama dengan alat dan sumber daya yang Anda butuhkan untuk mengembangkan, menguji, dan mempublikasikan cetak biru. Alur kerja rilis dibuat dan secara otomatis mempublikasikan cetak biru Anda ke ruang.
9. Sekarang proyek cetak biru dan cetak biru Anda dibuat, langkah selanjutnya adalah mengkonfigurasinya dengan memperbarui kode sumber. Anda dapat menggunakan Dev Environments untuk membuka dan mengedit repositori sumber Anda langsung di browser Anda.

   Di panel navigasi, pilih Kode, lalu pilih Lingkungan Dev.
10. Pilih Create Dev Environment dan kemudian pilih AWS Cloud9 (di browser).
11. Pertahankan pengaturan default dan pilih Buat.
12. Di AWS Cloud9 terminal, navigasikan ke direktori proyek cetak biru Anda dengan menjalankan perintah berikut:

    ```
    cd react-app-blueprint
    ```
13. `static-assets`Folder dibuat dan diisi secara otomatis saat cetak biru dibuat. Dalam tutorial ini, Anda akan menghapus folder default dan menghasilkan yang baru untuk cetak biru aplikasi reaksi.

    Hapus folder static-assets dengan menjalankan perintah berikut:

    ```
    rm -r static-assets
    ```

    AWS Cloud9 dibangun di atas platform berbasis Linux. Jika Anda menggunakan sistem operasi Windows, Anda dapat menggunakan perintah berikut sebagai gantinya:

```
rmdir /s /q static-assets
```

14. Sekarang folder default dihapus, buat `static-assets` folder untuk cetak biru aplikasi reaksi dengan menjalankan perintah berikut:

    ```
    npx create-react-app static-assets
    ```

    Jika diminta, masukkan y untuk melanjutkan.

    Aplikasi reaksi baru dibuat di `static-assets` folder dengan paket yang diperlukan. Perubahan perlu didorong ke repositori CodeCatalyst sumber jarak jauh Anda.

15. Pastikan Anda memiliki perubahan terbaru, lalu komit dan dorong perubahan ke repositori CodeCatalyst sumber cetak biru dengan menjalankan perintah berikut:

    ```
    git pull
    ```

    ```
    git add .
    ```

    ```
    git commit -m "Add React app to static-assets"
    ```

    ```
    git push
    ```

Saat perubahan didorong ke repositori sumber cetak biru, alur kerja rilis dimulai secara otomatis. Alur kerja ini menambah versi cetak biru, membangun cetak biru, dan menerbitkannya ke ruang Anda. Pada langkah berikutnya, Anda akan menavigasi ke alur kerja rilis untuk melihat bagaimana kinerjanya.

Langkah 2: Lihat alur kerja rilis

1. Di CodeCatalyst konsol, di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih alur kerja blueprint-release.
3. Anda dapat melihat alur kerja memiliki tindakan untuk membangun dan mempublikasikan cetak biru.

4. Di bawah Lari terbaru, pilih tautan jalankan alur kerja untuk melihat proses dari perubahan kode yang Anda buat.
5. Setelah proses selesai, versi cetak biru baru Anda diterbitkan. Versi cetak biru yang dipublikasikan dapat dilihat di Pengaturan ruang Anda, tetapi tidak dapat digunakan dalam proyek hingga ditambahkan ke katalog cetak biru ruang. Pada langkah berikutnya, Anda akan menambahkan cetak biru ke katalog.

Langkah 3: Tambahkan cetak biru ke katalog

Menambahkan cetak biru ke katalog cetak biru ruang membuat cetak biru tersedia untuk digunakan di semua proyek dalam satu ruang. Anggota luar angkasa dapat menggunakan cetak biru untuk membuat proyek baru atau menerapkannya pada proyek yang ada.

1. Di CodeCatalyst konsol, navigasikan kembali ke ruang.
2. Pilih Pengaturan, lalu pilih Blueprints.
3. Pilih react-app-blueprint, lalu pilih Tambahkan ke katalog.
4. Pilih Simpan.

Langkah 4: Buat proyek dengan cetak biru

Sekarang cetak biru ditambahkan ke katalog, itu dapat digunakan dalam proyek. Pada langkah ini, Anda akan membuat proyek dengan cetak biru yang baru saja Anda buat. Pada langkah selanjutnya, Anda akan memperbarui proyek ini dengan memperbarui dan menerbitkan versi baru cetak biru.

1. Pilih tab Projects dan kemudian pilih Create project.
2. Pilih Blueprints Space, lalu pilih. react-app-blueprint

> **Note**
>
> Setelah cetak biru dipilih, Anda dapat melihat isi file cetak biru. `README.md`

3. Pilih Berikutnya.
4.

> **Note**
>
> Isi wizard pembuatan proyek ini dapat dikonfigurasi dalam cetak biru.

Masukkan nama proyek sebagai pengguna cetak biru. Untuk tutorial ini, masukkan `react-app-project`. Untuk informasi selengkapnya, lihat Mengembangkan cetak biru khusus untuk memenuhi persyaratan proyek.

Selanjutnya, Anda akan membuat pembaruan ke cetak biru dan menambahkan versi baru ke katalog, yang akan Anda gunakan untuk memperbarui proyek ini.

Langkah 5: Perbarui cetak biru

Setelah cetak biru digunakan untuk membuat proyek baru atau diterapkan pada proyek yang ada, Anda dapat terus membuat pembaruan sebagai penulis cetak biru. Pada langkah ini, Anda akan membuat perubahan pada cetak biru dan secara otomatis menerbitkan versi baru ke ruang. Versi baru kemudian dapat ditambahkan sebagai versi katalog.

1. Arahkan ke react-app-blueprintproyek yang dibuat diTutorial: Membuat dan memperbarui aplikasi React.
2. Buka Lingkungan Pengembang yang dibuat diTutorial: Membuat dan memperbarui aplikasi React.
   a. Di panel navigasi, pilih Kode, lalu pilih Lingkungan Dev.
   b. Dari tabel, temukan Lingkungan Dev, lalu pilih Buka di AWS Cloud9 (di browser).
3. Ketika alur kerja rilis cetak biru dijalankan, itu menambah versi cetak biru dengan memperbarui file. `package.json` Tarik perubahan itu dengan menjalankan perintah berikut di AWS Cloud9 terminal:

```
git pull
```

4. Arahkan ke `static-assets` folder dengan menjalankan perintah berikut:

```
cd /projects/react-app-blueprint/static-assets
```

5. Buat `hello-world.txt` file dalam `static-assets` folder dengan menjalankan perintah berikut:

```
touch hello-world.txt
```

AWS Cloud9 dibangun di atas platform berbasis Linux. Jika Anda menggunakan sistem operasi Windows, Anda dapat menggunakan perintah berikut sebagai gantinya:

```
echo > hello-world.text
```

6. Di navigasi sebelah kiri, klik dua kali `hello-world.txt` file untuk membukanya di editor, dan tambahkan konten berikut:

```
Hello, world!
```

Simpan file tersebut.

7. Pastikan Anda memiliki perubahan terbaru, lalu komit dan dorong perubahan ke repositori CodeCatalyst sumber cetak biru dengan menjalankan perintah berikut:

```
git pull
```

```
git add .
```

```
git commit -m "prettier setup"
```

```
git push
```

Mendorong perubahan memulai alur kerja rilis, yang secara otomatis akan mempublikasikan versi baru cetak biru ke ruang.

Langkah 6: Perbarui versi katalog cetak biru yang diterbitkan ke versi baru

Setelah cetak biru digunakan untuk membuat proyek baru atau diterapkan pada proyek yang ada, Anda masih dapat memperbarui cetak biru sebagai penulis cetak biru. Pada langkah ini, Anda akan membuat perubahan pada cetak biru dan mengubah versi katalog cetak biru.

1. Di CodeCatalyst konsol, navigasikan kembali ke ruang.
2. Pilih Pengaturan, lalu pilih Blueprints.
3. Pilih react-app-blueprint, lalu pilih Kelola versi katalog.
4. Pilih versi baru, lalu pilih Simpan.

Langkah 7: Perbarui proyek dengan versi cetak biru baru

Versi baru sekarang tersedia di katalog cetak biru ruang angkasa. Sebagai pengguna cetak biru, Anda dapat memperbarui versi untuk proyek yang dibuat di. Langkah 4: Buat proyek dengan cetak biru Ini memastikan Anda memiliki perubahan dan perbaikan terbaru yang diperlukan untuk memenuhi praktik terbaik.

1. Di CodeCatalyst konsol, navigasikan ke react-app-projectproyek yang dibuat diLangkah 4: Buat proyek dengan cetak biru.
2. Di panel navigasi, pilih Cetak biru.
3. Pilih Perbarui cetak biru di kotak info.
4. Di panel perubahan Kode sisi kanan, Anda dapat melihat `hello-world.txt` dan `package.json` memperbarui.
5. Pilih Terapkan pembaruan.

Memilih Terapkan pembaruan akan membuat permintaan tarik dalam proyek dengan perubahan dari versi cetak biru yang diperbarui. Untuk membuat pembaruan pada proyek, Anda harus menggabungkan permintaan tarik. Lihat informasi yang lebih lengkap di Meninjau permintaan tarik dan Menggabungkan permintaan tarik.

1. Di tabel Blueprints, temukan cetak biru. Di kolom Status, pilih Permintaan tarik tertunda, lalu pilih tautan ke permintaan tarik terbuka.
2. Tinjau permintaan tarik, lalu pilih Gabung.
3. Pilih Fast forward merge untuk mempertahankan nilai default, lalu pilih Merge.

Langkah 8: Lihat perubahan dalam proyek

Perubahan pada cetak biru sekarang tersedia di proyek Anda setelahnya. Langkah 7: Perbarui proyek dengan versi cetak biru baru Sebagai pengguna cetak biru, Anda dapat melihat perubahan di repositori sumber.

1. Di panel navigasi, pilih Repositori sumber, lalu pilih nama repositori sumber yang dibuat saat proyek dibuat.
2. Di bawah File, Anda dapat melihat `hello-world.txt` file yang dibuat di fileLangkah 5: Perbarui cetak biru.
3. Pilih `hello-world.txt` untuk melihat konten file.

Manajemen siklus hidup memberi penulis cetak biru kemampuan mengelola siklus hidup pengembangan perangkat lunak secara terpusat dari setiap proyek yang berisi cetak biru tertentu. Seperti yang terlihat dalam tutorial ini, Anda dapat mendorong pembaruan ke cetak biru yang kemudian dapat digabungkan oleh proyek yang menggunakan cetak biru untuk membuat proyek baru atau menerapkannya pada proyek yang sudah ada. Untuk informasi selengkapnya, lihat Bekerja dengan manajemen siklus hidup sebagai penulis cetak biru.

Bekerja dengan manajemen siklus hidup sebagai penulis cetak biru

Manajemen siklus hidup memungkinkan Anda menyimpan sejumlah besar proyek yang disinkronkan dari satu sumber praktik terbaik yang umum. Ini menskalakan propagasi perbaikan dan pemeliharaan sejumlah proyek di seluruh siklus pengembangan perangkat lunak mereka. Manajemen siklus hidup merampingkan kampanye internal, perbaikan keamanan, audit, peningkatan runtime, perubahan praktik terbaik, dan praktik pemeliharaan lainnya karena standar tersebut didefinisikan di satu tempat dan secara otomatis selalu diperbarui secara terpusat saat standar baru diterbitkan.

Ketika versi baru cetak biru Anda diterbitkan, semua proyek yang berisi cetak biru tersebut diminta untuk memperbarui ke versi terbaru. Sebagai penulis cetak biru, Anda juga dapat melihat versi cetak biru tertentu yang berisi setiap proyek untuk tujuan kepatuhan. Ketika ada konflik dalam repositori sumber yang ada, manajemen siklus hidup membuat permintaan tarik. Untuk semua sumber daya lainnya, seperti Dev Environment, semua pembaruan manajemen siklus hidup secara ketat membuat sumber daya baru. Pengguna bebas untuk menggabungkan atau tidak menggabungkan permintaan tarik tersebut. Ketika permintaan tarik tertunda digabungkan, versi cetak biru, termasuk opsi, yang digunakan dalam proyek Anda kemudian diperbarui. Untuk mempelajari tentang bekerja dengan manajemen siklus hidup sebagai pengguna cetak biru, lihat dan. Menggunakan manajemen siklus hidup pada proyek yang ada Menggunakan manajemen siklus hidup pada beberapa cetak biru dalam sebuah proyek

Topik

- Menguji manajemen siklus hidup untuk keluaran bundel dan konflik gabungan
- Menggunakan strategi gabungan untuk menghasilkan bundel dan menentukan file
- Mengakses objek konteks untuk detail proyek

Menguji manajemen siklus hidup untuk keluaran bundel dan konflik gabungan

Anda dapat menguji manajemen siklus hidup cetak biru secara lokal dan menggabungkan resolusi konflik. Serangkaian bundel di bawah `synth/` direktori yang mewakili berbagai fase pembaruan

siklus hidup dihasilkan. Untuk menguji manajemen siklus hidup, Anda dapat menjalankan perintah yarn berikut pada cetak biru Anda:. `yarn blueprint: resynth` Untuk mempelajari lebih lanjut tentang resintesis dan bundel, lihat Resintesis dan. Menghasilkan file dengan resynthesis

Menggunakan strategi gabungan untuk menghasilkan bundel dan menentukan file

Topik

- Menghasilkan file dengan resynthesis
- Menggunakan strategi penggabungan
- Menentukan file untuk pembaruan manajemen siklus hidup
- Menulis strategi penggabungan

Menghasilkan file dengan resynthesis

Resintesis dapat menggabungkan kode sumber yang dihasilkan oleh cetak biru dengan kode sumber yang sebelumnya dihasilkan oleh cetak biru yang sama, memungkinkan perubahan cetak biru untuk disebarkan ke proyek yang ada. Penggabungan dijalankan dari `resynth()` fungsi di seluruh bundel keluaran cetak biru. Resintesis pertama menghasilkan tiga bundel yang mewakili berbagai aspek cetak biru dan status proyek. Ini dapat dijalankan secara manual secara lokal dengan `yarn blueprint:resynth` perintah, yang akan membuat bundel jika belum ada. Bekerja secara manual dengan bundel akan memungkinkan Anda untuk mengejek dan menguji perilaku resintesis secara lokal. Secara default, cetak biru hanya menjalankan resintesis di seluruh repositori di bawah `src/*` karena hanya bagian dari bundel yang biasanya berada di bawah kendali sumber.

- `existing-bundle`- Bundel ini merupakan representasi dari status proyek yang ada. Ini secara buatan dibangun oleh komputasi sintesis untuk memberikan konteks cetak biru tentang apa yang ada dalam proyek yang disebarkannya (jika ada). Jika sesuatu sudah ada di lokasi ini saat menjalankan resynthesis secara lokal, itu akan diatur ulang dan dihormati sebagai tiruan. Jika tidak, itu akan diatur ke isi`ancestor-bundle`.
- `ancestor-bundle`- Ini adalah bundel yang mewakili keluaran cetak biru jika disintesis dengan beberapa opsi dan/atau versi sebelumnya. Jika ini adalah pertama kalinya cetak biru ini ditambahkan ke proyek, maka leluhurnya tidak ada, jadi itu disetel ke konten yang sama dengan. `existing-bundle` Secara lokal, jika bundel ini sudah ada di lokasi ini, itu akan dihormati sebagai tiruan.

- `proposed-bundle`- Ini adalah bundel yang mengolok-olok cetak biru jika disintesis dengan beberapa opsi dan/atau versi baru. Ini adalah bundel yang sama yang akan diproduksi oleh `synth()` fungsi. Secara lokal, bundel ini selalu diganti.

Setiap bundel dibuat selama fase resintesis yang dapat diakses dari kelas cetak biru di bawah. `this.context.resynthesisPhase`

- `resolved-bundle`- Ini adalah bundel terakhir, yang merupakan representasi dari apa yang dikemas dan dikerahkan ke proyek. CodeCatalyst Anda dapat melihat file dan perbedaan mana yang dikirim ke mekanisme penerapan. Ini adalah output dari `resynth()` fungsi menyelesaikan gabungan antara tiga bundel lainnya.

Penggabungan tiga arah diterapkan dengan mengambil perbedaan antara `ancestor-bundle` dan `proposed-bundle` dan menerapkannya pada `existing-bundle` untuk menghasilkan. `resolved-bundle` Semua strategi penggabungan menyelesaikan file ke file. `resolved-bundle` Resintesis menyelesaikan jangkauan bundel ini dengan strategi penggabungan cetak biru selama `resynth()` dan menghasilkan bundel yang diselesaikan dari hasilnya.

Menggunakan strategi penggabungan

Anda dapat menggunakan strategi gabungan yang dijual oleh pustaka cetak biru. Strategi ini menyediakan cara untuk menyelesaikan output file dan konflik untuk file yang disebutkan di Menghasilkan file dengan resynthesis bagian.

- `alwaysUpdate`- Strategi yang selalu menyelesaikan file yang diusulkan.
- `neverUpdate`- Strategi yang selalu menyelesaikan file yang ada.
- `onlyAdd`- Strategi yang menyelesaikan file yang diusulkan ketika file yang ada belum ada. Jika tidak, selesaikan ke file yang ada.
- `threeWayMerge`- Strategi yang melakukan penggabungan tiga arah antara file leluhur yang ada, diusulkan, dan yang sama. File yang diselesaikan mungkin berisi penanda konflik jika file tidak dapat digabungkan dengan bersih. Isi file yang disediakan harus dikodekan UTF-8 agar strategi menghasilkan output yang berarti. Strategi ini mencoba mendeteksi apakah file input biner. Jika strategi mendeteksi konflik gabungan dalam file biner, selalu mengembalikan file yang diusulkan.
- `preferProposed`- Strategi yang melakukan penggabungan tiga arah antara file leluhur yang ada, diusulkan, dan yang sama. Strategi ini menyelesaikan konflik dengan memilih sisi file yang diusulkan dari setiap konflik.

- `preferExisting`- Strategi yang melakukan penggabungan tiga arah antara file leluhur yang ada, diusulkan, dan yang sama. Strategi ini menyelesaikan konflik dengan memilih sisi file yang ada dari setiap konflik.

Untuk melihat kode sumber strategi penggabungan, lihat repositori sumber terbuka GitHub .

Menentukan file untuk pembaruan manajemen siklus hidup

Selama resintesis, cetak biru mengontrol bagaimana perubahan digabungkan ke dalam repositori sumber yang ada. Namun, Anda mungkin tidak ingin mendorong pembaruan ke setiap file dalam cetak biru Anda. Misalnya, kode contoh seperti stylesheet CSS dimaksudkan untuk spesifik proyek. Strategi penggabungan tiga arah adalah opsi default jika Anda tidak menentukan strategi lain. Cetak biru dapat menentukan file mana yang mereka miliki dan file mana yang tidak mereka miliki dengan menentukan strategi penggabungan pada konstruksi repositori itu sendiri. Cetak biru dapat memperbarui strategi penggabungannya, dan strategi terbaru dapat digunakan selama resintesis.

```
const sourceRepo = new SourceRepository(this, {
      title: 'my-repo',
    });
    sourceRepo.setResynthStrategies([
      {
        identifier: 'dont-override-sample-code',
        description: 'This strategy is applied accross all sample code. The blueprint
 will create sample code, but skip attempting to update it.',
        strategy: MergeStrategies.neverUpdate,
        globs: [
          '**/src/**',
          '**/css/**',
        ],
      },
    ]);
```

Beberapa strategi penggabungan dapat ditentukan, dan strategi terakhir diutamakan. File terungkap secara default three-way-merge mirip dengan Git. Ada beberapa strategi gabungan yang disediakan melalui `MergeStrategies` konstruksi, tetapi Anda dapat menulis sendiri. Strategi yang disediakan mematuhi driver strategi git merge.

Menulis strategi penggabungan

Selain menggunakan salah satu strategi penggabungan build yang disediakan, Anda juga dapat menulis strategi Anda sendiri. Strategi harus mematuhi antarmuka strategi standar. Anda harus

menulis fungsi strategi yang mengambil versi file dari,, dan `existing-bundle` `proposed-bundle``ancestor-bundle`, dan menggabungkannya menjadi satu file yang diselesaikan. Sebagai contoh:

```
type StrategyFunction = (
   /**
   * file from the ancestor bundle (if it exists)
   */
    commonAncestorFile: ContextFile | undefined,
   /**
   * file from the existing bundle (if it exists)
   */
    existingFile: ContextFile | undefined,
   /**
   * file from the proposed bundle (if it exists)
   */
    proposedFile: ContextFile | undefined,
    options?: {})
    /**
    * Return: file you'd like in the resolved bundle
    * passing undefined will delete the file from the resolved bundle
    */
=> ContextFile | undefined;
```

Jika file tidak ada (tidak terdefinisi), maka jalur file itu tidak ada di bundel lokasi tertentu.

Contoh:

```
strategies: [
          {
            identifier: 'dont-override-sample-code',
            description: 'This strategy is applied across all sample code. The
 blueprint will create sample code, but skip attempting to update it.',
            strategy: (ancestor, existing, proposed) => {
                const resolvedfile = ...
                ...
                // do something
                ...
                return resolvedfile
            },
            globs: [
              '**/src/**',
              '**/css/**',
```

```
          ],
        },
      ],
```

Mengakses objek konteks untuk detail proyek

Sebagai penulis cetak biru, Anda dapat mengakses konteks dari proyek cetak biru selama sintesis untuk mendapatkan informasi seperti ruang dan nama proyek, atau file yang ada di repositori sumber proyek. Anda juga bisa mendapatkan detail seperti fase resintesis yang dihasilkan cetak biru. Misalnya, Anda dapat mengakses konteks untuk mengetahui apakah Anda mensintesis ulang untuk menghasilkan bundel leluhur atau bundel yang diusulkan. Konteks kode yang ada kemudian dapat digunakan untuk mengubah kode Anda di repositori Anda. Misalnya, Anda dapat menulis strategi resintesis Anda sendiri untuk menetapkan standar kode tertentu. Strategi dapat ditambahkan ke `blueprint.ts` file untuk cetak biru kecil, atau Anda dapat membuat file terpisah untuk strategi.

Contoh berikut menunjukkan bagaimana Anda dapat menemukan file dalam konteks proyek, menetapkan pembuat alur kerja, dan menetapkan strategi resynthesis yang dijual cetak biru untuk file tertentu:

```
const contextFiles = this.context.project.src.findAll({
      fileGlobs: ['**/package.json'],
    });

    // const workflows = this.context.project.src.findAll({
    //   fileGlobs: ['**/.codecatalyst/**.yaml'],
    // });

    const security = new WorkflowBuilder(this, {
      Name: 'security-workflow',
    });
    new Workflow(this, repo, security.getDefinition());
    repo.setResynthStrategies([
      {
        identifier: 'force-security',
        globs: ['**/.codecatalyst/security-workflow.yaml'],
        strategy: MergeStrategies.alwaysUpdate,
      },
    ]);


    for (const contextFile of contextFiles) {
      const packageObject = JSON.parse(contextFile.buffer.toString());
```

```
      new SourceFile(internalRepo, contextFile.path, JSON.stringify({
        ...packageObject,
      }, null, 2));
    }
  }
```

Mengembangkan cetak biru khusus untuk memenuhi persyaratan proyek

Sebelum menerbitkan cetak biru khusus, Anda dapat mengembangkan cetak biru untuk memenuhi persyaratan tertentu. Anda dapat mengembangkan cetak biru kustom Anda dan menguji cetak biru dengan membuat proyek saat melihat pratinjau. Anda dapat mengembangkan cetak biru khusus untuk menyertakan komponen proyek, seperti kode sumber tertentu, koneksi akun, alur kerja, masalah, atau komponen lain yang dapat dibuat. CodeCatalyst

> **Important**
>
> Jika Anda ingin menggunakan paket cetak biru dari sumber eksternal, pertimbangkan risiko yang mungkin datang dengan paket tersebut. Anda bertanggung jawab atas cetak biru khusus yang Anda tambahkan ke ruang Anda dan kode yang dihasilkannya.

Untuk mengembangkan atau memperbarui cetak biru kustom

1. Lanjutkan Lingkungan Pengembang Anda. Untuk informasi selengkapnya, lihat Melanjutkan Lingkungan Pengembang.

   Jika Anda tidak memiliki Lingkungan Pengembang, Anda harus membuatnya terlebih dahulu. Untuk informasi selengkapnya, lihat Membuat Lingkungan Dev.
2. Buka terminal yang berfungsi di Lingkungan Pengembang Anda.
3. Jika Anda memilih alur kerja rilis saat membuat cetak biru, versi cetak biru terbaru akan dipublikasikan secara otomatis. Tarik perubahan untuk memastikan `package.json` file memiliki versi tambahan. Gunakan perintah berikut ini.

   ```
   git pull
   ```
4. Dalam `src/blueprint.ts` file, edit opsi cetak biru kustom Anda. `Options`Antarmuka ditafsirkan oleh CodeCatalyst wizard secara dinamis untuk menghasilkan antarmuka pengguna pilihan (UI). Anda dapat mengembangkan cetak biru kustom Anda dengan menambahkan komponen dan tag yang didukung. Untuk informasi selengkapnya, lihat Memodifikasi fitur

cetak biru dengan wizard front-end, Menambahkan komponen lingkungan ke cetak biru, Menambahkan komponen wilayah ke cetak biru, Menambahkan komponen repositori dan kode sumber ke cetak biru, Menambahkan komponen alur kerja ke cetak biru, dan Menambahkan komponen Dev Environments ke cetak biru.

Anda juga dapat melihat cetak biru SDK dan contoh cetak biru untuk dukungan tambahan saat mengembangkan cetak biru kustom Anda. Untuk informasi selengkapnya, lihat GitHub repositori sumber terbuka.

Cetak biru khusus menyediakan bundel pratinjau sebagai hasil dari sintesis yang berhasil. Bundel proyek mewakili kode sumber, konfigurasi, dan sumber daya dalam proyek, dan digunakan oleh operasi API CodeCatalyst penerapan untuk menyebarkan ke dalam proyek. Jika Anda ingin terus mengembangkan cetak biru kustom Anda, jalankan kembali proses sintesis cetak biru. Untuk informasi selengkapnya, lihat Konsep cetak biru kustom.

Memodifikasi fitur cetak biru dengan wizard front-end

Wizard pemilihan cetak biru aktif dibuat CodeCatalyst secara otomatis oleh `Options` antarmuka dalam file. `blueprint.ts` Wizard front-end mendukung modifikasi dan fitur cetak biru `Options` menggunakan komentar dan tag gaya JSDOC. Anda dapat menggunakan komentar dan tag gaya JSDOC untuk melakukan tugas. Misalnya, Anda dapat memilih teks yang ditampilkan di atas opsi, mengaktifkan fitur seperti validasi input, atau membuat opsi dapat dilipat. Wizard bekerja dengan menafsirkan pohon sintaks abstrak (AST) yang dihasilkan dari TypeScript tipe dari antarmuka. `Options` Wizard mengkonfigurasi dirinya sendiri secara otomatis ke jenis yang dijelaskan sebaik mungkin. Tidak semua jenis didukung. Jenis lain yang didukung termasuk pemilih wilayah dan pemilih lingkungan.

Berikut ini adalah contoh wizard yang menggunakan komentar dan tag JSDOC dengan cetak biru: `Options`

```
export interface Options {
  /**
   * What do you want to call your new blueprint?
   * @validationRegex /^[a-zA-Z0-9_]+$/
   * @validationMessage Must contain only upper and lowercase letters, numbers and
 underscores
   */
  blueprintName: string;
```

```
  /**
   * Add a description for your new blueprint.
   */
   description?: string;

   /**
    * Tags for your Blueprint:
    * @collapsed true
    */
  tags?: string[];
}
```

Nama tampilan setiap opsi `Options` antarmuka muncul secara `camelCase` default. Teks biasa dalam komentar gaya JSDOC ditampilkan sebagai teks di atas opsi di wizard.

Topik

- Tag yang didukung
- TypeScript Tipe yang didukung
- Berkomunikasi dengan pengguna selama sintesis

Tag yang didukung

Tag JSDOC berikut didukung oleh cetak biru kustom di wizard front-end. `Options`

@inlinePolicy. /jalur/ke/policy/file.json

- Membutuhkan - Opsi untuk menjadi tipe`Role`.
- Penggunaan - Memungkinkan Anda mengkomunikasikan kebijakan inline yang dibutuhkan peran. `policy.json`Jalur tersebut diharapkan berada di bawah kode sumber. Gunakan tag ini saat Anda memerlukan kebijakan khusus untuk suatu peran.
- Dependensi - **`blueprint-cli 0.1.12`** dan di atas
- Contoh - `@inlinePolicy ./deployment-policy.json`

```
environment: EnvironmentDefinition{
    awsAccountConnection: AccountConnection{
      /**
       * @inlinePolicy ./path/to/deployment-policy.json
       */
```

```
      cdkRole: Role[];
    };
   };
```

@trustPolicy. /jalur/ke/policy/file.json

- Membutuhkan - Opsi untuk menjadi tipe`Role`.
- Penggunaan - Memungkinkan Anda mengomunikasikan kebijakan kepercayaan yang dibutuhkan peran. `policy.json`Jalur tersebut diharapkan berada di bawah kode sumber. Gunakan tag ini saat Anda memerlukan kebijakan khusus untuk suatu peran.
- Dependensi - **`blueprint-cli 0.1.12`** dan di atas
- Contoh - `@trustPolicy ./trust-policy.json`

```
environment: EnvironmentDefinition{
    awsAccountConnection: AccountConnection{
      /**
       * @trustPolicy ./path/to/trust-policy.json
       */
      cdkRole: Role[];
    };
   };
```

Ekspresi @validationRegex Regex

- Membutuhkan - Opsi untuk menjadi string.
- Penggunaan - Melakukan validasi input pada opsi dengan menggunakan ekspresi dan tampilan regex yang diberikan. `@validationMessage`
- Contoh - `@validationRegex /^[a-zA-Z0-9_]+$/`
- Rekomendasi - Gunakan dengan`@validationMessage`. Pesan validasi kosong secara default.

@validationMessage string

- Memerlukan - `@validationRegex` atau kesalahan lain untuk meninjau penggunaan.
- Penggunaan - Menampilkan pesan validasi pada `@validation*` kegagalan.
- Contoh -`@validationMessage Must contain only upper and lowercase letters, numbers, and underscores`.

- Rekomendasi - Gunakan dengan`@validationMessage`. Pesan validasi kosong secara default.

@collapsed boolean (opsional)

- Membutuhkan - N/A
- Penggunaan - Boolean yang memungkinkan suboption untuk dilipat. Jika anotasi yang diciutkan hadir, nilai defaultnya adalah true. Mengatur nilai untuk `@collapsed false` membuat bagian yang dapat dilipat yang awalnya terbuka.
- Contoh - `@collapsed true`

@displayName string

- Membutuhkan - N/A
- Penggunaan - Mengubah nama tampilan opsi. Mengizinkan format selain CamelCase untuk nama tampilan.
- Contoh - `@displayName Blueprint Name`

@displayName string

- Membutuhkan - N/A
- Penggunaan - Mengubah nama tampilan opsi. Mengizinkan format selain CamelCase untuk nama tampilan.
- Contoh - `@displayName Blueprint Name`

Nomor @defaultEntropy

- Membutuhkan - Opsi untuk menjadi string.
- Penggunaan - Menambahkan string alfanumerik acak dari panjang tertentu ke opsi.
- Contoh - `@defaultEntropy 5`

@placeholder string (opsional)

- Membutuhkan - N/A
- Penggunaan - Mengubah placeholder bidang teks default.

- Contoh - `@placeholder type project name here`

Nomor @textArea (opsional)

- Membutuhkan - N/A
- Penggunaan - Mengubah input string menjadi komponen area teks untuk bagian teks yang lebih besar. Menambahkan angka menentukan jumlah baris. Defaultnya adalah lima baris.
- Contoh - `@textArea 10`

@hidden boolean (opsional)

- Membutuhkan - N/A
- Penggunaan - Menyembunyikan file dari pengguna kecuali pemeriksaan validasi gagal. Nilai default adalah true.
- Contoh - `@hidden`

@button boolean (opsional)

- Membutuhkan - N/A
- Penggunaan - Anotasi harus pada properti Boolean. Menambahkan tombol yang akan mensintesis sebagai true saat dipilih. Bukan toggle.
- Contoh - `buttonExample: boolean;`

```
/**
  * @button
  */
buttonExample: boolean;
```

@showName boolean (opsional)

- Membutuhkan - N/A
- Penggunaan - Hanya dapat digunakan pada jenis koneksi akun. Menampilkan masukan nama tersembunyi. Default ke `default_environment`.
- Contoh - `@showName true`

```
/**
  * @showName true
  */
accountConnection: AccountConnection<{
    ...
}>;
```

@ showEnvironmentType boolean (opsional)

- Membutuhkan - N/A
- Penggunaan - Hanya dapat digunakan pada jenis koneksi akun. Menampilkan menu dropdown tipe lingkungan tersembunyi. Semua koneksi default ke`production`. Pilihannya adalah Non-produksi atau Produksi.
- Contoh - `@showEnvironmentType true`

```
/**
  * @showEnvironmentType true
  */
accountConnection: AccountConnection<{
    ...
}>;
```

@forceDefault boolean (opsional)

- Membutuhkan - N/A
- Penggunaan - Menggunakan nilai default yang disediakan oleh penulis cetak biru bukan nilai yang digunakan sebelumnya oleh pengguna.
- Contoh - `forceDeafultExample: any;`

```
/**
  * @forceDefault
  */
forceDeafultExample: any;
```

@requires BluePrintNama

- Membutuhkan - Menganotasi antarmuka `Options`
- Penggunaan - Memperingatkan pengguna untuk menerapkan yang ditentukan `blueprintName` untuk proyek sebagai persyaratan untuk cetak biru saat ini.
- Contoh - `@requires '@amazon-codecatalyst/blueprints.blueprint-builder'`

```
/*
 * @requires '@amazon-codecatalyst/blueprints.blueprint-builder'
 */
export interface Options extends ParentOptions {
...
```

TypeScript Tipe yang didukung

TypeScript Jenis berikut didukung oleh cetak biru kustom `Options` di wizard front-end.

Jumlah

- Membutuhkan - Opsi untuk menjadi tipe`number`.
- Penggunaan - Menghasilkan bidang input angka.
- Contoh - `age: number`

```
{
  age: number
  ...
}
```

String

- Membutuhkan - Opsi untuk menjadi tipe`string`.
- Penggunaan - Menghasilkan bidang input string.
- Contoh - `name: string`

```
{
  age: string
  ...
```

```
}
```

Daftar string

- Membutuhkan - Opsi untuk menjadi tipe`boolean`.
- Penggunaan - Hasilkan kotak centang.
- Contoh - `isProduction: boolean`

```
{
  isProduction: boolean
  ...
}
```

Radio

- Membutuhkan - Opsi untuk menjadi penyatuan tiga atau lebih sedikit string.
- Penggunaan - Hasilkan radio yang dipilih.

> **Note**
>
> Ketika ada empat atau lebih item, jenis ini dirender sebagai dropdown.

- Contoh - `color: 'red' | 'blue' | 'green'`

```
{
  color: 'red' | 'blue' | 'green'
  ...
}
```

Dropdown

- Membutuhkan - Opsi untuk menjadi penyatuan empat atau lebih string.
- Penggunaan - Menghasilkan dropdown.
- Contoh - `runtimes: 'nodejs' | 'python' | 'java' | 'dotnetcore' | 'ruby'`

```
{
```

```
  runtimes: 'nodejs' | 'python' | 'java' | 'dotnetcore' | 'ruby'
  ...
}
```

Bagian yang dapat diperluas

- Membutuhkan - Opsi untuk menjadi objek.
- Penggunaan - Hasilkan bagian yang dapat diperluas. Opsi dalam objek akan bersarang di dalam bagian yang dapat diperluas di wizard.
- Contoh -

```
{
     expandableSectionTitle: {
         nestedString: string;
         nestedNumber: number;
     }
}
```

Tupel

- Membutuhkan - Opsi untuk menjadi tipe`Tuple`.
- Penggunaan - Hasilkan input berbayar nilai-kunci.
- Contoh - `tuple: Tuple[string, string]>`

```
{
    tuple: Tuple[string, string]>;
    ...
}
```

Daftar Tuple

- Membutuhkan - Opsi untuk menjadi array tipe`Tuple`.
- Penggunaan - Menghasilkan masukan daftar Tuple.
- Contoh - `tupleList: Tuple[string, string]>[]`

```
{
```

```
  tupleList: Tuple[string, string]>[];
  ...
}
```

Pemilih

- Membutuhkan - Opsi untuk menjadi tipe`Selector`.
- Penggunaan - Menghasilkan dropdown repositori sumber atau cetak biru yang diterapkan pada proyek.
- Contoh - `sourceRepo: Selector<SourceRepository>`

```
{
    sourceRepo: Selector<SourceRepository>;
    sourceRepoOrAdd: Selector<SourceRepository | string>;
    blueprintInstantiation: Selector<BlueprintInstantiation>;
  ...
}
```

Multipilih

- Membutuhkan - Opsi untuk menjadi tipe`Selector`.
- Penggunaan - Menghasilkan input multiselect.
- Contoh - `multiselect: MultiSelect['A' | 'B' | 'C' | 'D' | 'E']>`

```
{
  multiselect: MultiSelect['A' | 'B' | 'C' | 'D' | 'E']>;
  ...
}
```

Berkomunikasi dengan pengguna selama sintesis

Sebagai penulis cetak biru, Anda dapat berkomunikasi kembali ke pengguna di luar hanya pesan validasi. Misalnya, anggota ruang mungkin melihat kombinasi opsi yang menghasilkan cetak biru yang tidak jelas. Cetak biru khusus mendukung kemampuan untuk mengkomunikasikan pesan kesalahan kembali ke pengguna dengan menjalankan sintesis. Cetak biru dasar mengimplementasikan `throwSynthesisError(...)` fungsi yang mengharapkan pesan kesalahan yang jelas. Anda dapat memanggil pesan dengan menggunakan yang berikut ini:

```
//blueprint.ts
this.throwSynthesisError({
   name: BlueprintSynthesisErrorTypes.BlueprintSynthesisError,
   message: 'hello from the blueprint! This is a custom error communicated to the
 user.'
})
```

Menambahkan komponen lingkungan ke cetak biru

Wizard cetak biru khusus dihasilkan secara dinamis dari `Options` antarmuka yang diekspos melalui wizard. Cetak biru mendukung pembuatan komponen antarmuka pengguna (UI) dari tipe yang terbuka.

Untuk mengimpor komponen CodeCatalyst lingkungan cetak biru Amazon

Dalam `blueprint.ts` file Anda, tambahkan yang berikut ini:

```
import {...} from '@amazon-codecatalyst/codecatalyst-environments'
```

Topik

- Menciptakan lingkungan pengembangan
- Contoh antarmuka tiruan

Menciptakan lingkungan pengembangan

Contoh berikut menunjukkan cara menerapkan aplikasi Anda ke cloud:

```
export interface Options extends ParentOptions {
        ...
        myNewEnvironment:  EnvironmentDefinition{
            thisIsMyFirstAccountConnection: AccountConnection{
                thisIsARole: Role['lambda', 's3', 'dynamo'];
             };
        };
    }
```

Antarmuka menghasilkan komponen UI yang meminta lingkungan baru (`myNewEnvironment`) dengan koneksi akun tunggal (`thisIsMyFirstAccountConnection`. Peran pada koneksi akun (`thisIsARole`) juga dihasilkan dengan `['lambda', 's3', 'dynamo']` kemampuan peran minimum yang diperlukan. Tidak semua pengguna memiliki koneksi akun, jadi Anda harus

memeriksa kasus di mana pengguna tidak menghubungkan akun atau tidak menghubungkan akun dengan peran. Peran juga dapat dijelaskan dengan. `@inlinePolicies` Untuk informasi selengkapnya, lihat @inlinePolicy. /jalur/ke/policy/file.json.

Komponen lingkungan membutuhkan `name` dan`environmentType`. Kode berikut adalah bentuk default minimum yang diperlukan:

```
{
  ...
  "myNewEnvironment": {
    "name": "myProductionEnvironment",
    "environmentType": "PRODUCTION"
  },
}
```

Komponen UI kemudian meminta Anda untuk berbagai bidang. Saat Anda mengisi bidang, cetak biru mendapatkan bentuk yang sepenuhnya diperluas. Ini dapat membantu bagi Anda untuk memasukkan tiruan penuh dalam `defaults.json` file untuk tujuan pengujian dan pengembangan.

Contoh antarmuka tiruan

Antarmuka tiruan sederhana

```
{
    ...
    "thisIsMyEnvironment": {
        "name": "myProductionEnvironment",
        "environmentType": "PRODUCTION",
        "thisIsMySecondAccountConnection": {
            "id": "12345678910",
            "name": "my-account-connection-name",
            "secondAdminRole": {
                "arn": "arn:aws:iam::12345678910:role/ConnectedQuokkaRole",
                "name": "ConnectedQuokkaRole",
                "capabilities": [
                    "lambda",
                    "s3",
                    "dynamo"
                ]
            }
        }
    }
```

```
}
```

Antarmuka tiruan yang kompleks

```
export interface Options extends ParentOptions {
  /**
   * The name of an environment
   * @displayName This is a Environment Name
   * @collapsed
   */
  thisIsMyEnvironment: EnvironmentDefinition{
    /**
     * comments about the account that is being deployed into
     * @displayName This account connection has an overriden name
     * @collapsed
     */
    thisIsMyFirstAccountConnection: AccountConnection{
      /**
       * Blah blah some information about the role that I expect
       * e.g. here's a copy-pastable policy: [to a link]
       * @displayName This role has an overriden name
       */
      adminRole: Role['admin', 'lambda', 's3', 'cloudfront'];
      /**
       * Blah blah some information about the second role that I expect
       * e.g. here's a copy-pastable policy: [to a link]
       */
      lambdaRole: Role['lambda', 's3'];
    };
    /**
     * comments about the account that is being deployed into
     */
    thisIsMySecondAccountConnection: AccountConnection{
      /**
         * Blah blah some information about the role that I expect
         * e.g. here's a copy-pastable policy: [to a link]
         */
      secondAdminRole: Role['admin', 'lambda', 's3', 'cloudfront'];
      /**
         * Blah blah some information about the second role that I expect
         * e.g. here's a copy-pastable policy: [to a link]
         */
      secondLambdaRole: Role['lambda', 's3'];
```

```
    };
  };
}
```

Antarmuka tiruan lengkap

```
{
  ...
  "thisIsMyEnvironment": {
    "name": "my-production-environment",
    "environmentType": "PRODUCTION",
    "thisIsMySecondAccountConnection": {
      "id": "12345678910",
      "name": "my-connected-account",
      "secondAdminRole": {
        "name": "LambdaQuokkaRole",
        "arn": "arn:aws:iam::12345678910:role/LambdaQuokkaRole",
        "capabilities": [
          "admin",
          "lambda",
          "s3",
          "cloudfront"
        ]
      },
      "secondLambdaRole": {
        "name": "LambdaQuokkaRole",
        "arn": "arn:aws:iam::12345678910:role/LambdaQuokkaRole",
        "capabilities": [
          "lambda",
          "s3"
        ]
      }
    },
    "thisIsMyFirstAccountConnection": {
      "id": "12345678910",
      "name": "my-connected-account",
      "adminRole": {
        "name": "LambdaQuokkaRole",
        "arn": "arn:aws:iam::12345678910:role/LambdaQuokkaRole",
        "capabilities": [
          "admin",
          "lambda",
          "s3",
```

```
          "cloudfront"
        ]
      },
      "lambdaRole": {
        "name": "LambdaQuokkaRole",
        "arn": "arn:aws:iam::12345678910:role/LambdaQuokkaRole",
        "capabilities": [
          "lambda",
          "s3"
        ]
      }
    }
  },
}
```

Menambahkan komponen rahasia ke cetak biru

Rahasia dapat digunakan CodeCatalyst untuk menyimpan data sensitif yang dapat direferensikan dalam alur kerja. Anda dapat menambahkan rahasia ke cetak biru kustom Anda dan mereferensikannya di alur kerja Anda. Untuk informasi selengkapnya, lihat Mengkonfigurasi dan menggunakan rahasia dalam alur kerja.

Untuk mengimpor jenis CodeCatalyst wilayah cetak biru Amazon

Dalam `blueprint.ts` file Anda, tambahkan yang berikut ini:

```
import { Secret, SecretDefinition } from '@amazon-codecatalyst/blueprint-
component.secrets'
```

Topik

- Membuat rahasia
- Mereferensikan rahasia dalam alur kerja

Membuat rahasia

Contoh berikut membuat komponen UI yang meminta pengguna untuk memasukkan nilai rahasia dan deskripsi opsional:

```
export interface Options extends ParentOptions {
```

```
    ...
    mySecret: SecretDefinition;
}


export class Blueprint extends ParentBlueprint {
  constructor(options_: Options) {
    new Secret(this, options.secret);
}
```

Komponen rahasia membutuhkan`name`. Kode berikut adalah bentuk default minimum yang diperlukan:

```
{
    ...
    "secret": {
        "name": "secretName"
    },

}
```

Mereferensikan rahasia dalam alur kerja

Contoh cetak biru berikut menciptakan rahasia dan alur kerja yang mereferensikan nilai rahasia. Untuk informasi selengkapnya, lihat Mereferensikan rahasia dalam alur kerja.

```
export interface Options extends ParentOptions {
    ...
/**
*
* @validationRegex /^\w+$/
*/
  username: string;


  password: SecretDefinition;
}


export class Blueprint extends ParentBlueprint {
  constructor(options_: Options) {
    const password = new Secret(this, options_.password);
```

```
    const workflowBuilder = new WorkflowBuilder(this, {
      Name: 'my_workflow',
    });


    workflowBuilder.addBuildAction({
      actionName: 'download_files',
      input: {
        Sources: ['WorkflowSource'],
      },
      output: {
        Artifacts: [{ Name: 'download', Files: ['file1'] }],
      },
      steps: [
        `curl -u ${options_.username}:${password.reference} https://example.com`,
      ],
    });

    new Workflow(
      this,
      repo,
      workflowBuilder.getDefinition(),
    );

}
```

Untuk mempelajari lebih lanjut tentang menggunakan rahasia di CodeCatalyst, lihatMengkonfigurasi dan menggunakan rahasia dalam alur kerja.

Menambahkan komponen wilayah ke cetak biru

Jenis wilayah dapat ditambahkan ke `Options` antarmuka cetak biru khusus Anda untuk menghasilkan komponen dalam panduan cetak biru, Anda dapat memasukkan satu atau beberapa gion AWS. Jenis gion dapat diimpor dari cetak biru dasar Anda di file Anda. `blueprint.ts` Untuk informasi selengkapnya, lihat wilayah AWS.

Untuk mengimpor jenis CodeCatalyst wilayah cetak biru Amazon

Dalam `blueprint.ts` file Anda, tambahkan yang berikut ini:

```
import { Region } from '@amazon-codecatalyst/blueprints.blueprint'
```

Parameter tipe wilayah adalah larik kode wilayah AWS yang dapat dipilih, atau Anda dapat menggunakannya * untuk menyertakan semua wilayah AWS yang didukung.

Topik

Anotasi

Tag JSDoc dapat ditambahkan ke setiap bidang di `Options` antarmuka untuk menyesuaikan bagaimana bidang muncul dan berperilaku di wizard. Untuk jenis wilayah, tag berikut didukung:

- `@displayName`Anotasi dapat digunakan untuk mengubah label bidang di wizard.

  Contoh: `@displayName AWS Region`
- `@placeholder`Anotasi dapat digunakan untuk mengubah placeholder komponen select/multiselect.

  Contoh: `@placeholder Choose AWS Region`

Contoh komponen wilayah

Memilih wilayah dari daftar tertentu

```
export interface Options extends ParentOptions {
    ...
  /**
   * @displayName Region
   */
  region: Region<['us-east-1', 'us-east-2', 'us-west-1', 'us-west-2']>;
}
```

Memilih satu atau beberapa wilayah dari daftar tertentu

```
export interface Options extends ParentOptions {
    ...
  /**
   * @displayName Regions
   */
  multiRegion: Region<['us-east-1', 'us-east-2', 'us-west-1', 'us-west-2']>[];
```

```
}
```

Memilih satu egion AWS

```
export interface Options extends ParentOptions {
    ...
  /**
   * @displayName Region
   */
  region: Region<['*']>;
}
```

Memilih satu atau beberapa wilayah dari daftar tertentu

```
export interface Options extends ParentOptions {
    ...
  /**
   * @displayName Regions
   */
  multiRegion: Region<['us-east-1', 'us-east-2', 'us-west-1', 'us-west-2']>[];
}
```

Menambahkan komponen repositori dan kode sumber ke cetak biru

Repositori digunakan oleh Amazon CodeCatalyst untuk menyimpan kode. Repositori mengambil nama sebagai input. Sebagian besar komponen disimpan dalam repositori, seperti file kode sumber, alur kerja, dan komponen lain seperti lingkungan pengembangan terkelola (MDE). Komponen repositori sumber juga mengekspor komponen yang digunakan untuk mengelola file dan aset statis. Repositori memiliki batasan nama. Untuk informasi selengkapnya, lihat Simpan dan berkolaborasi pada kode dengan repositori sumber di CodeCatalyst.

```
const repository = new SourceRepository(this, {
  title: 'my-new-repository-title',
});
```

Untuk mengimpor repositori CodeCatalyst cetak biru Amazon dan komponen kode sumber

Dalam `blueprint.ts` file Anda, tambahkan yang berikut ini:

```
import {...} from '@caws-blueprint-component/caws-source-repositories'
```

Topik

Menambahkan file

Anda dapat menulis file teks ke repositori dengan konstruksinya. `SourceFile` Operasi adalah salah satu kasus penggunaan yang paling umum dan mengambil repositori, jalur file, dan konten teks. Jika jalur file tidak ada dalam repositori, komponen akan membuat semua folder yang diperlukan.

```
new SourceFile(repository, `path/to/my/file/in/repo/file.txt`, 'my file contents');
```

> **Note**
>
> Jika Anda menulis dua file ke lokasi yang sama dalam repositori yang sama, implementasi terbaru menimpa yang sebelumnya. Anda dapat menggunakan fitur ini untuk melapisi kode yang dihasilkan, dan ini sangat berguna untuk memperluas kode yang mungkin dihasilkan oleh cetak biru khusus.

Menambahkan file generik

Anda dapat menulis bit arbitrer ke repositori Anda. Anda dapat membaca dari buffer dan menggunakan `File` konstruksinya.

```
new File(repository, `path/to/my/file/in/repo/file.img`, new Buffer(...));

new File(repository, `path/to/my/file/in/repo/new-img.img`, new StaticAsset('path/to/
image.png').content());
```

Menyalin file

Anda dapat memulai dengan kode yang dihasilkan dengan menyalin dan menempelkan kode awal dan kemudian menghasilkan lebih banyak kode di atas basis itu. Tempatkan kode di dalam `static-`

`assets` direktori, dan kemudian targetkan kode itu dengan `StaticAsset` konstruksinya. Jalur dalam hal ini selalu dimulai di root `static-assets` direktori.

```
const starterCode = new StaticAsset('path/to/file/file.txt')
const starterCodeText = new StaticAsset('path/to/file/file.txt').toString()
const starterCodeRawContent = new StaticAsset('path/to/image/hello.png').content()

const starterCodePath = new StaticAsset('path/to/image/hello.png').path()
// starterCodePath is equal to 'path/to/image/hello.png'
```

Sebuah subkelas dari `StaticAsset` adalah`SubstitutionAsset`. Fungsi subclass persis sama, tetapi sebagai gantinya Anda dapat menjalankan substitusi kumis di atas file sebagai gantinya. Ini dapat berguna untuk melakukan generasi copy-and-replace gaya.

Substitusi aset statis menggunakan mesin template kumis untuk merender file statis yang disematkan ke dalam repositori sumber yang dihasilkan. Aturan template kumis diterapkan selama rendering, yang berarti bahwa semua nilai dikodekan HTML secara default. Untuk merender HTML unescaped, gunakan sintaks triple mustache. `{{{name}}}` Untuk informasi lebih lanjut, lihat aturan template kumis.

> **Note**
>
> Menjalankan pengganti di atas file yang tidak dapat ditafsirkan teks dapat menghasilkan kesalahan.

```
const starterCodeText = new SubstitionAsset('path/to/file/file.txt').subsitite({
  'my_variable': 'subbed value1',
  'another_variable': 'subbed value2'
})
```

Menargetkan banyak file

Aset statis mendukung penargetan glob melalui fungsi statis `StaticAsset` aktif dan subkelasnya yang dipanggil`findAll(...)`, yang mengembalikan daftar aset statis yang dimuat sebelumnya dengan jalur, konten, dan lainnya. Anda dapat menghubungkan daftar dengan `File` konstruksi untuk menyalin dan menempelkan konten di `static-assets` direktori.

```
new File(repository, `path/to/my/file/in/repo/file.img`, new Buffer(...));
```

```
new File(repository, `path/to/my/file/in/repo/new-img.img`, new StaticAsset('path/to/
image.png').content());
```

Membuat repositori baru dan menambahkan file

Anda dapat menggunakan komponen repositori untuk membuat repositori baru dalam proyek yang dihasilkan. Anda kemudian dapat menambahkan file atau alur kerja ke repositori yang dibuat.

```
import { SourceRepository } from '@amazon-codecatalyst/codecatalyst-source-
repositories';
...
const repository = new SourceRepository(this, { title: 'myRepo' });
```

Contoh berikut menunjukkan cara menambahkan file dan alur kerja ke repositori yang ada:

```
import { SourceFile } from '@amazon-codecatalyst/codecatalyst-source-repositories';
import { Workflow } from '@amazon-codecatalyst/codecatalyst-workflows';
...
new SourceFile(repository, 'README.md', 'This is the content of my readme');
new Workflow(this, repository, {/**...workflowDefinition...**/});
```

Menggabungkan dua potongan kode menghasilkan satu repositori bernama `myRepo` dengan file sumber `README.md` dan CodeCatalyst alur kerja di root.

Menambahkan komponen alur kerja ke cetak biru

Alur kerja digunakan oleh CodeCatalyst proyek Amazon untuk menjalankan tindakan berdasarkan pemicu. Anda dapat menggunakan komponen alur kerja untuk membangun dan menyusun file YAMAL alur kerja. Untuk informasi selengkapnya, lihat Alur kerja definisi YAMAL.

Untuk mengimpor komponen alur CodeCatalyst kerja cetak biru Amazon

Dalam `blueprint.ts` file Anda, tambahkan yang berikut ini:

```
import { WorkflowBuilder, Workflow } from '@amazon-codecatalyst/codecatalyst-workflows'
```

Topik

- Contoh komponen alur kerja

- Menghubungkan ke lingkungan

Contoh komponen alur kerja

WorkflowBuilder komponen

Anda dapat menggunakan kelas untuk membangun definisi alur kerja. Definisi dapat diberikan ke komponen alur kerja untuk rendering dalam repositori.

```
import { WorkflowBuilder } from '@amazon-codecatalyst/codecatalyst-workflows'

const workflowBuilder = new WorkflowBuilder({} as Blueprint, {
  Name: 'my_workflow',
});

// trigger the workflow on pushes to branch 'main'
workflowBuilder.addBranchTrigger(['main']);

// add a build action
workflowBuilder.addBuildAction({
  // give the action a name
  actionName: 'build_and_do_some_other_stuff',

  // the action pulls from source code
  input: {
    Sources: ['WorkflowSource'],
  },

  // the output attempts to autodiscover test reports, but not in the node modules
  output: {
    AutoDiscoverReports: {
      Enabled: true,
      ReportNamePrefix: AutoDiscovered,
      IncludePaths: ['**/*'],
      ExcludePaths: ['*/node_modules/**/*'],
    },
  },
  // execute some arbitrary steps
  steps: [
    'npm install',
    'npm run myscript',
    'echo hello-world',
  ],
```

```
  // add an account connection to the workflow
  environment: convertToWorkflowEnvironment(myEnv),
});
```

Komponen Proyek Alur Kerja

Contoh berikut menunjukkan bagaimana komponen Projen dapat digunakan untuk menulis alur kerja YAMAL ke repositori:

```
import { Workflow } from '@amazon-codecatalyst/codecatalyst-workflows'

...

const repo = new SourceRepository
const blueprint = this;
const workflowDef = workflowBuilder.getDefinition()

// creates a workflow.yaml at .aws/workflows/${workflowDef.name}.yaml
new Workflow(blueprint, repo, workflowDef);

// can also pass in any object and have it rendered as a yaml. This is unsafe and may
 not produce a valid workflow
new Workflow(blueprint, repo, {... some object ...});
```

Menghubungkan ke lingkungan

Banyak alur kerja perlu dijalankan dalam koneksi akun AWS. Alur kerja menangani ini dengan mengizinkan tindakan untuk terhubung ke lingkungan dengan spesifikasi akun dan nama peran.

```
import { convertToWorkflowEnvironment } from '@amazon-codecatalyst/codecatalyst-
workflows'


const myEnv = new Environment(...);

// can be passed into a workflow constructor
const workflowEnvironment = convertToWorkflowEnvironment(myEnv);


// add a build action
workflowBuilder.addBuildAction({
  ...
```

```
  // add an account connection to the workflow
  environment: convertToWorkflowEnvironment(myEnv),
});
```

Menambahkan komponen Dev Environments ke cetak biru

Lingkungan pengembangan terkelola (MDE) digunakan untuk membuat dan mempertahankan Ruang Kerja MDE di. CodeCatalyst Komponen menghasilkan `devfile.yaml` file. Untuk informasi lebih lanjut, lihat Pengantar Devfile danMengedit devfile repositori untuk Lingkungan Dev.

```
new Workspace(this, repository, SampleWorkspaces.default);
```

Untuk mengimpor komponen ruang CodeCatalyst kerja cetak biru Amazon

Dalam `blueprint.ts` file Anda, tambahkan yang berikut ini:

```
import {...} from '@amazon-codecatalyst/codecatalyst-workspaces'
```

Menambahkan komponen masalah ke cetak biru

Di CodeCatalyst, Anda dapat memantau fitur, tugas, bug, dan pekerjaan lain yang terlibat dalam proyek Anda. Setiap karya disimpan dalam catatan yang berbeda yang disebut masalah. Setiap masalah dapat memiliki deskripsi, penerima tugas, status, dan properti lainnya, yang dapat Anda cari, kelompokkan, dan filter. Anda dapat melihat masalah menggunakan tampilan default, atau Anda dapat membuat tampilan sendiri dengan pemfilteran, pengurutan, atau pengelompokan kustom. Untuk informasi lebih lanjut tentang konsep yang terkait dengan masalah, lihat Isu konsep danKuota untuk masalah di CodeCatalyst.

Komponen masalah menghasilkan representasi JSON dari suatu masalah. Komponen mengambil bidang ID dan definisi masalah sebagai input.

Untuk mengimpor komponen CodeCatalyst cetak biru Amazon

Dalam `blueprint.ts` file Anda, tambahkan yang berikut ini:

```
import {...} from '@amazon-codecatalyst/blueprint-component.issues'
```

Topik

- Contoh komponen masalah

Contoh komponen masalah

Membuat masalah

```
import { Issue } from '@amazon-codecatalyst/blueprint-component.issues';
...
new Issue(this, 'myFirstIssue', {
  title: 'myFirstIssue',
  content: 'This is an example issue.',
});
```

Membuat masalah prioritas tinggi

```
import { Workflow } from '@amazon-codecatalyst/codecatalyst-workflows'
...
const repo = new SourceRepository
const blueprint = this;
const workflowDef = workflowBuilder.getDefinition()

// Creates a workflow.yaml at .aws/workflows/${workflowDef.name}.yaml
new Workflow(blueprint, repo, workflowDef);

// Can also pass in any object and have it rendered as a yaml. This is unsafe and may
 not produce a valid workflow
new Workflow(blueprint, repo, {... some object ...});
```

Membuat masalah prioritas rendah dengan label

```
import { Issue } from '@amazon-codecatalyst/blueprint-component.issues';
...
new Issue(this, 'myThirdIssue', {
  title: 'myThirdIssue',
  content: 'This is an example of a low priority issue with a label.',
  priority: 'LOW',
  labels: ['exampleLabel'],
});
```

Bekerja dengan perkakas cetak biru dan CLI

CLI cetak biru menyediakan perkakas untuk mengelola dan bekerja dengan cetak biru khusus Anda.

Topik

Bekerja dengan perkakas cetak biru

Untuk bekerja dengan alat cetak biru

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Lanjutkan Lingkungan Pengembang Anda. Untuk informasi selengkapnya, lihat Melanjutkan Lingkungan Pengembang.

   Jika Anda tidak memiliki Lingkungan Pengembang, Anda harus membuatnya terlebih dahulu. Untuk informasi selengkapnya, lihat Membuat Lingkungan Dev.
3. Di terminal yang berfungsi, jalankan perintah berikut untuk menginstal CLI cetak biru:

   ```
   npm install -g @amazon-codecatalyst/blueprint-util.cli
   ```
4. Dalam `blueprint.ts` file, impor alat yang ingin Anda gunakan dalam format berikut:

   ```
   import { <tooling-function-name> } from '@amazon-codecatalyst/blueprint-util.cli/
   lib/<tooling-folder-name>/<tooling-file-name>;
   ```

   > **Tip**
   >
   > Anda dapat ke `CodeCatalyst blueprints GitHub repository`untuk menemukan nama perkakas yang ingin Anda gunakan.

   Jika Anda ingin menggunakan alat pengunggahan gambar, tambahkan yang berikut ini ke skrip Anda:

   ```
   import { uploadImagePublicly } from '@amazon-codecatalyst/blueprint-util.cli/lib/
   image-upload-tool/upload-image-to-aws';
   ```

   Contoh

   - Jika Anda ingin menggunakan fungsi penerbitan, tambahkan yang berikut ini ke skrip Anda:

```
import { publish } from '@amazon-codecatalyst/blueprint-util.cli/lib/publish/
publish';
```

- Jika Anda ingin menggunakan alat pengunggahan gambar, tambahkan yang berikut ini ke skrip Anda:

```
import { uploadImagePublicly } from '@amazon-codecatalyst/blueprint-util.cli/lib/
image-upload-tool/upload-image-to-aws';
```

5. Panggil fungsinya.

   Contoh:

   - Jika Anda ingin menggunakan fungsi penerbitan, tambahkan yang berikut ini ke skrip Anda:

```
await publish(logger, config.publishEndpoint, {<your publishing options>});
```

   - Jika Anda ingin menggunakan alat pengunggahan gambar, tambahkan yang berikut ini ke skrip Anda:

```
const {imageUrl, imageName} = await uploadImagePublicly(logger, 'path/to/
image'));
```

Alat unggah gambar

Alat unggah gambar memberi Anda kemampuan untuk mengunggah gambar Anda sendiri ke bucket S3 di akun AWS Anda dan kemudian mendistribusikan gambar itu secara publik di belakang. CloudFront Alat ini mengambil jalur gambar di penyimpanan lokal (dan nama bucket opsional) sebagai input, dan mengembalikan URL ke gambar yang tersedia untuk umum. Untuk informasi selengkapnya, lihat Apa itu Amazon CloudFront? dan Apa itu Amazon S3?

Untuk bekerja dengan alat unggah gambar

1. Kloning GitHub repositori cetak biru sumber terbuka yang menyediakan akses ke SDK cetak biru dan cetak biru sampel. Di terminal yang berfungsi, jalankan perintah berikut:

```
git clone https://github.com/aws/codecatalyst-blueprints.git
```

2. Jalankan perintah berikut untuk menavigasi ke repositori cetak biru GitHub :

```
cd codecatalyst-blueprints
```

3. Jalankan perintah berikut untuk menginstal dependensi:

```
yarn && yarn build
```

4. Jalankan perintah berikut untuk memastikan versi CLI cetak biru terbaru diinstal:

```
yarn upgrade @amazon-codecatalyst/blueprint-util.cli
```

5. Masuk ke akun AWS dengan bucket S3 tempat Anda ingin mengunggah gambar Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi AWS CLI, dan Masuk melalui AWS Command Line Interface.
6. Jalankan perintah berikut dari root CodeCatalyst repositori Anda untuk menavigasi ke direktori dengan CLI cetak biru:

```
cd packages/utils/blueprint-cli
```

7. Jalankan perintah berikut untuk mengunggah gambar Anda ke bucket S3:

```
yarn blueprint upload-image-public <./path/to/your/image>
      <optional:optional-bucket-name>
```

URL ke gambar Anda dihasilkan. URL tidak akan segera tersedia karena memerlukan beberapa waktu agar CloudFront distribusi dapat diterapkan. Periksa status distribusi untuk mendapatkan status penerapan terbaru. Untuk informasi selengkapnya, lihat Bekerja dengan distribusi.

Menilai perubahan antarmuka dengan pengujian snapshot

Pengujian snapshot yang dihasilkan di beberapa konfigurasi cetak biru Anda didukung.

Blueprints mendukung pengujian snapshot pada konfigurasi yang disediakan oleh Anda sebagai penulis cetak biru. Konfigurasi adalah penggantian sebagian yang digabungkan di atas file defaults.json di root cetak biru. Saat pengujian snapshot diaktifkan dan dikonfigurasi, proses pembuatan dan pengujian mensintesis konfigurasi yang diberikan dan memverifikasi bahwa output yang disintesis tidak berubah dari snapshot referensi. Untuk melihat kode pengujian snapshot, lihat repositori CodeCatalyst cetak biru GitHub .

Untuk mengaktifkan pengujian snapshot

1. Dalam `.projenrc.ts` file, perbarui objek input ProjenBlueprint dengan file yang ingin Anda snapshot. Sebagai contoh:

```
{
  ....
  blueprintSnapshotConfiguration: {
    snapshotGlobs: ['**', '!environments/**', '!aws-account-to-environment/**'],
  },
}
```

2. Sintesis ulang cetak biru untuk membuat TypeScript file dalam proyek cetak biru Anda. Jangan mengedit file sumber karena mereka dipelihara dan dibuat ulang oleh Projen. Gunakan perintah berikut ini.

```
yarn projen
```

3. Arahkan ke `src/snapshot-configurations` direktori untuk melihat `default-config.json` file dengan objek kosong. Perbarui atau ganti file dengan satu atau lebih konfigurasi pengujian Anda sendiri. Setiap konfigurasi pengujian kemudian digabungkan dengan `defaults.json` file proyek, disintesis, dan dibandingkan dengan snapshot saat pengujian. Gunakan perintah berikut untuk menguji:

```
yarn test
```

   Pertama kali Anda menggunakan perintah uji, pesan berikut ditampilkan:`Snapshot Summary › NN snapshots written from 1 test suite`. Pengujian berikutnya akan memverifikasi bahwa output yang disintesis tidak berubah dari snapshot dan menampilkan pesan berikut: `Snapshots: NN passed, NN total`

   Jika Anda sengaja mengubah cetak biru Anda untuk menghasilkan output yang berbeda, maka jalankan perintah berikut untuk memperbarui snapshot referensi:

```
yarn test:update
```

Snapshot mengharapkan output yang disintesis konstan di antara setiap proses. Jika cetak biru Anda menghasilkan file yang bervariasi, Anda harus mengecualikan file tersebut dari pengujian snapshot.

Perbarui `blueprintSnapshotConfiguration` objek objek `ProjenBluerpint` masukan Anda untuk menambahkan `snapshotGlobs` properti. `snapshotGlobs`Properti adalah array gumpalan yang menentukan file mana yang disertakan atau dikecualikan dari snapshotting.

> **Note**
>
> Ada daftar default gumpalan. Jika Anda menentukan daftar Anda sendiri, Anda mungkin perlu secara eksplisit mengembalikan entri default.

Menerbitkan cetak biru khusus ke spasi

Sebelum Anda dapat cetak biru khusus untuk katalog cetak biru ruang Anda, Anda harus mempublikasikannya ke ruang. Anda juga dapat melihat cetak biru di CodeCatalyst konsol sebelum menerbitkan. Anda dapat mempublikasikan versi pratinjau atau versi normal cetak biru Anda.

> **Important**
>
> Jika Anda ingin menggunakan paket cetak biru dari sumber eksternal, pertimbangkan risiko yang mungkin datang dengan paket tersebut. Anda bertanggung jawab atas cetak biru khusus yang Anda tambahkan ke ruang Anda dan kode yang dihasilkannya.

Topik

- Melihat dan menerbitkan versi pratinjau cetak biru kustom
- Melihat dan menerbitkan versi normal dari cetak biru kustom
- Menerbitkan dan menerapkan cetak biru khusus di ruang dan proyek tertentu

Melihat dan menerbitkan versi pratinjau cetak biru kustom

Anda dapat mempublikasikan versi pratinjau cetak biru kustom Anda ke ruang Anda jika Anda ingin menambahkannya ke katalog cetak biru ruang Anda. Ini memungkinkan Anda untuk melihat cetak biru sebagai pengguna sebelum menambahkan versi non-pratinjau ke katalog. Versi pratinjau memungkinkan Anda untuk mempublikasikan tanpa mengambil versi sebenarnya. Misalnya, jika Anda bekerja pada `0.0.1` versi, Anda dapat mempublikasikan dan menambahkan versi pratinjau, sehingga pembaruan baru untuk versi kedua dapat dipublikasikan dan ditambahkan sebagai`0.0.2`.

Setelah membuat perubahan, buat kembali paket cetak biru kustom Anda dengan menjalankan `package.json` file, dan pratinjau perubahan Anda.

Untuk melihat dan mempublikasikan versi pratinjau cetak biru kustom

1. Lanjutkan Lingkungan Pengembang Anda. Untuk informasi selengkapnya, lihat Melanjutkan Lingkungan Pengembang
2. Buka terminal yang berfungsi di Lingkungan Pengembang Anda.
3. (Opsional) Di terminal yang berfungsi, instal dependensi yang diperlukan untuk proyek Anda jika Anda belum menginstalnya. Gunakan perintah berikut ini.

   ```
   yarn
   ```

4. (Opsional) Jika Anda membuat perubahan pada `.projenrc.ts` file, buat ulang konfigurasi proyek Anda sebelum membuat dan melihat pratinjau cetak biru Anda. Gunakan perintah berikut ini.

   ```
   yarn projen
   ```

5. Bangun kembali dan pratinjau cetak biru kustom Anda menggunakan perintah berikut. Gunakan perintah followng:

   ```
   yarn blueprint:preview
   ```

   Arahkan ke `See this blueprint at:` tautan yang disediakan untuk melihat pratinjau cetak biru kustom Anda. Periksa apakah UI, termasuk teks, muncul seperti yang Anda harapkan berdasarkan konfigurasi Anda. Jika Anda ingin mengubah cetak biru kustom Anda, Anda dapat mengedit `blueprint.ts` file, mensintesis ulang cetak biru, dan kemudian mempublikasikan versi pratinjau lagi. Untuk informasi selengkapnya, lihat Resintesis.
6. (Opsional) Anda dapat mempublikasikan versi pratinjau cetak biru kustom Anda, yang kemudian dapat ditambahkan ke katalog cetak biru ruang Anda. Arahkan ke `Enable version` *`[preview version number]`* `at:` tautan untuk mempublikasikan versi pratinjau ke ruang Anda.

Anda dapat meniru pembuatan proyek tanpa harus membuat proyek di CodeCatalyst. Untuk mensintesis proyek Anda, gunakan perintah berikut:

```
yarn blueprint:synth
```

Cetak biru dihasilkan di folder. `synth/synth.[options-name]/proposed-bundle/` Untuk informasi selengkapnya, lihat Sintesis.

Jika Anda memperbarui cetak biru kustom, gunakan perintah berikut untuk mensintesis ulang proyek Anda:

```
yarn blueprint:resynth
```

Cetak biru dihasilkan di folder. `synth/synth.[options-name]/proposed-bundle/` Untuk informasi selengkapnya, lihat Resintesis.

Setelah mempublikasikan versi pratinjau Anda, Anda kemudian dapat menambahkan cetak biru sehingga anggota ruang dapat menggunakannya untuk membuat proyek baru atau menerapkan dalam proyek yang ada. Untuk informasi selengkapnya, lihat Menambahkan cetak biru khusus ke katalog ruang.

Melihat dan menerbitkan versi normal dari cetak biru kustom

Setelah Anda selesai mengembangkan dan melihat pratinjau cetak biru kustom Anda, Anda dapat melihat dan menerbitkan versi baru yang ingin Anda tambahkan ke katalog cetak biru ruang Anda. Alur kerja rilis yang dihasilkan saat membuat proyek secara otomatis menerbitkan perubahan yang didorong. Jika Anda mematikan pembuatan alur kerja saat membuat cetak biru, cetak biru Anda tidak tersedia secara otomatis untuk ditambahkan ke katalog cetak biru ruang Anda. Anda masih dapat mempublikasikan cetak biru kustom Anda ke ruang Anda setelah menjalankan perintah. `yarn`

Untuk melihat dan menerbitkan cetak biru kustom

1. Lanjutkan Lingkungan Pengembang Anda. Untuk informasi selengkapnya, lihat Melanjutkan Lingkungan Pengembang
2. Buka terminal yang berfungsi di Lingkungan Pengembang Anda.
3. • Jika Anda memilih keluar dari pembuatan alur kerja rilis saat membuat cetak biru, gunakan perintah berikut:

   ```
   yarn blueprint:release
   ```

   Anda masih dapat menavigasi ke `See this blueprint at:` tautan yang disediakan untuk melihat cetak biru kustom Anda.

Publikasikan versi terbaru dari cetak biru kustom Anda, yang kemudian dapat ditambahkan ke katalog cetak biru ruang Anda. Arahkan ke `Enable version` *`[release version number]`* `at:` tautan untuk mempublikasikan versi terbaru ke ruang Anda.

- Jika Anda memilih alur kerja rilis saat membuat cetak biru, versi cetak biru terbaru akan dipublikasikan secara otomatis saat perubahan didorong. Gunakan salah satu perintah berikut ini:

```
git add .
```

```
git commit -m "commit message"
```

```
git push
```

Setelah menerbitkan versi normal Anda, Anda kemudian dapat menambahkan cetak biru sehingga anggota ruang dapat menggunakannya untuk membuat proyek baru atau menerapkan dalam proyek yang ada. Untuk informasi selengkapnya, lihat Menambahkan cetak biru khusus ke katalog ruang.

Menerbitkan dan menerapkan cetak biru khusus di ruang dan proyek tertentu

Secara default, `blueprint:release` perintah `blueprint:preview` and diterbitkan ke CodeCatalyst ruang tempat Anda membuat cetak biru. Jika Anda memiliki beberapa ruang Enterprise, Anda dapat melihat pratinjau dan mempublikasikan cetak biru yang sama di ruang tersebut juga. Anda juga dapat menerapkan cetak biru ke proyek ruang lain yang ada.

Untuk memublikasikan atau menerapkan cetak biru kustom di ruang tertentu

1. Lanjutkan Lingkungan Pengembang Anda. Untuk informasi selengkapnya, lihat Melanjutkan Lingkungan Pengembang.
2. Buka terminal yang berfungsi di Lingkungan Pengembang Anda.
3. (Opsional) Instal dependensi yang diperlukan untuk proyek Anda jika Anda belum menginstalnya. Gunakan perintah berikut ini.

   ```
   yarn
   ```

4. Gunakan `--space` tag untuk mempublikasikan pratinjau atau versi normal ke ruang tertentu. Sebagai contoh:

- ```
  yarn blueprint:preview --space my-awesome-space # publishes under a "preview"
   version tag to 'my-awesome-space'
  ```

  Contoh output:

  ```
  Enable version 0.0.1-preview.0 at: https://codecatalyst.aws/spaces/my-awesome-
  space/blueprints
  Blueprint applied to [NEW]: https://codecatalyst.aws/spaces/my-awesome-space/
  blueprints/%40amazon-codecatalyst%2Fmyspace.my-blueprint/publishers/1524817d-
  a69b-4abe-89a0-0e4a9a6c53b2/versions/0.0.1-preview.0/projects/create
  ```

- ```
  yarn blueprint:release --space my-awesome-space # publishes normal version to
   'my-awesome-space'
  ```

  Contoh output:

  ```
  Enable version 0.0.1 at: https://codecatalyst.aws/spaces/my-awesome-space/
  blueprints
  Blueprint applied to [NEW]: https://codecatalyst.aws/spaces/my-awesome-space/
  blueprints/%40amazon-codecatalyst%2Fmyspace.my-blueprint/publishers/1524817d-
  a69b-4abe-89a0-0e4a9a6c53b2/versions/0.0.1/projects/create
  ```

Gunakan `--project` untuk menerapkan versi pratinjau cetak biru kustom ke proyek yang ada di ruang tertentu. Sebagai contoh:

```
yarn blueprint:preview --space my-awesome-space --project my-project # previews
 blueprint application to an existing project
```

Contoh output:

```
Enable version 0.0.1-preview.1 at: https://codecatalyst.aws/spaces/my-awesome-
space/blueprints
Blueprint applied to [my-project]: https://codecatalyst.aws/spaces/my-awesome-
space/projects/my-project/blueprints/%40amazon-codecatalyst%2FmySpace.my-blueprint/
publishers/1524817d-a69b-4abe-89a0-0e4a9a6c53b2/versions/0.0.1-preview.1/add
```

Melihat detail, versi, dan proyek cetak biru kustom

Anda dapat melihat cetak biru kustom yang dipublikasikan di ruang Anda, termasuk detail cetak biru, versi, dan proyek yang dibuat dengan atau menerapkan cetak biru.

Topik

Melihat cetak biru khusus ruang

Untuk melihat cetak biru kustom spasi

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, navigasikan ke ruang tempat Anda ingin melihat cetak biru khusus.
3. Di dasbor ruang, pilih tab Pengaturan, lalu pilih Blueprints untuk melihat cetak biru Space. Rincian berikut ditampilkan dalam tabel:
   - Nama - Nama cetak biru kustom.
   - Status katalog - Apakah cetak biru kustom dipublikasikan ke katalog cetak biru ruang.
   - Versi terbaru - Versi terbaik dari cetak biru kustom.
   - Terbaru dimodifikasi - Tanggal ketika cetak biru ruang terakhir diperbarui.

Melihat proyek yang dibuat dengan atau menerapkan cetak biru khusus

Untuk melihat proyek yang dibuat dengan atau menerapkan cetak biru kustom

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, navigasikan ke ruang tempat Anda ingin melihat cetak biru khusus.
3. Di dasbor ruang, pilih tab Pengaturan, lalu pilih Blueprints.
4. Dari tabel cetak biru Space, pilih nama cetak biru kustom untuk melihat Proyek menggunakan cetak biru dan Proyek yang tidak menggunakan tabel cetak biru.

Menambahkan cetak biru khusus ke katalog ruang

Setelah Anda menerbitkan cetak biru khusus ke ruang Anda, itu dapat ditambahkan ke katalog cetak biru ruang Anda. Jika Anda menambahkan cetak biru khusus ke katalog cetak biru CodeCatalyst ruang Anda, maka cetak biru tersedia untuk semua anggota ruang untuk digunakan saat membuat proyek atau menerapkannya ke proyek yang ada. Sebelum menambahkan cetak biru khusus ke katalog cetak biru ruang, izin penerbitan cetak biru harus diaktifkan. Jika Anda memilih untuk pembuatan rilis alur kerja, izin penerbitan diaktifkan secara default. Untuk informasi selengkapnya, lihat Menyetel izin penerbitan untuk cetak biru kustom dan Menerbitkan cetak biru khusus ke spasi.

Untuk menambahkan cetak biru ke katalog cetak biru ruang

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Cetak biru hanya dapat ditambahkan dari cabang default dari repositori sumber. Jika Anda mengembangkan cetak biru pada cabang fitur, gabungkan cabang fitur Anda dengan perubahan pada cabang default. Buat permintaan tarik untuk menggabungkan perubahan apa pun ke cabang default. Untuk informasi selengkapnya, lihat Meninjau kode dengan permintaan tarik di Amazon CodeCatalyst.
3. Di CodeCatalyst konsol, navigasikan ke dasbor luar angkasa dengan cetak biru khusus Anda.
4. Di dasbor ruang, pilih tab Pengaturan, lalu pilih Blueprints.
5. Pilih nama cetak biru yang ingin Anda tambahkan, lalu pilih Tambahkan ke katalog. Jika Anda memiliki lebih dari satu versi, pilih versi dari menu tarik-turun versi Katalog
6. Pilih Simpan.

Menghapus cetak biru khusus dari katalog ruang

Cetak biru khusus dapat dihapus dari katalog cetak biru ruang Anda jika Anda tidak lagi ingin digunakan untuk membuat proyek baru atau diterapkan ke proyek yang ada.

> **Note**
>
> Jika Anda menghapus cetak biru kustom dari katalog spasi, itu tidak memengaruhi proyek yang dibuat dari cetak biru atau proyek yang menerapkan cetak biru. Sumber daya cetak biru tidak dihapus dari proyek.

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.

2. Di CodeCatalyst konsol, arahkan ke dasbor luar angkasa dengan cetak biru khusus Anda
3. Di dasbor ruang, pilih tab Pengaturan, lalu pilih Blueprints.
4. Pilih nama cetak biru yang ingin Anda hapus, lalu pilih Hapus cetak biru dari katalog.

Menyetel izin penerbitan untuk cetak biru kustom

Secara default, izin cetak biru kustom diaktifkan jika rilis alur kerja dibuat selama pembuatan proyek. Saat izin penerbitan diaktifkan, cetak biru dapat dipublikasikan ke ruang. Anda dapat menonaktifkan izin sehingga cetak biru tidak dapat dipublikasikan. Saat izin dinonaktifkan, alur kerja rilis yang dihasilkan selama pembuatan cetak biru tidak berjalan. Perubahan baru pada cetak biru tidak dapat dipublikasikan kecuali izin cetak biru diaktifkan.

> **Important**
>
> Untuk mengaktifkan atau menonaktifkan izin penerbitan proyek cetak biru, Anda harus memiliki peran administrator Space.

Untuk menetapkan izin penerbitan proyek cetak biru

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, navigasikan ke ruang tempat Anda ingin mengelola izin penerbitan cetak biru kustom.
3. Di dasbor ruang, pilih tab Pengaturan, lalu pilih Blueprints.
4. Pilih tab Izin penerbitan proyek untuk melihat izin penerbitan untuk semua cetak biru ruang Anda.
5. Pilih cetak biru yang ingin Anda kelola, lalu pilih Aktifkan atau Nonaktifkan untuk mengubah izin penerbitan. Jika Anda mengaktifkan izin, tinjau detail perubahan izin, lalu pilih Aktifkan penerbitan cetak biru untuk mengonfirmasi perubahan.

Mengubah versi katalog untuk cetak biru kustom

Sebagai penulis cetak biru, Anda dapat mengelola versi yang ingin Anda terbitkan ke katalog cetak biru ruang. Mengubah versi katalog cetak biru tidak memengaruhi proyek yang menggunakan versi cetak biru yang berbeda.

Untuk mengelola versi cetak biru kustom

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, navigasikan ke ruang tempat Anda ingin mengubah versi cetak biru khusus.
3. Di dasbor ruang, pilih tab Pengaturan, lalu pilih Blueprints.
4. Pada tabel cetak biru Space, pilih tombol radio untuk cetak biru khusus yang ingin Anda kelola.
5. Pilih Buat versi katalog, lalu pilih versi a dari menu tarik-turun versi Katalog.
6. Pilih Simpan.

Menghapus cetak biru atau versi kustom yang diterbitkan

Saat Anda menghapus versi cetak biru khusus atau cetak biru itu sendiri dari ruang CodeCatalyst Amazon Anda, semua akses Anda akan dihapus ke sumber daya proyek cetak biru atau versi cetak biru. Ketika Anda telah menghapus versi cetak biru atau cetak biru, anggota proyek tidak akan dapat mengakses sumber daya proyek, dan alur kerja apa pun yang diminta oleh repositori sumber pihak ketiga akan dihentikan.

> **Note**
>
> Jika Anda menghapus cetak biru, itu tidak memengaruhi proyek yang menerapkan cetak biru. Sumber daya cetak biru tidak dihapus dari proyek.

Jika versi cetak biru dipublikasikan ke katalog cetak biru ruang, pilih versi baru untuk katalog sebelum Anda menghapus versi yang dipublikasikan.

Untuk menghapus versi katalog cetak biru kustom

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, navigasikan ke ruang tempat Anda ingin menghapus versi katalog cetak biru kustom.
3. Di dasbor ruang, pilih tab Pengaturan, lalu pilih Blueprints.
4. Pilih nama cetak biru dengan versi katalog yang ingin Anda hapus.
5. Pilih tombol radio untuk versi katalog yang ingin Anda hapus, lalu pilih Hapus versi.
6. Tinjau detailnya, lalu pilih versi cetak biru lain dari menu tarik-turun Pilih versi katalog cetak biru baru.

7. Masukkan `delete` untuk mengonfirmasi penghapusan versi katalog cetak biru.
8. Pilih Delete (Hapus).

Jika versi cetak biru tidak ada dalam katalog cetak biru ruang, Anda dapat menghapus versi tanpa memilih versi baru.

Untuk menghapus versi cetak biru kustom

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, navigasikan ke ruang tempat Anda ingin menghapus versi cetak biru khusus.
3. Di dasbor ruang, pilih tab Pengaturan, lalu pilih Blueprints.
4. Pilih nama cetak biru dengan versi yang ingin Anda hapus.
5. Pilih tombol radio untuk versi yang ingin Anda hapus, lalu pilih Hapus versi.
6. Masukkan `delete` untuk mengonfirmasi penghapusan versi cetak biru.
7. Pilih Delete (Hapus).

Menghapus cetak biru dari katalog cetak biru ruang akan menghapus semua versi cetak biru. Proyek ruang yang menggunakan cetak biru tidak terpengaruh oleh penghapusan.

Untuk menghapus versi cetak biru kustom

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di CodeCatalyst konsol, navigasikan ke ruang tempat Anda ingin menghapus cetak biru khusus.
3. Di dasbor ruang, pilih tab Pengaturan, lalu pilih Blueprints.
4. Pada tabel Blueprints Space, pilih tombol radio untuk cetak biru kustom yang ingin Anda hapus, lalu pilih Hapus cetak biru.
5. Masukkan `delete` untuk mengonfirmasi penghapusan cetak biru kustom.
6. Pilih Hapus.

Menangani dependensi, ketidakcocokan, dan perkakas

Topik

- Menambahkan dependensi

- Menangani ketidakcocokan jenis ketergantungan
- Menggunakan benang dan npm
- Memutakhirkan perkakas dan komponen

Menambahkan dependensi

Sebagai penulis cetak biru, Anda mungkin perlu menambahkan paket ke cetak biru Anda, seperti. `@amazon-codecatalyst/blueprint-component.environments` Anda perlu memperbarui `projen.ts` file dengan paket itu, dan kemudian membuat ulang konfigurasi proyek Anda dengan Projen. Projen bertindak sebagai model proyek untuk setiap basis kode cetak biru, yang menyediakan kemampuan untuk mendorong pembaruan perkakas yang kompatibel dengan mundur dengan mengubah cara model merender file konfigurasi. `package.json`File tersebut adalah file yang sebagian dimiliki oleh model Projen. Projen mengakui versi ketergantungan yang disertakan dalam file package.json, tetapi opsi lain harus berasal dari model.

Untuk menambahkan dependensi dan memperbarui file **`projenrc.ts`**

1. Dalam `projen.ts` file, navigasikan ke bagian deps.
2. Tambahkan ketergantungan yang ingin Anda gunakan dalam cetak biru Anda.
3. Gunakan perintah berikut untuk membuat ulang konfigurasi proyek Anda:

```
yarn projen && yarn
```

Menangani ketidakcocokan jenis ketergantungan

Setelah pembaruan Yarn, Anda mungkin mendapatkan kesalahan berikut terkait parameter repositori:

```
Type 'SourceRepository' is missing the following properties from type
 'SourceRepository': synthesisSteps, addSynthesisStep
```

Kesalahan ini disebabkan ketidakcocokan ketergantungan yang terjadi ketika satu komponen bergantung pada versi yang lebih baru dari komponen lain, tetapi komponen yang mengandalkan disematkan ke versi yang lebih lama. Kesalahan dapat diperbaiki dengan membuat semua komponen Anda bergantung pada versi yang sama sehingga versi disinkronkan di antara mereka. Yang terbaik adalah menyimpan semua paket yang dijual cetak biru di bawah versi terbaru (`0.0.x`) yang sama, kecuali Anda yakin tentang bagaimana Anda menangani versi tersebut. Contoh berikut

menunjukkan bagaimana `package.json` file dapat dikonfigurasi sehingga semua dependensi bergantung pada versi yang sama:

```
...
"@caws-blueprint-component/caws-environments": "^0.1.12345",
"@caws-blueprint-component/caws-source-repositories": "^0.1.12345",
"@caws-blueprint-component/caws-workflows": "^0.1.12345",
"@caws-blueprint-component/caws-workspaces": "^0.1.12345",
"@caws-blueprint-util/blueprint-utils": "^0.1.12345",
...
"@caws-blueprint/blueprints.blueprint": "*",
```

Setelah mengonfigurasi versi untuk semua dependensi, gunakan perintah berikut:

```
yarn install
```

Menggunakan benang dan npm

Cetak biru menggunakan Benang untuk perkakas. Menggunakan npm dan Yarn akan menyebabkan masalah perkakas karena cara pohon ketergantungan diselesaikan oleh masing-masing berbeda. Untuk menghindari masalah seperti itu, yang terbaik adalah menggunakan Benang saja.

Jika Anda tidak sengaja menginstal dependensi menggunakan npm, Anda dapat menghapus file yang dihasilkan, dan memastikan `package-lock.json` `.projenrc.ts` file Anda diperbarui dengan dependensi yang Anda butuhkan. Anda membuat ulang konfigurasi proyek Anda dengan Projen.

Gunakan yang berikut ini untuk regenerasi dari model:

```
yarn projen
```

Setelah memastikan file.projenrc.ts Anda diperbarui dengan dependensi yang diperlukan, gunakan perintah berikut:

```
yarn
```

Memutakhirkan perkakas dan komponen

Terkadang, Anda mungkin ingin memutakhirkan perkakas dan komponen Anda untuk menghadirkan fitur baru yang tersedia. Anda disarankan untuk menyimpan semua komponen pada versi yang sama

kecuali Anda yakin tentang bagaimana Anda menangani versi. Versi disinkronkan antar komponen, sehingga versi yang sama untuk semua komponen memastikan ketergantungan yang tepat di antara mereka.

Menggunakan Yarn workspace monorepo

Gunakan perintah berikut untuk memutakhirkan utilitas dan komponen dari root repositori cetak biru kustom:

```
yarn upgrade @amazon-codecatalyst/*
```

Gunakan perintah berikut jika Anda tidak menggunakan monorepo:

```
yarn upgrade —pattern @amazon-codecatalyst/*
```

Opsi lain yang dapat Anda gunakan untuk meningkatkan perkakas dan komponen:

- Gunakan npm view `@caws-blueprint-component/<some-component>` untuk mendapatkan versi terbaru.
- Tingkatkan ke versi terbaru secara manual dengan menyetel versi di file package.json Anda dan menggunakan perintah berikut:. `yarn` Semua komponen dan utilitas harus memiliki versi yang sama.

Berkontribusi

Kit pengembangan perangkat lunak cetak biru (SDK) adalah pustaka sumber terbuka yang dapat Anda sumbangkan. Sebagai kontributor, pertimbangkan pedoman kontribusi, umpan balik, dan cacat. Untuk informasi selengkapnya, lihat repositori cetak biru GitHub .

Kuota untuk cetak biru di CodeCatalyst

Tabel berikut menjelaskan kuota dan batas untuk cetak biru di Amazon. CodeCatalyst Untuk informasi selengkapnya tentang kuota di Amazon CodeCatalyst, lihatKuota untuk CodeCatalyst.

Jumlah maksimum cetak biru yang diterapkan per proyek CodeCatalyst	100

Simpan dan berkolaborasi pada kode dengan repositori sumber di CodeCatalyst

CodeCatalyst repositori sumber adalah repositori Git yang dihosting di Amazon. CodeCatalyst Anda dapat menggunakan repositori sumber CodeCatalyst untuk menyimpan, versi, dan mengelola aset untuk proyek dengan aman.

Aset dalam CodeCatalyst repositori dapat mencakup:

- dokumen
- kode sumber
- file biner

CodeCatalyst juga menggunakan repositori sumber untuk proyek untuk menyimpan informasi konfigurasi untuk proyek Anda, seperti file konfigurasi alur kerja.

Anda dapat memiliki lebih dari satu repositori sumber dalam sebuah CodeCatalyst proyek. Misalnya, Anda mungkin ingin memiliki repositori sumber terpisah untuk kode sumber front-end, kode sumber back-end, utilitas, dan dokumentasi.

Berikut adalah salah satu alur kerja yang mungkin untuk bekerja dengan kode di repositori sumber, permintaan tarik, dan Lingkungan Pengembang di: CodeCatalyst

Mary Major membuat proyek aplikasi web dalam CodeCatalyst menggunakan cetak biru, yang menciptakan repositori sumber dengan kode sampel di dalamnya. Dia mengundang teman-temannya Li Juan, Saanvi Sarkar, dan Jorge Souza untuk mengerjakan proyek bersamanya. Li Juan melihat kode sampel di repositori sumber dan memutuskan untuk membuat beberapa perubahan cepat untuk menambahkan tes ke kode. *Li menciptakan Lingkungan Dev, memilih AWS Cloud9 sebagai IDE, dan menentukan cabang baru, test-code.* Lingkungan Dev terbuka. Li dengan cepat menambahkan kode, lalu melakukan dan mendorong cabang dengan perubahan pada repositori sumber masuk. CodeCatalyst Li kemudian membuat permintaan tarik. Sebagai bagian dari pembuatan permintaan tarik itu, Li menambahkan Jorge Souza dan Saanvi Sarkar sebagai pengulas untuk memastikan bahwa kode tersebut ditinjau.

Saat meninjau kode, Jorge Souza ingat bahwa ia memiliki repositori proyeknya sendiri yang berisi prototipe aplikasi GitHub yang sedang mereka kerjakan. Dia meminta Mary Major untuk menginstal

dan mengkonfigurasi ekstensi yang akan memungkinkannya untuk menghubungkan GitHub repositori ke proyek sebagai repositori sumber tambahan. Mary meninjau repositori GitHub dan bekerja dengan Jorge untuk mengonfigurasi GitHub ekstensi sehingga ia dapat menautkan repositori sebagai GitHub repositori sumber tambahan untuk proyek tersebut.

CodeCatalyst repositori sumber mendukung fungsionalitas standar Git dan bekerja dengan alat berbasis Git yang ada. Anda dapat membuat dan menggunakan token akses pribadi (PAT) sebagai kata sandi khusus aplikasi saat mengkloning dan bekerja dengan repositori sumber dari klien Git atau lingkungan pengembangan terintegrasi (IDE). PAT ini terkait dengan identitas CodeCatalyst pengguna Anda. Untuk informasi selengkapnya, lihat Berikan akses repositori pengguna dengan token akses pribadi.

CodeCatalyst repositori sumber mendukung permintaan tarik. Ini adalah cara sederhana bagi Anda dan anggota proyek lainnya untuk meninjau dan mengomentari perubahan kode sebelum Anda menggabungkannya dari satu cabang ke cabang lainnya. Anda dapat melihat perubahan di CodeCatalyst konsol dan mengomentari baris kode.

Dorongan ke cabang di repositori CodeCatalyst sumber dapat secara otomatis memulai proses dalam alur kerja, di mana perubahan dapat dibuat, diuji, dan diterapkan. Jika repositori sumber Anda dibuat sebagai bagian dari proyek menggunakan template proyek, satu atau lebih alur kerja dikonfigurasi untuk Anda sebagai bagian dari proyek. Anda dapat menambahkan alur kerja tambahan untuk repositori kapan saja. File konfigurasi YAMAL untuk alur kerja dalam proyek disimpan dalam repositori sumber yang dikonfigurasi dalam aksi sumber untuk alur kerja tersebut. Untuk informasi selengkapnya, lihat Memulai dengan alur kerja.

Topik

Konsep repositori sumber

Berikut adalah beberapa konsep yang perlu diketahui saat Anda bekerja dengan repositori CodeCatalyst sumber.

Topik

Proyek

Sebuah proyek merupakan upaya kolaboratif CodeCatalyst yang mendukung tim dan tugas pengembangan. Setelah memiliki proyek, Anda dapat menambahkan, memperbarui, atau menghapus pengguna dan sumber daya, menyesuaikan dasbor proyek Anda, dan memantau kemajuan pekerjaan tim Anda. Anda dapat memiliki beberapa proyek dalam satu ruang.

Repositori sumber khusus untuk proyek tempat Anda membuat atau menautkannya di ruang. Anda tidak dapat membagikan repositori antar proyek, dan Anda tidak dapat menautkan repositori ke lebih dari satu proyek dalam satu spasi. Pengguna dengan peran administrator Kontributor atau Proyek dalam proyek dapat berinteraksi dengan repositori sumber yang terkait dengan proyek tersebut sesuai dengan izin yang diberikan untuk peran tersebut. Untuk informasi selengkapnya, lihat Memberikan akses dengan peran pengguna.

Repositori sumber

Repositori sumber adalah tempat Anda menyimpan kode dan file dengan aman untuk proyek Anda. Ini juga menyimpan riwayat versi file Anda. Secara default, repositori sumber dibagikan

dengan pengguna lain dalam proyek Anda CodeCatalyst . Anda dapat memiliki lebih dari satu repositori sumber untuk sebuah proyek. Anda dapat membuat repositori sumber untuk proyek di CodeCatalyst, atau Anda dapat memilih untuk menautkan repositori sumber yang ada yang dihosting oleh layanan lain jika layanan tersebut didukung oleh ekstensi yang diinstal. Misalnya, Anda dapat menautkan GitHub repositori ke proyek setelah Anda menginstal ekstensi GitHub Repositori. Untuk informasi selengkapnya, lihat Menyimpan kode sumber di repositori untuk proyek di CodeCatalyst dan Mulai cepat: Menginstal ekstensi, menghubungkan penyedia, dan menautkan sumber daya di CodeCatalyst.

Lingkungan Dev

Lingkungan Dev adalah lingkungan pengembangan berbasis cloud yang dapat Anda gunakan CodeCatalyst untuk bekerja dengan cepat pada kode yang disimpan dalam repositori sumber proyek Anda. Alat proyek dan pustaka aplikasi yang disertakan dalam Lingkungan Dev Anda ditentukan oleh devfile di repositori sumber proyek Anda. Jika Anda tidak memiliki devfile di repositori sumber Anda, devfile default akan diterapkan secara otomatis. Devfile default mencakup alat untuk bahasa dan kerangka kerja pemrograman yang paling sering digunakan. Secara default, Dev Environment dikonfigurasi untuk memiliki prosesor 2-core, RAM 4 GB, dan penyimpanan persisten 16 GiB.

Anda dapat memilih untuk mengkloning cabang yang ada dari repositori sumber Anda ke Lingkungan Dev Anda, atau Anda dapat memilih untuk membuat cabang baru sebagai bagian dari pembuatan Lingkungan Dev.

Token akses pribadi (PATs)

Token akses pribadi (PAT) mirip dengan kata sandi. Ini terkait dengan identitas pengguna Anda untuk digunakan di semua ruang dan proyek di CodeCatalyst. Anda menggunakan PAT untuk mengakses CodeCatalyst sumber daya yang mencakup lingkungan pengembangan terintegrasi (IDE) dan repositori sumber berbasis Git. PAT mewakili Anda CodeCatalyst dan Anda dapat mengelolanya di pengaturan pengguna Anda. Seorang pengguna dapat memiliki lebih dari satu PAT. Token akses pribadi hanya ditampilkan sekali. Sebagai praktik terbaik, pastikan untuk menyimpannya dengan aman di komputer lokal Anda. Secara default, PAT kedaluwarsa setelah satu tahun.

Saat bekerja dengan lingkungan pengembangan terintegrasi (IDE), PAT setara dengan kata sandi Git. Berikan PAT ketika diminta kata sandi saat menyiapkan IDE Anda untuk bekerja dengan repositori Git. Untuk informasi selengkapnya tentang cara menghubungkan IDE Anda dengan repositori berbasis Git, lihat dokumentasi untuk IDE Anda.

Cabang

Cabang adalah pointer atau referensi ke komit di Git dan di CodeCatalyst. Anda dapat menggunakan cabang untuk mengatur pekerjaan Anda. Misalnya, Anda dapat menggunakan cabang untuk mengerjakan versi file baru atau berbeda tanpa memengaruhi file di cabang lain. Anda dapat menggunakan cabang untuk mengembangkan fitur baru, menyimpan versi spesifik proyek Anda, dan banyak lagi. Repositori sumber dapat memiliki satu cabang atau banyak cabang. Saat Anda membuat proyek menggunakan template, repositori sumber yang dibuat untuk proyek berisi file sampel di cabang yang disebut main. Cabang utama adalah cabang default untuk repositori.

Cabang default

Repositori sumber CodeCatalyst memiliki cabang default terlepas dari bagaimana Anda membuatnya. Jika Anda memilih untuk membuat proyek menggunakan template, repositori sumber yang dibuat untuk proyek tersebut menyertakan file README.md selain kode sampel, definisi alur kerja, dan sumber daya lainnya. Jika Anda membuat repositori sumber tanpa menggunakan template, file README.md ditambahkan untuk Anda sebagai komit pertama dan cabang default dibuat untuk Anda sebagai bagian dari pembuatan repositori. Cabang default ini bernama main. Cabang default ini adalah cabang yang digunakan sebagai dasar atau cabang default dalam repositori lokal (repos) ketika pengguna mengkloning repositori. Anda dapat mengubah cabang mana yang digunakan sebagai cabang default. Untuk informasi selengkapnya, lihat Mengelola cabang default untuk repositori.

Anda tidak dapat menghapus cabang default untuk repositori sumber. Hasil pencarian hanya menyertakan hasil dari cabang default.

Berkomitmen

Komit adalah perubahan ke file atau kumpulan file. Di CodeCatalyst konsol Amazon, komit menyimpan perubahan Anda dan mendorongnya ke repositori sumber. Komit mencakup informasi tentang perubahan, termasuk identitas pengguna yang melakukan perubahan, waktu dan tanggal perubahan, judul komit, dan pesan apa pun yang disertakan tentang perubahan tersebut. Untuk informasi selengkapnya, lihat Memahami perubahan kode sumber dengan komit di Amazon CodeCatalyst.

Dalam konteks repositori sumber di CodeCatalyst, komit adalah snapshot dari konten dan perubahan pada konten repositori Anda. Anda juga dapat menambahkan tag Git ke commit, untuk mengidentifikasi commit tertentu.

Tarik permintaan

Permintaan tarik adalah cara utama Anda dan pengguna lain meninjau, mengomentari, dan menggabungkan perubahan kode dari satu cabang ke cabang lainnya dalam repositori sumber. Anda dapat menggunakan permintaan tarik untuk meninjau perubahan kode secara kolaboratif untuk perubahan kecil atau perbaikan, penambahan fitur utama, atau versi baru perangkat lunak yang dirilis. Dalam permintaan tarik, Anda dapat meninjau perubahan antara cabang sumber dan tujuan atau perbedaan antara revisi cabang tersebut. Anda dapat menambahkan komentar ke setiap baris perubahan kode serta komentar pada permintaan tarik secara keseluruhan.

> **Tip**
>
> Saat Anda membuat permintaan tarik, perbedaan yang ditampilkan adalah perbedaan antara ujung cabang sumber dan ujung cabang tujuan. Setelah permintaan tarik dibuat, perbedaan yang ditampilkan adalah antara revisi permintaan tarik yang Anda pilih dan komit yang merupakan ujung cabang tujuan saat Anda membuat permintaan tarik. Untuk informasi selengkapnya tentang perbedaan dan penggabungan basis di Git, lihat git-merge-basedi dokumentasi Git.

Revisi

Revisi adalah versi terbaru dari permintaan tarik. Setiap push ke cabang sumber permintaan tarik membuat revisi yang berisi perubahan yang dibuat dalam komit yang disertakan dalam push tersebut. Anda dapat melihat perbedaan antara revisi permintaan tarik selain perbedaan antara cabang sumber dan tujuan. Untuk informasi selengkapnya, lihat Meninjau kode dengan permintaan tarik di Amazon CodeCatalyst.

Alur Kerja

Alur kerja adalah prosedur otomatis yang menjelaskan cara membangun, menguji, dan menyebarkan kode Anda sebagai bagian dari sistem integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD). Alur kerja mendefinisikan serangkaian langkah, atau tindakan, yang harus diambil selama menjalankan alur kerja. Alur kerja juga mendefinisikan peristiwa, atau pemicu, yang menyebabkan alur kerja dimulai. Untuk menyiapkan alur kerja, Anda membuat file definisi alur kerja menggunakan editor visual atau YAMAL CodeCatalyst konsol.

> **Tip**
>
> Untuk melihat sekilas bagaimana Anda dapat menggunakan alur kerja dalam sebuah proyek, buat proyek dengan cetak biru. Setiap cetak biru menerapkan alur kerja yang berfungsi yang dapat Anda tinjau, jalankan, dan bereksperimen.

Repositori sumber juga dapat menyimpan file konfigurasi dan informasi lain untuk alur kerja, pemberitahuan, masalah, dan informasi konfigurasi lainnya untuk proyek. File konfigurasi dibuat dan disimpan dalam repositori sumber ketika Anda membuat sumber daya yang memerlukan file konfigurasi, atau ketika Anda menentukan repositori sebagai tindakan sumber untuk alur kerja. Jika Anda membuat proyek dari cetak biru, Anda akan memiliki file konfigurasi yang sudah disimpan di repositori sumber yang dibuat untuk Anda sebagai bagian dari proyek. Informasi konfigurasi ini disimpan dalam folder bernama `.codecatalyst` di cabang default repositori Anda. Setiap kali Anda membuat cabang cabang default, Anda membuat salinan folder ini dan konfigurasinya selain semua file dan folder lain di cabang itu.

Menyiapkan untuk bekerja dengan repositori sumber

Ketika Anda bekerja dengan repositori sumber CodeCatalyst di Amazon pada mesin lokal Anda, Anda dapat menggunakan Git sendiri atau dalam lingkungan pengembangan terintegrasi (IDE) yang didukung untuk membuat perubahan kode dan mendorong dan menarik kode Anda. Sebagai praktik terbaik, kami menyarankan Anda menggunakan versi terbaru dari Git, dan perangkat lunak lainnya.

> **Note**
>
> Jika Anda menggunakan Dev Environments, Anda tidak perlu menginstal Git. Versi terbaru dari Git disertakan dalam Lingkungan Dev Anda.

Informasi kompatibilitas versi untuk CodeCatalyst

Komponen	Versi
Git	terbaru

Instal Git

Untuk bekerja dengan file, commit, cabang, dan informasi lain di repositori sumber dari klien Git tanpa IDE, instal Git di mesin lokal Anda.

Untuk menginstal Git, kami merekomendasikan situs web seperti Pengunduhan Git.

Buat token akses pribadi

Untuk mengkloning repositori sumber ke mesin lokal Anda atau ke IDE pilihan Anda, Anda harus membuat token akses pribadi (PAT).

Untuk membuat token akses pribadi (PAT)

1. Di bilah menu atas, pilih lencana profil Anda, lalu pilih Pengaturan saya.

   > **Tip**
   >
   > Anda juga dapat menemukan profil pengguna Anda dengan membuka halaman anggota untuk proyek atau ruang dan memilih nama Anda dari daftar anggota.

2. Dalam nama PAT, masukkan nama deskriptif untuk PAT Anda.
3. Di Tanggal kedaluwarsa, tinggalkan tanggal default atau pilih ikon kalender untuk memilih tanggal kustom. Tanggal kedaluwarsa default menjadi satu tahun dari tanggal saat ini.
4. Pilih Buat.

   Anda juga dapat membuat token ini ketika Anda memilih Repositori klon untuk repositori sumber.

5. Simpan rahasia PAT di lokasi yang aman.

   > **Important**
   >
   > Rahasia PAT hanya ditampilkan sekali. Anda tidak dapat mengambilnya setelah Anda menutup jendela.

Memulai dengan repositori CodeCatalyst sumber dan cetak biru aplikasi Single-page

Ikuti langkah-langkah dalam tutorial ini untuk mempelajari cara bekerja dengan repositori sumber di Amazon. CodeCatalyst

Cara tercepat untuk mulai bekerja dengan repositori sumber di Amazon CodeCatalyst adalah dengan membuat proyek menggunakan templat. Saat Anda membuat proyek menggunakan templat, sumber daya dibuat untuk Anda, termasuk repositori sumber yang menyertakan kode sampel. Anda dapat menggunakan contoh repositori dan kode ini untuk mempelajari cara:

- Lihat repositori sumber proyek dan telusuri isinya
- Buat Lingkungan Pengembang dengan cabang baru tempat Anda dapat mengerjakan kode
- Ubah file, dan komit dan dorong perubahan Anda
- Buat permintaan tarik dan tinjau perubahan kode Anda dengan anggota proyek lainnya
- Lihat alur kerja untuk proyek Anda secara otomatis membangun dan menguji perubahan di cabang sumber permintaan tarik
- Gabungkan perubahan Anda dari cabang sumber Anda ke cabang tujuan dan tutup permintaan tarik
- Lihat perubahan gabungan yang dibuat dan diterapkan secara otomatis

Untuk mendapatkan hasil maksimal dari tutorial ini, undang orang lain ke proyek Anda sehingga Anda dapat bekerja sama pada permintaan tarik. Anda juga dapat menjelajahi fitur tambahan CodeCatalyst, seperti membuat masalah dan mengaitkannya dengan permintaan tarik, atau mengonfigurasi notifikasi dan mendapatkan peringatan saat alur kerja terkait berjalan. Untuk eksplorasi lengkap CodeCatalyst, lihatMemulai tutorial.

Membuat proyek dengan cetak biru

Membuat proyek adalah langkah pertama untuk dapat bekerja sama. Anda dapat menggunakan cetak biru untuk membuat proyek Anda, yang juga akan membuat repositori sumber dengan kode sampel dan alur kerja yang secara otomatis akan membangun dan menyebarkan kode Anda ketika Anda mengubahnya. Dalam tutorial ini, kami akan memandu Anda melalui proyek yang dibuat dengan cetak biru aplikasi Single-page, tetapi Anda dapat mengikuti prosedur untuk proyek apa pun dengan repositori sumber. Pastikan untuk memilih peran IAM atau menambahkan peran IAM

jika Anda tidak memilikinya sebagai bagian dari pembuatan proyek. Kami menyarankan Anda menggunakan peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan untuk proyek ini.

Jika Anda sudah memiliki proyek, Anda dapat melompat ke depanMelihat repositori untuk sebuah proyek.

> **Note**
>
> Hanya pengguna dengan administrator Space atau peran pengguna Power yang dapat membuat proyek CodeCatalyst. Jika Anda tidak memiliki peran ini dan Anda memerlukan proyek untuk dikerjakan untuk tutorial ini, mintalah seseorang dengan salah satu peran ini untuk membuat proyek untuk Anda dan menambahkan Anda ke proyek yang dibuat. Untuk informasi selengkapnya, lihat Memberikan akses dengan peran pengguna.

Untuk membuat proyek dengan cetak biru

1. Di CodeCatalyst konsol, arahkan ke ruang tempat Anda ingin membuat proyek.
2. Di dasbor ruang, pilih Buat proyek.
3. Pilih Mulai dengan cetak biru.
4. Dari cetak biru atau tab CodeCatalyst Blueprints Space, pilih cetak biru, lalu pilih Berikutnya.
5. Di bawah Nama proyek Anda, masukkan nama yang ingin Anda tetapkan ke proyek Anda dan nama sumber daya yang terkait. Nama harus unik di dalam ruang Anda.
6. (Opsional) Secara default, kode sumber yang dibuat oleh cetak biru disimpan dalam repositori. CodeCatalyst Atau, Anda dapat memilih untuk menyimpan kode sumber cetak biru di repositori pihak ketiga. Untuk informasi selengkapnya, lihat Tambahkan fungsionalitas ke proyek dengan ekstensi di CodeCatalyst.

   Lakukan salah satu hal berikut tergantung pada penyedia repositori pihak ketiga yang ingin Anda gunakan:

   - GitHub repositori: Hubungkan akun. GitHub

     Pilih menu tarik-turun Advanced, pilih GitHub sebagai penyedia repositori, lalu pilih GitHub akun tempat Anda ingin menyimpan kode sumber yang dibuat oleh cetak biru.

> **Note**
>
> Jika Anda menghubungkan GitHub akun, Anda harus membuat koneksi pribadi untuk membuat pemetaan identitas antara CodeCatalyst identitas dan GitHub identitas Anda. Untuk informasi selengkapnya, lihat Koneksi pribadi dan Mengakses GitHub sumber daya dengan koneksi pribadi.

- Repositori Bitbucket: Hubungkan ruang kerja Bitbucket.

  Pilih menu tarik-turun Advanced, pilih Bitbucket sebagai penyedia repositori, lalu pilih ruang kerja Bitbucket tempat Anda ingin menyimpan kode sumber yang dibuat oleh cetak biru.

7. Di bawah sumber daya Proyek, konfigurasikan parameter cetak biru. Tergantung pada cetak biru, Anda mungkin memiliki opsi untuk memberi nama nama repositori sumber.
8. (Opsional) Untuk melihat file definisi dengan pembaruan berdasarkan pilihan parameter proyek yang Anda buat, pilih Lihat kode atau Lihat alur kerja dari Hasilkan pratinjau proyek.
9. (Opsional) Pilih Lihat detail dari kartu cetak biru untuk melihat detail spesifik tentang cetak biru, seperti ikhtisar arsitektur cetak biru, koneksi dan izin yang diperlukan, dan jenis sumber daya yang dibuat cetak biru.
10. Pilih Buat proyek.

Halaman ikhtisar proyek terbuka segera setelah Anda membuat proyek atau menerima undangan ke proyek dan menyelesaikan proses masuk. Halaman ikhtisar proyek untuk proyek baru tidak berisi masalah terbuka atau permintaan tarik. Anda secara opsional dapat memilih untuk membuat masalah dan menetapkannya untuk diri Anda sendiri. Anda juga dapat memilih untuk mengundang orang lain ke proyek Anda. Untuk informasi selengkapnya, lihat Membuat masalah di CodeCatalyst dan Mengundang pengguna ke proyek.

Melihat repositori untuk sebuah proyek

Sebagai anggota proyek, Anda dapat melihat repositori sumber untuk proyek tersebut. Anda juga dapat memilih untuk membuat repositori tambahan. Jika seseorang dengan peran administrator Space telah menginstal dan mengonfigurasi GitHub repositori atau ekstensi Bitbucket, Anda juga dapat menambahkan tautan ke repositori pihak ketiga di GitHub akun atau ruang kerja Bitbucket yang dikonfigurasi untuk ekstensi. Untuk informasi selengkapnya, lihat Membuat repositori sumber dan Mulai cepat: Menginstal ekstensi, menghubungkan penyedia, dan menautkan sumber daya di CodeCatalyst.

> **Note**
>
> ***Untuk proyek yang dibuat dengan cetak biru aplikasi Single-page, nama default untuk repositori sumber yang berisi kode sampel adalah spa-app.***

Untuk menavigasi ke repositori sumber untuk sebuah proyek

1. Arahkan ke proyek Anda, dan lakukan salah satu hal berikut:
   - Pada halaman ringkasan untuk proyek Anda, pilih repositori yang Anda inginkan dari daftar, lalu pilih Lihat repositori.
   - Di panel navigasi, pilih Kode, lalu pilih Repositori sumber. Di repositori Sumber, pilih nama repositori dari daftar. Anda dapat memfilter daftar repositori dengan mengetikkan bagian dari nama repositori di bilah filter.
2. Pada halaman beranda untuk repositori, lihat isi repositori dan informasi tentang sumber daya terkait seperti jumlah permintaan tarik dan alur kerja. Secara default, konten untuk cabang default ditampilkan. Anda dapat mengubah tampilan dengan memilih cabang yang berbeda dari daftar drop-down.

Halaman ikhtisar untuk repositori mencakup informasi tentang alur kerja dan permintaan tarik yang dikonfigurasi untuk cabang repositori ini dan file-nya. Jika Anda baru saja membuat proyek, alur kerja awal untuk membangun, menguji, dan menyebarkan kode akan tetap berjalan, karena butuh beberapa menit untuk menyelesaikannya. Anda dapat melihat alur kerja terkait dan statusnya dengan memilih nomor di bawah alur kerja Terkait, tetapi ini membuka halaman Alur Kerja di CI/CD. Untuk tutorial ini, tetap di halaman ikhtisar dan jelajahi kode di repositori. Isi `README.md` file ditampilkan pada halaman ini di bawah file repositori. Di File, isi cabang default ditampilkan. Anda dapat mengubah tampilan file untuk menampilkan isi cabang lain jika Anda memilikinya. `.codecatalyst`Folder berisi kode yang digunakan untuk bagian lain dari proyek, seperti alur kerja file YAMAL.

Untuk melihat konten folder, pilih panah di sebelah nama folder untuk memperluasnya. Misalnya, pilih panah di sebelah `src` untuk melihat file untuk aplikasi web satu halaman yang terdapat dalam folder itu. Untuk melihat isi dari sebuah file, pilih file tersebut dari daftar. Ini akan membuka Lihat file, di mana Anda dapat menelusuri konten beberapa file. Anda dapat mengedit file tunggal di konsol juga, tetapi untuk mengedit beberapa file, Anda akan ingin membuat Lingkungan Dev.

Membuat Lingkungan Dev

Anda dapat menambahkan dan mengubah file di repositori sumber di konsol Amazon CodeCatalyst . Namun, untuk bekerja secara efektif dengan banyak file dan cabang, kami sarankan menggunakan Dev Environment atau mengkloning repositori ke komputer lokal Anda. Dalam tutorial ini, kita akan membuat AWS Cloud9 Dev Environment dengan cabang bernama**`develop`**. Anda dapat memilih nama cabang yang berbeda, tetapi dengan menamai cabang**`develop`**, alur kerja akan secara otomatis berjalan untuk membangun dan menguji kode Anda ketika Anda membuat permintaan tarik nanti dalam tutorial ini.

> **Tip**
>
> Jika Anda memutuskan untuk mengkloning repositori lokal alih-alih atau selain menggunakan Lingkungan Pengembang, pastikan bahwa Anda memiliki Git di komputer lokal Anda atau IDE Anda menyertakan Git. Untuk informasi selengkapnya, lihat Menyiapkan untuk bekerja dengan repositori sumber.

Untuk membuat Lingkungan Dev dengan cabang baru

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek tempat Anda ingin membuat Lingkungan Dev.
3. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, pilih repositori Sumber, dan pilih repositori yang ingin Anda buat Lingkungan Dev.
4. Pada halaman rumah repositori, pilih Create Dev Environment.
5. Pilih IDE yang didukung dari menu tarik-turun. Untuk informasi selengkapnya, lihat Lingkungan pengembangan terintegrasi yang didukung untuk Lingkungan Dev.
6. Pilih repositori untuk dikloning, pilih Bekerja di cabang baru, masukkan nama cabang ke bidang Nama cabang, dan pilih cabang untuk membuat cabang baru dari Buat cabang dari menu drop-down.
7. Secara opsional, tambahkan alias untuk Lingkungan Dev.
8. Secara opsional, pilih tombol edit konfigurasi Lingkungan Dev untuk mengedit konfigurasi komputasi, penyimpanan, atau batas waktu Dev Environment.
9. Pilih Buat. Saat Dev Environment Anda sedang dibuat, kolom status Dev Environment akan menampilkan Mulai, dan kolom status akan ditampilkan Running setelah Dev Environment telah

dibuat. Tab baru akan terbuka dengan Lingkungan Dev Anda di IDE pilihan Anda. Anda dapat mengedit kode dan melakukan dan mendorong perubahan Anda.

Setelah Anda membuat Lingkungan Dev, Anda dapat mengedit file, melakukan perubahan, dan mendorong perubahan Anda ke **`test`** cabang. Untuk tutorial ini, edit konten antara <p> tag dalam `App.tsx` file di `src` folder untuk mengubah teks yang ditampilkan di halaman web. Komit dan dorong perubahan Anda, lalu kembali ke CodeCatalyst tab.

Untuk membuat dan mendorong perubahan dari Lingkungan AWS Cloud9 Pengembang

1. Di AWS Cloud9, perluas menu navigasi samping untuk menelusuri file. Perluas`src`, dan buka`App.tsx`.
2. Buat perubahan teks di dalam <p> tag.
3. Simpan file, lalu komit dan dorong perubahan Anda dengan menggunakan menu Git. Atau, di jendela terminal, komit dan dorong perubahan Anda dengan git push perintah git commit dan.

```
git commit -am "Making an example change"
git push
```

> **Tip**
>
> Anda mungkin perlu mengubah direktori di terminal ke direktori repositori Git sebelum Anda berhasil menjalankan perintah Git.

Membuat permintaan pull

Anda dapat menggunakan permintaan tarik untuk meninjau perubahan kode secara kolaboratif untuk perubahan kecil atau perbaikan, penambahan fitur utama, atau versi baru perangkat lunak yang dirilis. Dalam tutorial ini, Anda akan membuat permintaan tarik untuk meninjau perubahan yang Anda buat pada cabang *`pengujian`* dibandingkan dengan cabang utama. Membuat permintaan tarik dalam proyek yang dibuat dengan template juga akan memulai menjalankan alur kerja terkait, jika ada.

Untuk membuat permintaan tarik

1. Arahkan ke proyek Anda.

2. Lakukan salah satu hal berikut ini:
   - Di panel navigasi, pilih Kode, pilih Tarik permintaan, lalu pilih Buat permintaan tarik.
   - Pada halaman beranda repositori, pilih Lainnya, lalu pilih Buat permintaan tarik.
   - Pada halaman proyek, pilih Buat permintaan tarik.
3. Di repositori Sumber, pastikan bahwa repositori sumber yang ditentukan adalah yang berisi kode yang dikomit. Opsi ini hanya muncul jika Anda tidak membuat permintaan tarik dari halaman utama repositori.
4. Di cabang Tujuan, pilih cabang untuk menggabungkan kode setelah ditinjau.
5. Di cabang Sumber, pilih cabang yang berisi kode komit.
6. Dalam judul permintaan tarik, masukkan judul yang membantu pengguna lain memahami apa yang perlu ditinjau dan alasannya.
7. (Opsional) Dalam deskripsi permintaan Tarik, berikan informasi seperti tautan ke masalah atau deskripsi perubahan Anda.

   > **Tip**
   >
   > Anda dapat memilih Tulis deskripsi agar saya CodeCatalyst secara otomatis menghasilkan deskripsi tentang perubahan yang terkandung dalam permintaan tarik. Anda dapat membuat perubahan pada deskripsi yang dibuat secara otomatis setelah Anda menambahkannya ke permintaan tarik.
   > Fungsionalitas ini mengharuskan fitur AI generatif diaktifkan untuk ruang tersebut. Untuk informasi selengkapnya, lihat Mengelola fitur AI generatif.

8. (Opsional) Dalam Masalah, pilih Masalah tautan, lalu pilih masalah dari daftar atau masukkan ID-nya. Untuk memutuskan tautan masalah, pilih ikon batalkan tautan.
9. (Opsional) Di Reviewer yang diperlukan, pilih Tambahkan pengulas yang diperlukan. Pilih dari daftar anggota proyek untuk menambahkannya. Pengulas yang diperlukan harus menyetujui perubahan sebelum permintaan tarik dapat digabungkan ke cabang tujuan.

   > **Note**
   >
   > Anda tidak dapat menambahkan pengulas sebagai pengulas yang diperlukan dan pengulas opsional. Anda tidak dapat menambahkan diri Anda sebagai reviewer.

10. (Opsional) Di pengulas opsional, pilih Tambahkan pengulas opsional. Pilih dari daftar anggota proyek untuk menambahkannya. Peninjau opsional tidak harus menyetujui perubahan sebagai persyaratan sebelum permintaan tarik dapat digabungkan ke cabang tujuan.
11. Tinjau perbedaan antara cabang. Perbedaan yang ditampilkan dalam permintaan tarik adalah perubahan antara revisi di cabang sumber dan basis gabungan, yang merupakan komit kepala cabang tujuan pada saat permintaan tarik dibuat. Jika tidak ada perubahan yang ditampilkan, cabang mungkin identik, atau Anda mungkin telah memilih cabang yang sama untuk sumber dan tujuan.
12. Ketika Anda puas bahwa permintaan tarik berisi kode dan perubahan yang ingin Anda tinjau, pilih Buat.

> **Note**
>
> Setelah Anda membuat permintaan tarik, Anda dapat menambahkan komentar. Komentar dapat ditambahkan ke permintaan tarik atau ke baris individual dalam file serta permintaan tarik keseluruhan. Anda dapat menambahkan tautan ke sumber daya, seperti file, dengan menggunakan tanda @ diikuti dengan nama file.

Anda dapat melihat informasi tentang alur kerja terkait yang dimulai dengan pembuatan permintaan tarik ini dengan memilih Ikhtisar dan kemudian meninjau informasi di area detail permintaan Tarik di bawah Alur kerja berjalan. Untuk melihat alur kerja berjalan, pilih run.

> **Tip**
>
> Jika Anda menamai cabang Anda sesuatu selain**`develop`**, alur kerja tidak akan otomatis berjalan untuk membangun dan menguji perubahan Anda. Jika Anda ingin mengonfigurasinya, edit file YAMAL untuk onPullRequestBuildAndTestalur kerja. Untuk informasi selengkapnya, lihat Membuat alur kerja.

Anda dapat mengomentari permintaan tarik ini dan meminta anggota proyek lain untuk mengomentarinya. Anda juga dapat memilih untuk menambah atau mengubah pengulas opsional atau wajib. Anda dapat memilih untuk membuat lebih banyak perubahan pada cabang sumber untuk repositori, dan melihat bagaimana perubahan komit tersebut membuat revisi untuk permintaan tarik. Untuk informasi lebih lanjut, lihatMeninjau permintaan tarik,Memperbarui permintaan tarik,Meninjau

kode dengan permintaan tarik di Amazon CodeCatalyst, danMelihat alur kerja menjalankan status dan detail.

Menggabungkan permintaan tarik

Setelah permintaan tarik ditinjau dan telah menerima persetujuan dari pengulas yang diperlukan, Anda dapat menggabungkan cabang sumbernya ke cabang tujuan di konsol. CodeCatalyst Menggabungkan permintaan tarik juga akan memulai proses perubahan melalui alur kerja apa pun yang terkait dengan cabang tujuan. Dalam tutorial ini, Anda akan menggabungkan cabang pengujian menjadi main, yang akan memulai menjalankan onPushToMainDeployPipelinealur kerja.

Untuk menggabungkan permintaan tarik (konsol)

1. Di Permintaan tarik, pilih permintaan tarik yang Anda buat di langkah sebelumnya. Dalam permintaan tarik, pilih Gabung.
2. Pilih dari strategi penggabungan yang tersedia untuk permintaan tarik. Secara opsional pilih atau batalkan pilihan untuk menghapus cabang sumber setelah menggabungkan permintaan tarik, dan kemudian pilih Gabung. Setelah penggabungan selesai, status permintaan tarik berubah menjadi Gabungan dan tidak lagi muncul dalam tampilan default permintaan tarik. Tampilan default menunjukkan permintaan tarik dengan status Open. Anda masih dapat melihat permintaan tarik gabungan, tetapi Anda tidak dapat menyetujuinya atau mengubah statusnya.

> **Note**
>
> Jika tombol Gabung tidak aktif, atau Anda melihat label Tidak dapat digabungkan, pengulas wajib belum menyetujui permintaan tarik, atau permintaan tarik tidak dapat digabungkan di konsol. CodeCatalyst Peninjau yang belum menyetujui permintaan tarik ditunjukkan oleh ikon jam di Ikhtisar di area detail permintaan Tarik. Jika semua pengulas yang diperlukan telah menyetujui permintaan tarik tetapi tombol Gabung masih belum aktif, Anda mungkin memiliki konflik gabungan, atau Anda telah melampaui kuota penyimpanan untuk ruang tersebut. Anda dapat menyelesaikan konflik gabungan untuk cabang tujuan di Lingkungan Pengembang, mendorong perubahan, lalu menggabungkan permintaan tarik, atau Anda dapat menyelesaikan konflik dan menggabungkan secara lokal, lalu mendorong komit yang berisi penggabungan. CodeCatalyst Untuk informasi selengkapnya, lihat Menggabungkan permintaan tarik (Git) dan dokumentasi Git Anda.

Melihat kode yang digunakan

Sekarang saatnya untuk melihat kode yang awalnya diterapkan yang ada di cabang default, dan perubahan gabungan Anda setelah kode tersebut dibuat, diuji, dan diterapkan secara otomatis. Untuk melakukannya, Anda dapat kembali ke halaman ikhtisar untuk repositori dan memilih nomor di sebelah ikon alur kerja terkait, atau di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.

Untuk melihat kode yang digunakan

1. Di Alur Kerja, dion`PushToMainDeployPipeline`, perluas Runs terbaru.

   > **Note**
   >
   > Ini adalah nama default alur kerja untuk proyek yang dibuat dengan cetak biru aplikasi Single-page.

2. Proses terbaru adalah yang dimulai oleh komit permintaan tarik gabungan Anda ke `main` cabang dan kemungkinan akan menampilkan status Sedang berlangsung. Pilih proses yang berhasil diselesaikan dari daftar untuk membuka detail proses tersebut.
3. Pilih Variabel. Salin nilai untuk AppUrl. Ini adalah URL untuk aplikasi web halaman tunggal yang digunakan. Buka tab browser baru dan tempel nilainya untuk melihat kode yang dibangun dan diterapkan. Biarkan tab terbuka.
4. Kembali ke daftar alur kerja yang berjalan dan tunggu hingga proses terbaru selesai. Ketika itu terjadi, kembali ke tab yang Anda buka untuk melihat aplikasi web dan menyegarkan browser Anda. Anda akan melihat perubahan yang Anda buat dalam permintaan tarik gabungan Anda.

Membersihkan sumber daya

Setelah Anda menjelajahi bekerja dengan repositori sumber dan permintaan tarik, Anda mungkin ingin menghapus sumber daya apa pun yang tidak Anda butuhkan. Anda tidak dapat menghapus permintaan tarik, tetapi Anda dapat menutupnya. Anda dapat menghapus cabang apa pun yang Anda buat.

Jika Anda tidak lagi memerlukan repositori sumber atau proyek, Anda juga dapat menghapus sumber daya tersebut. Lihat informasi yang lebih lengkap di Menghapus repositori sumber dan Menghapus proyek.

Menyimpan kode sumber di repositori untuk proyek di CodeCatalyst

Repositori sumber adalah tempat Anda menyimpan kode dan file dengan aman untuk proyek Anda. Ini juga menyimpan riwayat sumber Anda, dari komit pertama hingga perubahan terbaru. Jika Anda memilih cetak biru yang menyertakan repositori sumber, repositori itu juga berisi file konfigurasi dan informasi lain untuk alur kerja dan notifikasi untuk proyek. Informasi konfigurasi ini disimpan dalam folder bernama .codecatalyst.

Anda dapat membuat repositori sumber CodeCatalyst baik dengan membuat proyek dengan cetak biru yang membuat repositori sumber sebagai bagian dari pembuatan proyek, atau dengan membuat repositori sumber dalam proyek yang ada. Pengguna proyek akan secara otomatis melihat dan dapat menggunakan repositori yang Anda buat untuk sebuah proyek. Anda juga dapat memilih untuk menautkan repositori Git yang dihosting di GitHub atau Bibbucket ke proyek Anda. Ketika Anda melakukannya, pengguna proyek Anda dapat melihat dan mengakses repositori tertaut itu dalam daftar repositori untuk proyek.

> **Note**
>
> Sebelum Anda dapat menautkan repositori, Anda harus menginstal ekstensi untuk layanan yang menghostingnya. Anda tidak dapat menautkan repositori yang diarsipkan. Meskipun Anda dapat menautkan repositori kosong, Anda tidak dapat menggunakannya CodeCatalyst sampai Anda menginisialisasinya dengan komit awal yang membuat cabang default. Untuk informasi selengkapnya, lihat Memasang ekstensi di ruang.

Secara default, repositori sumber dibagikan dengan anggota lain dari proyek Amazon CodeCatalyst Anda. Anda dapat membuat repositori sumber tambahan untuk proyek atau menautkan repositori ke proyek. Semua anggota proyek dapat melihat, menambah, mengedit, dan menghapus file dan folder di repositori sumber proyek.

Untuk mengerjakan kode dengan cepat di repositori sumber, Anda dapat membuat Lingkungan Dev yang mengkloning repositori tertentu dan bercabang ke dalamnya di mana Anda dapat mengerjakan kode di lingkungan pengembangan terintegrasi (IDE) yang Anda pilih untuk Lingkungan Dev. Anda dapat mengkloning repositori sumber di komputer lokal Anda dan menarik dan mendorong perubahan antara repo lokal Anda dan repositori jarak jauh. CodeCatalyst Anda juga dapat bekerja dengan repositori sumber dengan mengonfigurasi akses ke mereka di IDE pilihan Anda selama IDE tersebut mendukung manajemen kredensi.

Nama repositori harus unik dalam sebuah CodeCatalyst proyek.

Topik

Membuat repositori sumber

Saat Anda membuat proyek menggunakan cetak biru di Amazon CodeCatalyst, CodeCatalyst buat repositori sumber untuk Anda. Repositori sumber itu berisi kode contoh selain informasi konfigurasi untuk alur kerja dan sumber daya lain yang dibuat untuk Anda. Ini adalah cara yang disarankan untuk memulai dengan repositori di. CodeCatalyst Anda dapat memilih untuk membuat repositori untuk sebuah proyek. Repositori tersebut akan berisi satu file, file README.md yang dapat Anda edit atau hapus kapan saja. Bergantung pada pilihan Anda saat membuat repositori sumber, repositori mungkin juga berisi file. `.gitignore`

Untuk membuat repositori sumber

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda.
3. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
4. Pilih Tambahkan repositori, lalu pilih Buat repositori.
5. Dalam nama Repositori, berikan nama untuk repositori. Dalam panduan ini, kami menggunakan *`codecatalyst-source-repository`*, tetapi Anda dapat memilih nama yang berbeda. Nama repositori harus unik dalam sebuah proyek. Untuk informasi selengkapnya tentang persyaratan untuk nama repositori, lihat. Kuota untuk repositori sumber di CodeCatalyst
6. (Opsional) Dalam Deskripsi, tambahkan deskripsi untuk repositori yang akan membantu pengguna lain dalam proyek memahami untuk apa repositori digunakan.
7. (Opsional) Tambahkan `.gitignore` file untuk jenis kode yang Anda rencanakan untuk dorong.
8. Pilih Buat.

> **Note**
>
> CodeCatalyst menambahkan `README.md` file ke repositori Anda saat Anda membuatnya. CodeCatalystjuga membuat komit awal untuk repositori di cabang default bernama main. Anda dapat mengedit atau menghapus file README.md, tetapi Anda tidak dapat mengubah atau menghapus cabang default.

Menautkan repositori sumber

Saat menautkan repositori sumber ke proyek, Anda dapat menyertakan repositori yang memiliki CodeCatalyst ekstensi untuk layanan yang menghosting repositori, jika ekstensi itu diinstal untuk ruang Anda. Hanya pengguna dengan peran administrator Space yang dapat menginstal ekstensi. Setelah ekstensi diinstal, Anda dapat menautkan ke repositori yang dikonfigurasi untuk akses oleh ekstensi itu. Untuk informasi lebih lanjut, lihat Memasang ekstensi di ruang atau ikutiMenautkan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.

> **Important**
>
> Meskipun Anda dapat menautkan repositori GitHub atau Bitbucket sebagai Kontributor, Anda hanya dapat memutuskan tautan repositori pihak ketiga sebagai administrator Space atau administrator Project. Untuk informasi selengkapnya, lihat Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.

> **Important**
>
> Setelah Anda menginstal ekstensi repositori, repositori apa pun yang Anda tautkan CodeCatalyst akan memiliki kode mereka diindeks dan disimpan. CodeCatalyst Ini akan membuat kode dapat dicari di. CodeCatalyst Untuk lebih memahami perlindungan data untuk kode Anda saat menggunakan repositori tertaut di CodeCatalyst, lihat Perlindungan data di CodeCatalyst Panduan Pengguna Amazon.

> **Note**
>
> - Repositori GitHub atau Bitbucket hanya dapat ditautkan ke satu CodeCatalyst proyek dalam satu spasi.
> - Anda tidak dapat menggunakan repositori kosong atau diarsipkan GitHub atau Bitbucket dengan proyek. CodeCatalyst
> - Anda tidak dapat menautkan repositori GitHub atau Bitbucket yang memiliki nama yang sama dengan repositori dalam proyek. CodeCatalyst
> - Ekstensi GitHub repositori tidak kompatibel dengan repositori GitHub Enterprise Server.
> - Ekstensi repositori Bitbucket tidak kompatibel dengan repositori Bitbucket Data Center.

Untuk menautkan repositori sumber

1. Arahkan ke proyek tempat Anda ingin menautkan repositori.

   > **Note**
   >
   > Sebelum Anda dapat menautkan repositori, pengguna dengan peran administrator Space harus terlebih dahulu menginstal ekstensi untuk penyedia yang menghosting repositori. Untuk informasi selengkapnya, lihat Memasang ekstensi di ruang.

2. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
3. Pilih Tambahkan repositori, lalu pilih Repositori tautan.
4. Dari menu tarik-turun penyedia Repositori, pilih salah satu penyedia repositori pihak ketiga berikut: atau Bitbucket. GitHub
5. Lakukan salah satu hal berikut tergantung pada penyedia repositori pihak ketiga yang Anda pilih untuk ditautkan:

   - GitHub repositori: Tautkan repositori. GitHub

     1. Dari menu dropdown GitHub akun, pilih GitHub akun yang berisi repositori yang ingin Anda tautkan.
     2. Dari menu tarik-turun GitHub repositori, pilih GitHub akun yang ingin Anda tautkan proyek Anda. CodeCatalyst

3. (Opsional) Jika Anda tidak melihat GitHub repositori dalam daftar repositori, mungkin tidak dikonfigurasi untuk akses repositori di aplikasi Amazon di. CodeCatalyst GitHub Anda dapat mengonfigurasi GitHub repositori mana yang dapat digunakan CodeCatalyst di akun yang terhubung.

   a. Arahkan ke GitHubakun Anda, pilih Pengaturan, lalu pilih Aplikasi.

   b. Di tab GitHub Aplikasi Terinstal, pilih Konfigurasi untuk CodeCatalyst aplikasi Amazon.

   c. Lakukan salah satu hal berikut untuk mengonfigurasi akses GitHub repositori yang ingin Anda tautkan: CodeCatalyst

      - Untuk menyediakan akses ke semua repositori saat ini dan masa depan, pilih Semua repositori.
      - Untuk menyediakan akses ke repositori tertentu, pilih Hanya pilih repositori, pilih menu tarik-turun Pilih repositori, lalu pilih repositori yang ingin Anda izinkan untuk ditautkan. CodeCatalyst

- Repositori Bitbucket: Tautkan repositori Bitbucket.

  1. Dari menu tarik-turun ruang kerja Bitbucket, pilih ruang kerja Bitbucket yang berisi repositori yang ingin Anda tautkan.
  2. Dari menu dropdown repositori Bitbucket, pilih repositori Bitbucket yang ingin Anda tautkan proyek Anda. CodeCatalyst

> **Tip**
>
> Jika nama repositori berwarna abu-abu, Anda tidak dapat menautkan repositori itu karena telah ditautkan ke proyek lain di Amazon. CodeCatalyst

6. Pilih Tautkan.

Jika Anda tidak lagi ingin menggunakan repositori GitHub atau Bitbucket CodeCatalyst, Anda dapat memutuskan tautannya dari proyek. CodeCatalyst Ketika repositori tidak terhubung, peristiwa dalam repositori itu tidak akan memulai alur kerja berjalan, dan Anda tidak akan dapat menggunakan repositori itu dengan Dev Environments. CodeCatalyst Untuk informasi selengkapnya, lihat Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.

Melihat repositori sumber

Anda dapat melihat repositori sumber yang terkait dengan proyek di Amazon. CodeCatalyst Untuk repositori sumber di CodeCatalyst, halaman ikhtisar untuk repositori memberikan gambaran singkat tentang informasi dan aktivitas di repositori tersebut, termasuk:

- Deskripsi repositori, jika ada
- Jumlah cabang di repositori
- Jumlah permintaan tarik terbuka untuk repositori
- Jumlah alur kerja terkait untuk repositori
- File dan folder di cabang default, atau cabang yang Anda pilih
- Judul, penulis, dan tanggal komit terakhir ke cabang yang ditampilkan
- Isi file README.md dirender dalam Markdown, jika ada file README.md disertakan

Halaman ini juga menyediakan tautan ke komit, cabang, dan permintaan tarik untuk repositori, serta cara cepat untuk membuka, melihat, dan mengedit file individual.

> **Note**
>
> Anda tidak dapat melihat informasi ini tentang repositori tertaut di konsol. CodeCatalyst Untuk melihat informasi tentang repositori tertaut, pilih tautan dalam daftar repositori untuk membuka repositori itu di layanan yang menghostingnya.

Untuk menavigasi ke repositori sumber untuk sebuah proyek

1. Arahkan ke proyek Anda, dan lakukan salah satu hal berikut:
   - Pada halaman ringkasan untuk proyek Anda, pilih repositori yang Anda inginkan dari daftar, lalu pilih Lihat repositori.
   - Di panel navigasi, pilih Kode, lalu pilih Repositori sumber. Di repositori Sumber, pilih nama repositori dari daftar. Anda dapat memfilter daftar repositori dengan mengetikkan bagian dari nama repositori di bilah filter.
2. Pada halaman beranda untuk repositori, lihat isi repositori dan informasi tentang sumber daya terkait seperti jumlah permintaan tarik dan alur kerja. Secara default, konten untuk cabang default ditampilkan. Anda dapat mengubah tampilan dengan memilih cabang yang berbeda dari daftar drop-down.

> **Tip**
>
> Anda juga dapat dengan cepat menavigasi ke repositori proyek Anda dengan memilih Lihat kode proyek dari halaman ringkasan proyek.

Mengedit pengaturan untuk repositori sumber

Anda dapat mengelola pengaturan untuk repositori Anda, termasuk mengedit deskripsi repositori, memilih cabang default, membuat dan mengelola aturan cabang, dan membuat dan mengelola aturan persetujuan untuk permintaan tarik masuk. CodeCatalyst Ini dapat membantu anggota proyek memahami untuk apa repositori digunakan, dan membantu Anda menegakkan praktik dan proses terbaik yang digunakan oleh tim.

> **Note**
>
> Anda tidak dapat mengedit nama repositori sumber.
> Anda tidak dapat mengedit nama, deskripsi, atau informasi lain untuk repositori tertaut di. CodeCatalyst Untuk mengubah informasi tentang repositori tertaut, Anda harus mengeditnya di penyedia yang menghosting repositori tertaut. Untuk informasi selengkapnya, lihat dokumentasi untuk layanan yang menghosting repositori tertaut.

Untuk mengedit pengaturan repositori

1. Di CodeCatalyst konsol, arahkan ke proyek yang berisi repositori sumber yang pengaturannya ingin Anda edit.
2. Pada halaman ringkasan untuk proyek Anda, pilih repositori yang Anda inginkan dari daftar, lalu pilih Lihat repositori. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber. Pilih nama repositori dari daftar repositori sumber untuk proyek.
3. Pada halaman ikhtisar untuk repositori, pilih Lainnya, lalu pilih Kelola pengaturan.
4. Lakukan salah satu atau beberapa hal berikut:
   - Edit deskripsi repositori dan kemudian pilih Simpan.
   - Untuk mengubah cabang default untuk repositori, di cabang Default, pilih Edit. Untuk informasi selengkapnya, lihat Mengelola cabang default untuk repositori.

- Untuk menambah, menghapus, atau mengubah aturan untuk peran proyek apa yang memiliki izin untuk melakukan tindakan tertentu di cabang, dalam aturan Branch, pilih Edit. Untuk informasi selengkapnya, lihat Mengelola tindakan yang diizinkan untuk cabang dengan aturan cabang.
- Untuk menambah, menghapus, atau mengubah aturan persetujuan untuk menggabungkan tarik tarik ke cabang, dalam aturan Persetujuan, pilih Edit. Untuk informasi selengkapnya, lihat Mengelola persyaratan untuk menggabungkan permintaan tarik dengan aturan persetujuan.

Mengkloning repositori sumber

Untuk bekerja secara efektif dengan beberapa file, cabang, dan komit di repositori sumber, kloning repositori sumber ke komputer lokal Anda dan gunakan klien Git atau lingkungan pengembangan terintegrasi (IDE) untuk membuat perubahan. Komit dan dorong perubahan Anda ke repositori sumber agar dapat bekerja dengan CodeCatalyst fitur seperti masalah dan permintaan tarik. Anda juga dapat memilih untuk membuat Lingkungan Dev untuk bekerja pada kode. Membuat Lingkungan Dev secara otomatis mengkloning repositori dan cabang yang Anda tentukan ke dalam Lingkungan Dev.

> **Note**
>
> Anda tidak dapat mengkloning repositori tertaut di CodeCatalyst konsol atau membuat Lingkungan Dev untuk mereka. Untuk mengkloning repositori tertaut secara lokal, pilih tautan dalam daftar repositori untuk membuka repositori itu di layanan yang menghostingnya, lalu kloningnya. Untuk informasi selengkapnya, lihat dokumentasi untuk layanan yang menghosting repositori tertaut.

Untuk membuat Lingkungan Dev dari repositori sumber

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
3. Pilih repositori sumber tempat Anda ingin mengerjakan kode.
4. Pilih Buat Lingkungan Pengembang.
5. Pilih IDE yang didukung dari menu tarik-turun. Untuk informasi selengkapnya, lihat Lingkungan pengembangan terintegrasi yang didukung untuk Lingkungan Dev.
6. Lakukan salah satu hal berikut ini:

- Pilih Bekerja di cabang yang ada, lalu pilih cabang dari menu drop-down cabang yang ada.
- Pilih Bekerja di cabang baru, masukkan nama cabang ke bidang Nama cabang, dan pilih cabang untuk membuat cabang baru dari menu drop-down Buat cabang dari.

7. Secara opsional tambahkan nama untuk Lingkungan Dev atau edit konfigurasinya.
8. Pilih Buat.

Untuk mengkloning repositori sumber

1. Arahkan ke proyek Anda.
2. Pada halaman ringkasan untuk proyek Anda, pilih repositori yang Anda inginkan dari daftar, lalu pilih Lihat repositori. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber. Pilih nama repositori dari daftar repositori sumber untuk proyek. Anda dapat memfilter daftar repositori dengan mengetikkan bagian dari nama repositori di bilah filter.
3.
4. Pilih Repositori klon. Salin URL klon untuk repositori.

> **Note**
>
> Jika Anda tidak memiliki token akses pribadi (PAT), pilih Buat token. Salin token dan simpan di lokasi yang aman. Anda akan menggunakan PAT ini ketika diminta untuk kata sandi oleh klien Git Anda atau lingkungan pengembangan terintegrasi (IDE).

5. Lakukan salah satu hal berikut ini:
   - Untuk mengkloning repositori ke komputer lokal Anda, buka terminal atau baris perintah dan jalankan git clone perintah dengan URL klon setelah perintah. Sebagai contoh:

     ```
     git clone https://LiJuan@git.us-west-2.codecatalyst.aws/
     v1/ExampleCorp/MyExampleProject/MyExampleRepo
     ```

     Saat diminta kata sandi, tempel PAT yang Anda simpan sebelumnya.

     > **Note**
     >
     > Jika sistem operasi Anda menyediakan manajemen kredensi atau Anda telah menginstal sistem manajemen kredensi, Anda hanya perlu memberikan PAT sekali.

Jika tidak, Anda mungkin harus menyediakan PAT untuk setiap operasi Git. Sebagai praktik terbaik, pastikan bahwa sistem manajemen kredensional Anda menyimpan PAT Anda dengan aman. Jangan sertakan PAT sebagai bagian dari string URL klon.

- Untuk mengkloning repositori menggunakan IDE, ikuti dokumentasi untuk IDE Anda. Pilih opsi untuk mengkloning repositori Git dan berikan URL. Saat diminta kata sandi, berikan PAT.

Menghapus repositori sumber

Jika repositori sumber untuk CodeCatalyst proyek Amazon tidak lagi diperlukan, Anda dapat menghapusnya. Menghapus repositori sumber juga menghapus informasi proyek apa pun yang disimpan dalam repositori. Jika ada alur kerja yang bergantung pada repositori sumber, alur kerja tersebut akan dihapus dari daftar alur kerja proyek setelah repositori dihapus. Masalah yang mereferensikan repositori sumber tidak akan dihapus atau diubah, tetapi tautan apa pun ke repositori sumber yang ditambahkan ke masalah akan gagal setelah repositori dihapus.

Important

Menghapus repositori sumber tidak dapat dibatalkan. Setelah Anda menghapus repositori sumber, Anda tidak lagi dapat mengkloningnya, menarik data darinya, atau mendorong data ke sana. Menghapus repositori sumber tidak menghapus salinan lokal dari repositori itu (repo lokal). Untuk menghapus repo lokal, gunakan direktori komputer lokal dan alat manajemen file.

Note

Anda tidak dapat menghapus repositori tertaut di konsol. CodeCatalyst Untuk menghapus repositori tertaut, pilih tautan dalam daftar repositori untuk membuka repositori itu di layanan yang menghostingnya, lalu hapus. Untuk informasi selengkapnya, lihat dokumentasi untuk layanan yang menghosting repositori tertaut.
Untuk menghapus repositori tertaut dari proyek, lihat. Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst

Untuk menghapus repositori sumber

1. Arahkan ke proyek yang berisi repositori sumber yang ingin Anda hapus.
2. Pada halaman ringkasan untuk proyek Anda, pilih repositori yang Anda inginkan dari daftar, lalu pilih Lihat repositori. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber. Pilih nama repositori dari daftar repositori sumber untuk proyek.
3. Pada halaman beranda untuk repositori, pilih Lainnya, dan kemudian pilih Hapus repositori.
4. Tinjau cabang, permintaan tarik, dan informasi alur kerja terkait untuk membantu memastikan bahwa Anda tidak menghapus repositori yang masih digunakan atau memiliki pekerjaan yang belum selesai. Jika Anda ingin melanjutkan, ketik hapus, lalu pilih Hapus.

Mengatur kode sumber Anda bekerja dengan cabang di Amazon CodeCatalyst

Dalam Git, cabang adalah pointer atau referensi ke commit. Dalam pengembangan, mereka adalah cara yang nyaman untuk mengatur pekerjaan Anda. Anda dapat menggunakan cabang untuk memisahkan pekerjaan pada versi baru atau berbeda dari file tanpa mempengaruhi pekerjaan di cabang-cabang lain. Anda dapat menggunakan cabang untuk mengembangkan fitur baru, menyimpan versi spesifik proyek Anda, dan banyak lagi. Anda dapat mengonfigurasi aturan untuk cabang di repositori sumber untuk membatasi tindakan tertentu pada cabang ke peran tertentu dalam proyek tersebut.

Repositori sumber di Amazon CodeCatalyst memiliki konten dan cabang default terlepas dari bagaimana Anda membuatnya. Repositori tertaut mungkin tidak memiliki cabang atau konten default, tetapi tidak dapat digunakan CodeCatalyst sampai Anda menginisialisasi dan membuat cabang default. Saat Anda membuat proyek menggunakan cetak biru, CodeCatalyst buat repositori sumber untuk proyek tersebut yang menyertakan file README.md, kode sampel, definisi alur kerja, dan sumber daya lainnya. Saat Anda membuat repositori sumber tanpa menggunakan cetak biru, file README.md ditambahkan untuk Anda sebagai komit pertama, dan cabang default dibuat untuk Anda. Cabang default ini diberi nama main. Cabang default ini adalah cabang yang digunakan sebagai dasar atau cabang default dalam repositori lokal (repos) ketika pengguna mengkloning repositori.

> **Note**
>
> Anda tidak dapat menghapus cabang default. Cabang pertama yang dibuat untuk repositori sumber adalah cabang default untuk repositori itu. Selain itu, pencarian hanya menampilkan hasil dari cabang default. Anda tidak dapat mencari kode di cabang lain.

Membuat repositori CodeCatalyst juga membuat komit pertama, yang membuat cabang default dengan file README.md yang disertakan di dalamnya. Nama cabang default itu adalah main. Ini adalah nama cabang default yang digunakan dalam contoh dalam panduan ini.

Topik

Membuat cabang

Anda dapat menggunakan CodeCatalyst konsol untuk membuat cabang di CodeCatalyst repositori. Cabang yang Anda buat akan terlihat oleh pengguna lain saat berikutnya mereka menarik perubahan dari repositori.

> **Tip**
>
> Anda juga dapat membuat cabang sebagai bagian dari pembuatan Lingkungan Pengembang untuk mengerjakan kode Anda. Untuk informasi selengkapnya, lihat Membuat Lingkungan Dev.

Anda juga dapat menggunakan Git untuk membuat cabang. Untuk informasi selengkapnya, lihat Perintah Git umum untuk cabang.

Untuk membuat cabang (konsol)

1. Di CodeCatalyst konsol, arahkan ke proyek tempat repositori sumber Anda berada.
2. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
3. Pilih repositori tempat Anda ingin membuat cabang.
4. Pada halaman ikhtisar repositori, pilih Lainnya, lalu pilih Buat cabang.
5. Masukkan nama untuk cabang.
6. Pilih cabang untuk membuat cabang, lalu pilih Buat.

Mengelola cabang default untuk repositori

Anda dapat menentukan cabang mana yang akan digunakan sebagai cabang default di repositori sumber di Amazon. CodeCatalyst Semua repositori sumber CodeCatalyst memiliki konten dan cabang default terlepas dari bagaimana Anda membuatnya. Jika Anda menggunakan cetak biru untuk membuat proyek, cabang default di repositori sumber yang dibuat untuk proyek tersebut diberi nama main. Isi cabang default ditampilkan secara otomatis pada halaman ikhtisar untuk repositori itu.

Cabang default diperlakukan sedikit berbeda dari semua cabang lain dalam repositori sumber. Ini memiliki label khusus di sebelah namanya, Default. Cabang default adalah cabang yang digunakan sebagai basis atau cabang default di repositori lokal (repo) ketika pengguna mengkloning repositori ke komputer lokal dengan klien Git. Ini juga merupakan default yang digunakan saat membuat alur kerja untuk menyimpan file YAMAL alur kerja, dan untuk menyimpan informasi untuk masalah. Saat menggunakan search in CodeCatalyst, hanya cabang default dari repositori yang dicari. Karena cabang default sangat penting untuk begitu banyak aspek proyek, Anda tidak dapat menghapus cabang jika ditentukan sebagai cabang default. Namun, Anda dapat memilih untuk menggunakan cabang yang berbeda sebagai cabang default. Jika Anda melakukannya, aturan cabang apa pun yang diterapkan ke cabang default sebelumnya akan diterapkan secara otomatis ke cabang yang Anda tentukan sebagai cabang default.

> **Note**
>
> Anda harus memiliki peran administrator Proyek untuk mengubah cabang default untuk repositori sumber dalam CodeCatalyst proyek. Ini tidak berlaku untuk repositori tertaut.

Untuk melihat dan mengubah cabang default untuk repositori

1. Arahkan ke proyek tempat repositori Anda berada.
2. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber.

   Pilih repositori tempat Anda ingin melihat pengaturan, termasuk cabang default.
3. Pada halaman ikhtisar repositori, pilih Lainnya, lalu pilih Kelola pengaturan.
4. Di cabang Default, nama cabang yang ditentukan sebagai cabang default ditampilkan bersama dengan label bernama Default di sebelah nama. Label yang sama ini muncul di sebelah nama cabang dalam daftar cabang di Cabang.
5. Untuk mengubah cabang default, pilih Edit.

   > **Note**
   >
   > Anda harus memiliki peran administrator Proyek dalam proyek untuk mengubah cabang default.

6. Pilih nama cabang yang ingin Anda jadikan cabang default dari daftar drop-down lalu pilih Simpan.

Mengelola tindakan yang diizinkan untuk cabang dengan aturan cabang

Saat Anda membuat cabang, tindakan tertentu diizinkan untuk cabang tersebut berdasarkan izin untuk peran tersebut. Anda dapat mengubah tindakan apa yang diizinkan untuk cabang tertentu dengan mengonfigurasi aturan cabang. Aturan cabang didasarkan pada peran yang dimiliki pengguna dalam proyek Anda. Anda dapat memilih untuk membatasi beberapa tindakan yang telah ditentukan sebelumnya, seperti mendorong komit ke cabang, kepada pengguna dengan peran tertentu dalam proyek. Ini dapat membantu Anda melindungi cabang tertentu dalam proyek dengan membatasi peran mana yang diizinkan untuk melakukan tindakan tertentu. Misalnya, jika Anda mengonfigurasi aturan cabang untuk hanya mengizinkan pengguna dengan peran administrator Project untuk menggabungkan atau mendorong ke cabang tersebut, pengguna dengan peran lain dalam proyek tidak akan dapat membuat perubahan pada kode di cabang tersebut.

Anda harus mempertimbangkan dengan cermat semua implikasi pembuatan aturan untuk cabang. Misalnya, jika Anda memilih untuk membatasi push ke cabang ke pengguna dengan peran administrator Project, pengguna dengan peran Kontributor tidak akan dapat membuat atau mengedit

alur kerja di cabang tersebut, karena alur kerja YAMAL disimpan di cabang tersebut, dan pengguna tersebut tidak dapat melakukan dan mendorong perubahan ke YAMAL. Sebagai praktik terbaik, uji aturan cabang apa pun setelah Anda membuatnya untuk memastikan bahwa aturan tersebut tidak memiliki dampak yang tidak Anda inginkan. Anda juga dapat menggunakan aturan cabang bersama dengan aturan persetujuan untuk permintaan tarik. Untuk informasi selengkapnya, lihat Mengelola persyaratan untuk menggabungkan permintaan tarik dengan aturan persetujuan.

> **Note**
>
> Anda harus memiliki peran administrator Proyek untuk mengelola aturan cabang untuk repositori sumber dalam CodeCatalyst proyek. Anda tidak dapat membuat aturan cabang untuk repositori tertaut.
> Anda hanya dapat membuat aturan cabang yang lebih ketat daripada izin default untuk peran tersebut. Anda tidak dapat membuat aturan cabang yang lebih permisif daripada yang diizinkan oleh peran pengguna dalam proyek. Misalnya, Anda tidak dapat membuat aturan cabang yang memungkinkan pengguna dengan peran Reviewer untuk mendorong ke cabang.

Aturan cabang yang diterapkan ke cabang default repositori sumber Anda akan berperilaku sedikit berbeda dari aturan cabang yang diterapkan ke cabang lain. Aturan apa pun yang diterapkan ke cabang default akan diterapkan secara otomatis ke cabang mana pun yang Anda tentukan sebagai cabang default. Cabang yang sebelumnya ditetapkan sebagai cabang default akan tetap mempertahankan aturan yang diterapkan padanya, kecuali bahwa ia tidak lagi memiliki perlindungan terhadap penghapusan. Perlindungan itu hanya diterapkan ke cabang default saat ini.

Aturan cabang memiliki dua negara bagian, Standar dan Kustom. Standar menunjukkan bahwa tindakan yang diizinkan di cabang adalah tindakan yang cocok dengan izin untuk peran yang dimiliki pengguna CodeCatalyst untuk tindakan cabang. Untuk mempelajari lebih lanjut tentang peran apa yang memiliki izin, lihatMemberikan akses dengan peran pengguna. Kustom menunjukkan bahwa satu atau beberapa tindakan cabang memiliki tindakan yang memiliki daftar peran tertentu yang diizinkan untuk melakukan tindakan yang berbeda dari izin default yang diberikan oleh roe pengguna dalam proyek.

> **Note**
>
> Jika Anda membuat aturan cabang untuk membatasi satu atau beberapa tindakan untuk cabang, tindakan Hapus cabang secara otomatis diatur untuk hanya mengizinkan pengguna dengan peran administrator Project untuk menghapus cabang tersebut.

Tabel berikut mencantumkan tindakan dan pengaturan default untuk peran yang diizinkan untuk melakukan tindakan ini di cabang.

Tindakan dan peran cabang

Tindakan cabang	Peran diizinkan untuk melakukan tindakan ini ketika tidak ada aturan cabang yang diterapkan
Gabungkan ke cabang (ini termasuk menggabungkan permintaan tarik ke cabang)	Administrator proyek, Kontributor
Dorong ke cabang	Administrator proyek, Kontributor
Hapus cabang	Administrator proyek, Kontributor
Hapus cabang (cabang default)	Tidak diizinkan

Anda tidak dapat menghapus aturan cabang, tetapi Anda dapat memperbaruinya untuk mengizinkan tindakan dari semua peran yang diizinkan untuk melakukan tindakan ini di cabang, yang secara efektif menghapus aturan tersebut.

> **Note**
>
> Anda harus memiliki peran administrator Project untuk mengonfigurasi aturan cabang untuk repositori sumber dalam CodeCatalyst proyek. Ini tidak berlaku untuk repositori tertaut. Repositori tertaut tidak mendukung aturan cabang di. CodeCatalyst

Untuk melihat dan mengedit aturan cabang untuk repositori

1. Arahkan ke proyek tempat repositori Anda berada.

2. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber.

   Pilih repositori tempat Anda ingin melihat aturan cabang.

3. Pada halaman ikhtisar repositori, pilih Cabang.

4. Di kolom Aturan cabang, lihat status aturan untuk setiap cabang repositori. Standar menunjukkan bahwa aturan untuk tindakan cabang adalah aturan default untuk setiap cabang yang dibuat dalam repositori sumber dan cocok dengan izin yang diberikan untuk peran tersebut dalam proyek. Kustom menunjukkan bahwa satu atau beberapa tindakan cabang memiliki aturan yang membatasi satu atau beberapa tindakan yang diizinkan untuk cabang tersebut ke serangkaian peran yang berbeda.

   Untuk melihat spesifikasi aturan cabang untuk cabang, pilih kata Standar atau Kustom di sebelah cabang yang ingin Anda tinjau.

5. Untuk membuat atau mengubah aturan cabang, pilih Kelola pengaturan. Pada halaman pengaturan untuk repositori sumber, dalam aturan cabang, pilih Edit.

6. Di Cabang, pilih nama cabang yang ingin Anda konfigurasikan aturannya dari daftar drop-down. Untuk setiap jenis tindakan yang diizinkan, pilih peran yang ingin Anda izinkan untuk melakukan tindakan tersebut dari daftar drop-down, lalu pilih Simpan.

Perintah Git untuk cabang

Anda dapat menggunakan Git untuk membuat, mengelola, dan menghapus cabang di klon repositori sumber yang Anda miliki di komputer Anda (repo lokal Anda) atau di Lingkungan Dev Anda, lalu komit dan dorong perubahan Anda ke repositori CodeCatalyst sumber Anda (repositori jarak jauh). Sebagai contoh:

Perintah Git umum untuk cabang

Mendaftar semua cabang di repo lokal dengan tanda bintang (*) yang ditampilkan di sebelah cabang Anda saat ini.	`git branch`
Menarik informasi tentang semua cabang yang ada di repositori jarak jauh ke repo lokal.	`git fetch`

Mendaftar semua cabang di repo lokal dan cabang pelacakan jarak jauh di repo lokal.	`git branch -a`
Mendaftar hanya cabang pelacakan jarak jauh di repo lokal.	`git branch -r`
Membuat cabang di repo lokal menggunakan nama cabang yang ditentukan. Cabang ini tidak akan muncul di repositori jarak jauh sampai Anda melakukan dan mendorong perubahan.	`git branch branch-name`
Membuat cabang di repo lokal menggunakan nama cabang yang ditentukan, dan kemudian beralih ke sana.	`git checkout -b branch-name`
Beralih ke cabang lain di repo lokal dengan menggunakan nama cabang yang ditentukan.	`git checkout other-branch-name`
Mendorong cabang dari repo lokal ke repositori jarak jauh menggunakan nama panggilan yang ditentukan repo lokal untuk repositori jarak jauh dan nama cabang yang ditentukan. Juga mengatur informasi pelacakan hulu untuk cabang di repo lokal.	`git push -u remote-name branch-name`
Menggabungkan perubahan dari cabang lain di repo lokal untuk cabang saat ini di repo lokal.	`git merge from-other-branch-name`
Menghapus cabang di repo lokal kecuali cabang tersebut berisi pekerjaan yang belum digabung.	`git branch -d branch-name`

Menghapus cabang di repositori jarak jauh menggunakan nama panggilan yang ditentukan yang dimiliki repo lokal untuk repositori jarak jauh dan nama cabang yang ditentukan. (Perhatikan penggunaan titik dua (`:`).) Atau, tentukan `--delete` sebagai bagian dari perintah.	`git push` *`remote-name`* `:`*`branch-name`* `git push` *`remote-name`* `--delete` *`branch-name`*

Untuk informasi selengkapnya, lihat dokumentasi Git Anda.

Melihat cabang dan detail

Anda dapat melihat informasi tentang cabang jarak jauh di Amazon CodeCatalyst, termasuk spesifikasi file, folder, dan komit terbaru untuk cabang tertentu, di CodeCatalyst konsol Amazon. Anda juga dapat menggunakan perintah Git dan sistem operasi lokal Anda untuk melihat informasi ini untuk cabang jarak jauh dan lokal.

Untuk melihat cabang (konsol)

1. Di CodeCatalyst konsol, arahkan ke proyek yang berisi repositori sumber tempat Anda ingin melihat cabang. Pilih Kode, pilih repositori sumber, dan kemudian pilih repositori sumber.
2. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber.

   Pilih repositori tempat Anda ingin melihat cabang.
3. Cabang default repositori ditampilkan. Anda dapat melihat daftar file dan folder di cabang, informasi tentang komit terbaru, dan isi file README.md, jika ada di cabang. Untuk melihat informasi untuk cabang yang berbeda, pilih dari daftar drop-down cabang untuk repositori.
4. Untuk melihat semua cabang untuk repositori, pilih Lihat semua. Halaman Cabang menampilkan informasi tentang nama, komit terbaru, dan aturan untuk setiap cabang.

Untuk informasi tentang cara menggunakan Git dan sistem operasi Anda untuk melihat cabang dan detail, lihat Perintah Git Umum untuk cabang, dokumentasi Git Anda, dan dokumentasi sistem operasi Anda.

Menghapus cabang

Jika Anda tidak lagi membutuhkan cabang, Anda dapat menghapusnya. Misalnya, jika Anda telah menggabungkan cabang dengan perubahan fitur ke cabang default dan fitur tersebut telah dirilis, Anda mungkin ingin menghapus cabang fitur asli, karena perubahan sudah menjadi bagian dari cabang default. Menjaga jumlah cabang tetap rendah dapat membantu pengguna menemukan cabang yang berisi perubahan yang ingin mereka kerjakan. Saat Anda menghapus cabang, salinan cabang itu tetap berada di klon repositori di komputer lokal sampai pengguna menarik dan menyinkronkan perubahan tersebut.

Untuk menghapus cabang (konsol)

1. Arahkan ke proyek tempat repositori Anda berada.
2. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber.

   Pilih repositori tempat Anda ingin menghapus cabang.
3. Pada halaman ikhtisar repositori, pilih pemilih drop-down di sebelah nama cabang, lalu pilih Lihat semua.
4. Pilih cabang yang ingin Anda hapus lalu pilih Hapus cabang.

   > **Note**
   >
   > Anda tidak dapat menghapus cabang default untuk repositori.

5. Sebuah kotak dialog konfirmasi kemudian muncul. Ini menunjukkan repositori, jumlah permintaan tarik terbuka, dan jumlah alur kerja yang terkait dengan cabang.
6. Untuk mengonfirmasi penghapusan cabang, ketik hapus ke dalam kotak teks, lalu pilih Hapus.

Anda juga dapat menggunakan Git untuk menghapus cabang. Untuk informasi selengkapnya, lihat Perintah Git umum untuk cabang.

Mengelola file kode sumber di Amazon CodeCatalyst

Di Amazon CodeCatalyst, file adalah informasi mandiri yang dikendalikan versi yang tersedia untuk Anda dan pengguna lain dari repositori sumber dan cabang tempat file disimpan. Anda dapat

mengatur file repositori Anda dengan struktur direktori. CodeCatalystsecara otomatis melacak setiap perubahan yang dilakukan ke file. Anda dapat menyimpan versi file yang berbeda di cabang repositori yang berbeda.

Untuk menambah atau mengedit beberapa file dalam repositori sumber, Anda dapat menggunakan klien Git, Lingkungan Dev, atau lingkungan pengembangan terintegrasi (IDE). Untuk menambah atau mengedit satu file, Anda dapat menggunakan CodeCatalyst konsol.

Topik

Membuat atau menambahkan file

Untuk membuat dan menambahkan file ke repositori sumber, Anda dapat menggunakan CodeCatalyst konsol Amazon, Lingkungan Dev, lingkungan pengembangan terintegrasi (IDE) yang terhubung, atau klien Git. CodeCatalyst Konsol menyertakan editor kode untuk membuat file. Editor ini adalah cara mudah untuk membuat atau mengedit file sederhana, seperti file README.md, di cabang di repositori. Saat mengerjakan lebih dari satu file, pertimbangkan untuk membuat Lingkungan Dev.

Untuk membuat Lingkungan Dev dari repositori sumber

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
3. Pilih repositori sumber tempat Anda ingin mengerjakan kode.
4. Pilih Buat Lingkungan Pengembang.
5. Pilih IDE yang didukung dari menu tarik-turun. Untuk informasi selengkapnya, lihat Lingkungan pengembangan terintegrasi yang didukung untuk Lingkungan Dev.
6. Lakukan salah satu hal berikut ini:
   - Pilih Bekerja di cabang yang ada, lalu pilih cabang dari menu drop-down cabang yang ada.

- Pilih Bekerja di cabang baru, masukkan nama cabang ke bidang Nama cabang, dan pilih cabang untuk membuat cabang baru dari menu drop-down Buat cabang dari.

7. Secara opsional tambahkan nama untuk Lingkungan Dev atau edit konfigurasinya.
8. Pilih Buat.

Untuk membuat file di CodeCatalyst konsol

1. Arahkan ke proyek tempat Anda ingin membuat file. Untuk informasi selengkapnya tentang cara menavigasi ke repositori, lihat. Melihat repositori sumber
2. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber.

   Pilih repositori tempat Anda ingin membuat file.
3. (Opsional) Pilih cabang tempat Anda ingin membuat file, jika Anda ingin membuat file di cabang yang berbeda dari cabang default.
4. Pilih Buat file.
5. Masukkan nama file di Nama file. Tambahkan konten file di editor.

   > **Tip**
   >
   > Jika Anda ingin membuat file di sub-folder atau subdirektori dari root cabang, sertakan struktur itu sebagai bagian dari nama file.

   Ketika Anda puas dengan perubahan Anda, pilih Komit.
6. Dalam nama File, tinjau nama file dan buat perubahan apa pun yang mungkin Anda inginkan. Secara opsional pilih cabang tempat Anda ingin membuat file dari daftar cabang yang tersedia di Cabang. Dalam pesan Komit, secara opsional masukkan deskripsi singkat namun informatif tentang mengapa Anda melakukan perubahan ini. Ini akan ditampilkan sebagai informasi komit dasar untuk komit yang menambahkan file ke repositori sumber.
7. Pilih Komit untuk melakukan dan mendorong file ke repositori sumber.

Anda juga dapat menambahkan file ke repositori sumber dengan mengkloning ke komputer lokal Anda dan menggunakan klien Git atau lingkungan pengembangan terintegrasi terhubung (IDE) untuk mendorong file dan perubahan Anda.

> **Note**
>
> Jika Anda ingin menambahkan submodul Git, Anda harus menggunakan klien Git atau Lingkungan Dev dan menjalankan perintah. git submodule add Anda tidak dapat menambahkan atau melihat submodul Git di CodeCatalyst konsol atau melihat perbedaan submodul Git dalam permintaan tarik. Untuk informasi selengkapnya tentang submodul Git, lihat dokumentasi Git.

Untuk menambahkan file menggunakan klien Git atau lingkungan pengembangan terintegrasi yang terhubung (IDE)

1. Kloning repositori sumber Anda ke komputer lokal Anda. Untuk informasi selengkapnya, lihat Mengkloning repositori sumber.
2. Buat file di repo lokal Anda atau salin file ke repo lokal Anda.
3. Buat dan dorong komit dengan melakukan salah satu hal berikut:

   - Jika Anda menggunakan klien Git, di terminal atau baris perintah, jalankan git add perintah, tentukan nama file yang ingin Anda tambahkan. Atau, untuk menambahkan semua file yang ditambahkan atau diubah, jalankan git add perintah diikuti oleh periode tunggal atau ganda untuk menunjukkan apakah Anda ingin memasukkan semua perubahan pada tingkat direktori saat ini (periode tunggal) atau semua perubahan dalam direktori saat ini dan semua subdirektori (periode ganda). Untuk melakukan perubahan, jalankan git commit -m perintah dan berikan pesan komit. Untuk mendorong perubahan Anda ke repositori sumber CodeCatalyst, jalankan. git push Untuk informasi selengkapnya tentang perintah Git, lihat dokumentasi Git Anda danPerintah Git untuk cabang.
   - Jika Anda menggunakan Dev Environment atau IDE, buat file dan tambahkan file di IDE, lalu komit dan dorong perubahan Anda. Untuk informasi selengkapnya, lihat Menulis dan memodifikasi kode dengan Dev Environments di CodeCatalyst atau lihat dokumentasi IDE Anda.

Melihat file

Anda dapat melihat file di repositori sumber Anda di konsol Amazon CodeCatalyst . Anda dapat melihat file di cabang default dan di cabang lainnya. Isi file dapat bervariasi tergantung pada cabang yang Anda pilih untuk dilihat.

Untuk melihat file di CodeCatalyst konsol

1. Arahkan ke proyek tempat Anda ingin melihat file. Untuk informasi selengkapnya, lihat Melihat repositori sumber.
2. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber.

   Pilih repositori tempat Anda ingin melihat file.
3. Daftar file dan folder ditampilkan untuk cabang default. File ditunjukkan oleh ikon paper, sedangkan folder ditunjukkan oleh ikon folder.
4. Lakukan salah satu langkah berikut ini:
   - Untuk melihat file dan folder di cabang yang berbeda, pilih dari daftar cabang.
   - Untuk memperluas folder, pilih dari daftar.
5. Untuk melihat isi file tertentu, pilih dari daftar. Isi file akan ditampilkan di cabang. Untuk melihat isi file di cabang yang berbeda, pilih cabang yang Anda inginkan dari pemilih cabang.

> **Tip**
>
> Saat melihat konten file, Anda dapat memilih file tambahan untuk dilihat dari Lihat file. Untuk mengedit file, pilih Edit.

Anda dapat melihat beberapa file di konsol. Anda juga dapat melihat file yang telah Anda kloning ke komputer lokal Anda menggunakan klien Git atau lingkungan pengembangan terintegrasi (IDE). Untuk informasi selengkapnya, lihat dokumentasi untuk klien Git atau IDE Anda.

> **Note**
>
> Anda tidak dapat melihat submodul Git di CodeCatalyst konsol. Untuk informasi selengkapnya tentang submodul Git, lihat dokumentasi Git.

Melihat riwayat perubahan pada file

Anda dapat melihat riwayat perubahan pada file di repositori sumber Anda di konsol Amazon CodeCatalyst . Ini dapat membantu Anda memahami perubahan yang dibuat pada file oleh berbagai

komit ke cabang tempat Anda memilih untuk melihat riwayat file. Misalnya, jika Anda melihat riwayat perubahan pada **`readme.md`** file di **`main`** cabang repositori sumber, Anda akan melihat daftar komit yang menyertakan perubahan pada file itu di cabang itu.

> **Note**
>
> Anda tidak dapat melihat riwayat file di repositori tertaut di konsol. CodeCatalyst

Untuk melihat riwayat file di CodeCatalyst konsol

1. Arahkan ke proyek tempat Anda ingin melihat riwayat file. Untuk informasi selengkapnya, lihat Melihat repositori sumber.
2. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
3. Pilih repositori tempat Anda ingin melihat riwayat file. Pilih cabang tempat Anda ingin melihat riwayat file, lalu pilih file dari daftar. Pilih Lihat riwayat.
4. Tinjau daftar komit yang menyertakan perubahan pada file ini di cabang yang ditentukan. Untuk melihat detail perubahan yang disertakan dalam komit tertentu, pilih pesan komit untuk komit tersebut dalam daftar. Perbedaan antara komit itu dan komit induknya ditampilkan.
5. Untuk meninjau riwayat perubahan pada file di cabang lain, gunakan pemilih cabang untuk mengubah tampilan ke cabang itu, pilih file dari daftar file, lalu pilih Lihat riwayat.

> **Note**
>
> Anda tidak dapat melihat riwayat perubahan pada submodul Git di CodeCatalyst konsol. Untuk informasi selengkapnya tentang submodul Git, lihat dokumentasi Git.

Mengedit file

Anda dapat mengedit file individual di CodeCatalyst konsol Amazon. Untuk mengedit beberapa file sekaligus, buat Lingkungan Dev atau kloning repositori dan buat perubahan Anda menggunakan klien Git atau lingkungan pengembangan terintegrasi (IDE). Untuk informasi selengkapnya, lihat Menulis dan memodifikasi kode dengan Dev Environments di CodeCatalyst atau Mengkloning repositori sumber.

Untuk mengedit file di CodeCatalyst konsol

1. Arahkan ke proyek tempat Anda ingin mengedit file. Untuk informasi selengkapnya tentang cara menavigasi ke repositori, lihat. Melihat repositori sumber
2. Pilih repositori tempat Anda ingin mengedit file. Pilih Lihat cabang dan kemudian pilih cabang yang ingin Anda kerjakan. Pilih file dari daftar file dan folder di cabang itu.

   Isi file ditampilkan.
3. Pilih Edit.
4. Di editor, edit konten file dan kemudian pilih Komit. Secara opsional, dalam perubahan Komit, tambahkan informasi selengkapnya tentang perubahan dalam pesan Komit. Ketika Anda puas dengan perubahan Anda, pilih Komit.

Mengganti nama atau menghapus file

Anda dapat mengganti nama atau menghapus file di Lingkungan Pengembang, secara lokal di komputer Anda, atau dalam lingkungan pengembangan terintegrasi (IDE). Setelah Anda mengganti nama atau menghapus file, komit dan dorong perubahan tersebut ke repositori sumber. Anda tidak dapat mengganti nama atau menghapus file di CodeCatalyst konsol Amazon.

Meninjau kode dengan permintaan tarik di Amazon CodeCatalyst

Permintaan tarik adalah cara utama Anda dan anggota proyek lainnya dapat meninjau, mengomentari, dan menggabungkan perubahan kode dari satu cabang ke cabang lainnya. Anda dapat menggunakan permintaan tarik untuk meninjau perubahan kode secara kolaboratif untuk perubahan kecil atau perbaikan, penambahan fitur utama, atau versi baru perangkat lunak yang dirilis. Jika Anda menggunakan masalah untuk melacak pekerjaan pada proyek Anda, Anda dapat menautkan masalah tertentu ke permintaan tarik untuk membantu Anda melacak masalah apa yang sedang ditangani oleh perubahan kode dalam permintaan tarik. Saat Anda membuat, memperbarui, mengomentari, menggabungkan, atau menutup permintaan tarik, email secara otomatis dikirim ke pembuat permintaan tarik serta pengulas wajib atau opsional untuk permintaan tarik.

> **Tip**
>
> Anda dapat mengonfigurasi peristiwa permintaan tarik apa yang akan Anda terima email sebagai bagian dari profil Anda. Untuk informasi selengkapnya, lihat Mengelola notifikasi di Amazon CodeCatalyst.

Permintaan tarik memerlukan dua cabang dalam repositori sumber: cabang sumber yang berisi kode yang ingin Anda tinjau, dan cabang tujuan, tempat Anda ingin menggabungkan kode yang ditinjau. Cabang sumber berisi commit SETELAH, yang merupakan komit yang berisi perubahan yang ingin Anda gabungkan ke cabang tujuan. Cabang tujuan berisi commit SEBELUM, yang mewakili keadaan kode sebelum cabang permintaan pull digabung ke cabang tujuan.

> **Note**
>
> Saat Anda membuat permintaan tarik, perbedaan yang ditampilkan adalah perbedaan antara ujung cabang sumber dan ujung cabang tujuan. Setelah Anda membuat permintaan tarik, perbedaan yang ditampilkan adalah antara revisi permintaan tarik yang Anda pilih dan komit yang merupakan ujung cabang tujuan saat Anda membuat permintaan tarik. Untuk informasi selengkapnya tentang perbedaan dan penggabungan basis di Git, lihat git-merge-basedi dokumentasi Git.

Sementara permintaan tarik dibuat untuk repositori sumber dan cabang tertentu, Anda dapat membuat, melihat, meninjau, dan menutupnya sebagai bagian dari bekerja dengan proyek Anda. Anda tidak perlu melihat repositori sumber untuk melihat dan bekerja dengan permintaan tarik. Status permintaan tarik diatur ke Buka saat Anda membuatnya. Permintaan tarik tetap terbuka hingga Anda menggabungkannya di CodeCatalyst konsol, yang mengubah status menjadi Gabungan, atau menutupnya, yang mengubah status menjadi Ditutup.

Ketika kode Anda telah ditinjau, Anda dapat mengubah status permintaan tarik dengan salah satu dari beberapa cara:

- Gabungkan permintaan tarik di CodeCatalyst konsol. Kode di cabang sumber permintaan tarik akan digabungkan ke cabang tujuan. Status pull request akan berubah menjadi Merged. Tidak dapat diubah kembali ke Open.
- Gabungkan cabang secara lokal dan dorong perubahan Anda, lalu tutup permintaan tarik di CodeCatalyst konsol.

- Gunakan CodeCatalyst konsol untuk menutup permintaan tarik tanpa menggabungkan. Ini akan mengubah status menjadi Closed, dan itu tidak akan menggabungkan kode dari cabang sumber ke cabang tujuan.

Sebelum Anda membuat permintaan pull:

- Komit dan dorong perubahan kode yang ingin Anda tinjau ke cabang (cabang sumber).
- Siapkan notifikasi untuk proyek Anda, sehingga pengguna lain dapat diberi tahu tentang alur kerja apa pun yang berjalan saat Anda membuat permintaan tarik. (Langkah ini opsional tetapi disarankan.)

Topik

Membuat permintaan pull

Membuat pull request akan membantu pengguna lain melihat dan meninjau perubahan kode Anda sebelum Anda menggabungkan mereka ke cabang lain. Pertama, Anda harus membuat sebuah cabang untuk perubahan kode Anda. Hal ini disebut sebagai cabang sumber untuk sebuah pull request. Setelah Anda melakukan dan mendorong perubahan ke repositori, Anda dapat membuat permintaan tarik yang membandingkan konten cabang sumber dengan konten cabang tujuan.

Anda dapat membuat permintaan tarik di CodeCatalyst konsol Amazon dari cabang tertentu, dari halaman permintaan tarik, atau dari ikhtisar proyek. Membuat permintaan tarik dari cabang tertentu secara otomatis menyediakan nama repositori dan cabang sumber pada halaman pembuatan permintaan tarik. Saat Anda membuat permintaan tarik, Anda akan secara otomatis menerima email tentang pembaruan apa pun pada permintaan tarik, serta saat permintaan tarik digabungkan atau ditutup.

> **Note**
>
> Saat Anda membuat permintaan tarik, perbedaan yang ditampilkan adalah perbedaan antara ujung cabang sumber dan ujung cabang tujuan. Setelah permintaan tarik dibuat, perbedaan yang ditampilkan adalah antara revisi permintaan tarik yang Anda pilih dan komit yang merupakan ujung cabang tujuan saat Anda membuat permintaan tarik. Untuk informasi selengkapnya tentang perbedaan dan penggabungan basis di Git, lihat git-merge-basedi dokumentasi Git.

Anda dapat menggunakan fitur Tulis deskripsi untuk saya saat membuat permintaan tarik agar Amazon Q secara otomatis membuat deskripsi perubahan yang terkandung dalam permintaan tarik. Saat Anda memilih opsi ini, Amazon Q menganalisis perbedaan antara cabang sumber yang berisi perubahan kode dan cabang tujuan tempat Anda ingin menggabungkan perubahan ini. Ini kemudian menciptakan ringkasan tentang apa perubahan itu, serta interpretasi terbaiknya tentang maksud dan efek dari perubahan tersebut. Fitur ini hanya tersedia di Wilayah AS Barat (Oregon).

> **Note**
>
> Didukung oleh Amazon Bedrock: AWS mengimplementasikan deteksi penyalahgunaan otomatis. Karena deskripsi Tulis untuk saya dan Buat fitur ringkasan konten dibangun di Amazon Bedrock, pengguna dapat memanfaatkan sepenuhnya kontrol yang diterapkan di Amazon Bedrock untuk menegakkan keselamatan, keamanan, dan penggunaan kecerdasan buatan (AI) yang bertanggung jawab.

Untuk membuat permintaan tarik

1. Arahkan ke proyek Anda.
2. Lakukan salah satu hal berikut ini:
   - Di panel navigasi, pilih Kode, pilih Tarik permintaan, lalu pilih Buat permintaan tarik.
   - Pada halaman beranda repositori, pilih Lainnya, lalu pilih Buat permintaan tarik.
   - Pada halaman proyek, pilih Buat permintaan tarik.
3. Di repositori Sumber, pastikan bahwa repositori sumber yang ditentukan adalah yang berisi kode yang dikomit. Opsi ini hanya muncul jika Anda tidak membuat permintaan tarik dari halaman utama repositori.

4. Di cabang Tujuan, pilih cabang untuk menggabungkan kode setelah ditinjau.
5. Di cabang Sumber, pilih cabang yang berisi kode komit.
6. Dalam judul permintaan tarik, masukkan judul yang membantu pengguna lain memahami apa yang perlu ditinjau dan alasannya.
7. (Opsional) Dalam deskripsi permintaan Tarik, berikan informasi seperti tautan ke masalah atau deskripsi perubahan Anda.

> **Tip**
>
> Anda dapat memilih Tulis deskripsi agar saya CodeCatalyst secara otomatis menghasilkan deskripsi tentang perubahan yang terkandung dalam permintaan tarik. Anda dapat membuat perubahan pada deskripsi yang dibuat secara otomatis setelah Anda menambahkannya ke permintaan tarik.
> Fungsionalitas ini mengharuskan fitur AI generatif diaktifkan untuk ruang tersebut. Untuk informasi selengkapnya, lihat Mengelola fitur AI generatif.

8. (Opsional) Dalam Masalah, pilih Masalah tautan, lalu pilih masalah dari daftar atau masukkan ID-nya. Untuk memutuskan tautan masalah, pilih ikon batalkan tautan.
9. (Opsional) Di Reviewer wajib, pilih Tambahkan pengulas yang diperlukan. Pilih dari daftar anggota proyek untuk menambahkannya. Peninjau yang diperlukan harus menyetujui perubahan sebelum permintaan tarik dapat digabungkan ke cabang tujuan.

> **Note**
>
> Anda tidak dapat menambahkan reviewer sebagai reviewer wajib dan reviewer opsional. Anda tidak dapat menambahkan diri Anda sebagai reviewer.

10. (Opsional) Di pengulas opsional, pilih Tambahkan pengulas opsional. Pilih dari daftar anggota proyek untuk menambahkannya. Peninjau opsional tidak harus menyetujui perubahan sebagai persyaratan sebelum permintaan tarik dapat digabungkan ke cabang tujuan.
11. Tinjau perbedaan antara cabang. Perbedaan yang ditampilkan dalam permintaan tarik adalah perubahan antara revisi di cabang sumber dan basis gabungan, yang merupakan komit kepala cabang tujuan pada saat permintaan tarik dibuat. Jika tidak ada perubahan yang ditampilkan, cabang mungkin identik, atau Anda mungkin telah memilih cabang yang sama untuk sumber dan tujuan.

12. Ketika Anda puas bahwa permintaan tarik berisi kode dan perubahan yang ingin Anda tinjau, pilih Buat.

> **Note**
>
> Setelah Anda membuat permintaan tarik, Anda dapat menambahkan komentar. Komentar dapat ditambahkan ke permintaan tarik atau ke baris individual dalam file serta permintaan tarik keseluruhan. Anda dapat menambahkan tautan ke sumber daya, seperti file, dengan menggunakan tanda @ diikuti dengan nama file.

Untuk membuat permintaan tarik dari cabang

1. Arahkan ke proyek tempat Anda ingin membuat permintaan tarik.
2. Di panel navigasi, pilih repositori Sumber, lalu pilih repositori yang berisi cabang tempat Anda memiliki perubahan kode untuk ditinjau.
3. Pilih panah drop-down di sebelah nama cabang default, lalu pilih cabang yang Anda inginkan dari daftar. Untuk melihat semua cabang untuk repositori, pilih Lihat semua.
4. Pilih Lainnya, lalu pilih Buat permintaan tarik.
5. Repositori dan cabang sumber telah dipilih sebelumnya untuk Anda. Di cabang Tujuan, pilih cabang tempat Anda akan menggabungkan kode setelah ditinjau. Dalam judul permintaan Tarik, masukkan judul yang akan membantu pengguna proyek lain memahami apa yang harus ditinjau dan alasannya. Secara opsional, berikan informasi lebih lanjut dalam deskripsi permintaan Tarik, seperti menempelkan tautan ke masalah terkait CodeCatalyst, atau menambahkan deskripsi perubahan yang Anda buat.

> **Note**
>
> Alur kerja yang dikonfigurasi untuk menjalankan acara pembuatan permintaan tarik akan berjalan setelah permintaan tarik dibuat, jika cabang tujuan untuk permintaan tarik cocok dengan salah satu cabang yang ditentukan dalam alur kerja.

6. Tinjau perbedaan antara cabang. Jika tidak ada perubahan yang ditampilkan, cabang mungkin identik, atau Anda mungkin telah memilih cabang yang sama untuk sumber dan tujuan.
7. (Opsional) Dalam Masalah, pilih Masalah tautan, lalu pilih masalah dari daftar atau masukkan ID-nya. Untuk memutuskan tautan masalah, pilih ikon batalkan tautan.

8. (Opsional) Di Reviewer wajib, pilih Tambahkan pengulas yang diperlukan. Pilih dari daftar anggota proyek untuk menambahkannya. Peninjau yang diperlukan harus menyetujui perubahan sebelum permintaan tarik dapat digabungkan ke cabang tujuan.

   > **Note**
   >
   > Anda tidak dapat menambahkan pengulas sebagai wajib dan opsional. Anda tidak dapat menambahkan diri Anda sebagai reviewer.

9. (Opsional) Di pengulas opsional, pilih Tambahkan pengulas opsional. Pilih dari daftar anggota proyek untuk menambahkannya. Peninjau opsional tidak harus menyetujui perubahan sebelum permintaan tarik dapat digabungkan ke cabang tujuan.
10. Jika Anda puas bahwa permintaan tarik berisi perubahan yang ingin Anda tinjau dan menyertakan pengulas yang diperlukan, pilih Buat.

Jika Anda memiliki alur kerja yang dikonfigurasi untuk menjalankan di mana cabang cocok dengan cabang tujuan dalam permintaan tarik, Anda akan melihat informasi tentang alur kerja tersebut berjalan di Ikhtisar di area detail permintaan tarik setelah permintaan tarik dibuat. Untuk informasi selengkapnya, lihat Menambahkan pemicu push, pull, atau schedule.

Melihat permintaan tarik

Anda dapat melihat permintaan tarik untuk proyek di CodeCatalyst konsol Amazon. Halaman ringkasan proyek menampilkan semua permintaan tarik terbuka untuk sebuah proyek. Untuk melihat semua permintaan tarik terlepas dari statusnya, navigasikan ke halaman permintaan tarik untuk proyek Anda. Saat melihat permintaan tarik, Anda dapat memilih untuk memiliki ringkasan dari semua komentar yang tersisa pada perubahan permintaan tarik yang dibuat untuk Anda.

> **Note**
>
> Didukung oleh Amazon Bedrock: AWS mengimplementasikan deteksi penyalahgunaan otomatis. Karena deskripsi Tulis untuk saya dan Buat fitur ringkasan konten dibangun di Amazon Bedrock, pengguna dapat memanfaatkan sepenuhnya kontrol yang diterapkan di Amazon Bedrock untuk menegakkan keselamatan, keamanan, dan penggunaan kecerdasan buatan (AI) yang bertanggung jawab.

Untuk melihat permintaan tarik terbuka

1. Arahkan ke proyek tempat Anda ingin melihat permintaan tarik.
2. Pada halaman proyek, permintaan tarik terbuka ditampilkan, termasuk informasi tentang siapa yang membuat permintaan tarik, repositori apa yang berisi cabang untuk permintaan tarik, dan tanggal permintaan tarik dibuat. Anda dapat memfilter tampilan permintaan tarik terbuka berdasarkan repositori sumber.
3. Untuk melihat semua permintaan tarik, pilih Lihat semua. Anda dapat menggunakan penyeleksi untuk memilih di antara opsi. Misalnya, untuk melihat semua permintaan tarik, pilih Status apa pun dan Penulis apa pun.

   Atau, di panel navigasi, pilih Kode, lalu pilih Tarik permintaan, lalu gunakan pemilih untuk mempersempit tampilan Anda.
4. Pada halaman Permintaan tarik, Anda dapat mengurutkan permintaan tarik berdasarkan ID, judul, status, dan lainnya. Untuk menyesuaikan informasi apa dan berapa banyak informasi yang ditampilkan di halaman permintaan tarik, pilih ikon roda gigi.
5. Untuk melihat permintaan tarik tertentu, pilih dari daftar.
6. Untuk melihat status alur kerja yang terkait dengan permintaan tarik ini, jika ada, pilih Ikhtisar dan tinjau informasi di area detail permintaan tarik pada permintaan tarik di bawah Alur kerja berjalan.

   Proses alur kerja akan terjadi jika alur kerja dikonfigurasi dengan peristiwa pembuatan atau revisi permintaan tarik, dan jika persyaratan cabang tujuan dalam alur kerja cocok dengan cabang tujuan yang ditentukan dalam permintaan tarik. Untuk informasi selengkapnya, lihat Menambahkan pemicu push, pull, atau schedule.
7. Untuk melihat masalah terkait, jika ada, pilih Ikhtisar dan tinjau informasi di detail permintaan Tarik di bawah Masalah. Jika Anda ingin melihat masalah tertaut, pilih ID-nya dari daftar.
8. (Opsional) Untuk membuat ringkasan komentar yang tersisa pada perubahan kode dalam revisi permintaan tarik, pilih Buat ringkasan konten. Ringkasan tidak akan menyertakan komentar yang tersisa pada permintaan tarik keseluruhan.

> **Note**
>
> Fungsionalitas ini mengharuskan fitur AI generatif diaktifkan untuk ruang dan hanya tersedia di Wilayah AS Barat (Oregon). Untuk informasi selengkapnya, lihat Mengelola fitur AI generatif.

9. Untuk melihat perubahan kode dalam permintaan tarik, pilih Perubahan. Anda dapat dengan cepat melihat berapa banyak file yang memiliki perubahan dalam permintaan tarik, dan file apa dalam permintaan tarik memiliki komentar pada mereka, di File diubah. Jumlah komentar yang ditampilkan di sebelah folder menunjukkan jumlah komentar pada file di folder itu. Perluas folder untuk melihat jumlah komentar untuk setiap file di folder. Anda juga dapat melihat komentar yang tersisa pada baris kode tertentu.

> **Note**
>
> Tidak semua perubahan dalam permintaan tarik dapat ditampilkan di konsol. Misalnya, Anda tidak dapat melihat submodul Git di konsol, sehingga Anda tidak dapat melihat perbedaan dalam submodul dalam permintaan tarik. Beberapa perbedaan mungkin terlalu besar untuk ditampilkan. Untuk informasi selengkapnya, lihat Kuota untuk repositori sumber di CodeCatalyst dan Melihat file.

10. Untuk melihat laporan kualitas permintaan tarik ini, pilih Laporan.

> **Note**
>
> Alur kerja harus dikonfigurasi untuk menghasilkan laporan agar laporan tersebut muncul dalam permintaan tarik Anda. Untuk informasi selengkapnya, lihat Pengujian dengan alur kerja.

Mengelola persyaratan untuk menggabungkan permintaan tarik dengan aturan persetujuan

Saat membuat permintaan tarik, Anda dapat memilih untuk menambahkan pengulas wajib atau opsional ke permintaan tarik individual tersebut. Namun, Anda juga dapat membuat persyaratan yang harus dipenuhi oleh semua permintaan tarik saat menggabungkan ke cabang tujuan tertentu. Persyaratan ini disebut aturan persetujuan. Aturan persetujuan dikonfigurasi untuk cabang di repositori. Saat Anda membuat permintaan tarik yang cabang tujuannya memiliki aturan persetujuan yang dikonfigurasi untuknya, persyaratan untuk aturan tersebut harus dipenuhi selain persetujuan dari pengulas yang diperlukan sebelum Anda dapat menggabungkan permintaan tarik ke cabang tersebut. Membuat aturan persetujuan dapat membantu Anda mempertahankan standar kualitas untuk penggabungan ke cabang seperti cabang default Anda.

Aturan persetujuan yang diterapkan ke cabang default repositori sumber Anda akan berperilaku sedikit berbeda dari aturan persetujuan yang diterapkan ke cabang lain. Aturan apa pun yang diterapkan ke cabang default akan diterapkan secara otomatis ke cabang mana pun yang Anda tentukan sebagai cabang default. Cabang yang sebelumnya ditetapkan sebagai cabang default akan tetap mempertahankan aturan yang diterapkan padanya.

Ketika Anda membuat aturan persetujuan, Anda harus mempertimbangkan bagaimana aturan itu akan dipenuhi oleh pengguna proyek Anda baik di masa sekarang maupun di masa depan. Misalnya, jika Anda memiliki enam pengguna dalam proyek Anda, dan Anda membuat aturan persetujuan yang memerlukan lima persetujuan sebelum dapat digabungkan ke cabang tujuan, Anda telah secara efektif membuat aturan yang mengharuskan semua orang kecuali orang yang membuat permintaan tarik untuk menyetujui permintaan tarik tersebut sebelum dapat digabungkan.

> **Note**
>
> Anda harus memiliki peran administrator Proyek untuk membuat dan mengelola aturan persetujuan dalam CodeCatalyst proyek. Anda tidak dapat membuat aturan persetujuan untuk repositori tertaut.

Anda tidak dapat menghapus aturan persetujuan, tetapi Anda dapat memperbaruinya agar tidak memerlukan persetujuan, yang secara efektif menghapus aturan tersebut.

Untuk melihat dan mengedit aturan persetujuan untuk cabang tujuan untuk permintaan tarik

1. Arahkan ke proyek tempat repositori Anda berada.
2. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber.

   Pilih repositori tempat Anda ingin melihat aturan persetujuan.
3. Pada halaman ikhtisar repositori, pilih Cabang.
4. Di kolom Aturan persetujuan, pilih Lihat untuk melihat status aturan apa pun untuk setiap cabang repositori.

   Dalam jumlah minimum persetujuan, jumlah tersebut sesuai dengan jumlah persetujuan yang diperlukan sebelum permintaan tarik dapat digabungkan ke cabang tersebut.
5. Untuk membuat atau mengubah aturan persetujuan, pilih Kelola pengaturan. Pada halaman pengaturan untuk repositori sumber, dalam aturan Persetujuan, pilih Edit.

> **Note**
>
> Anda harus memiliki peran administrator Project untuk mengedit aturan persetujuan.

6. Di Cabang, pilih nama cabang yang ingin Anda konfigurasikan aturan persetujuan dari daftar drop-down. Dalam Jumlah persetujuan minimum, masukkan nomor, lalu pilih Simpan.

Meninjau permintaan tarik

Anda dapat menggunakan CodeCatalyst konsol Amazon untuk meninjau dan mengomentari perubahan yang disertakan dalam permintaan tarik secara kolaboratif. Anda dapat menambahkan komentar ke baris kode individual dalam perbedaan antara cabang sumber dan tujuan, atau perbedaan antara revisi permintaan tarik. Anda dapat memilih untuk membuat ringkasan komentar yang tersisa pada perubahan kode dalam permintaan tarik untuk membantu Anda dengan cepat memahami umpan balik yang ditinggalkan oleh pengguna lain. Anda juga dapat memilih untuk membuat Lingkungan Dev untuk bekerja pada kode.

> **Note**
>
> Didukung oleh Amazon Bedrock: AWS mengimplementasikan deteksi penyalahgunaan otomatis. Karena deskripsi Tulis untuk saya dan Buat fitur ringkasan konten dibangun di Amazon Bedrock, pengguna dapat memanfaatkan sepenuhnya kontrol yang diterapkan di Amazon Bedrock untuk menegakkan keselamatan, keamanan, dan penggunaan kecerdasan buatan (AI) yang bertanggung jawab.

> **Tip**
>
> Anda dapat mengonfigurasi peristiwa permintaan tarik apa yang akan Anda terima email sebagai bagian dari profil Anda. Untuk informasi selengkapnya, lihat Mengelola notifikasi di Amazon CodeCatalyst.

Permintaan tarik menunjukkan perbedaan antara revisi permintaan tarik dan komit yang merupakan ujung cabang tujuan saat Anda membuat permintaan tarik. Ini disebut basis penggabungan. Untuk

informasi selengkapnya tentang perbedaan dan penggabungan basis di Git, lihat git-merge-basedi dokumentasi Git.

> **Tip**
>
> Saat bekerja di konsol, terutama jika Anda memiliki permintaan tarik terbuka untuk sementara waktu, pertimbangkan untuk menyegarkan browser Anda untuk memastikan Anda memiliki revisi terbaru yang tersedia untuk permintaan tarik sebelum Anda mulai memeriksanya.

Untuk meninjau permintaan tarik di CodeCatalyst konsol

1. Arahkan ke proyek Anda.
2. Arahkan ke pull request dengan melakukan salah satu hal berikut:
   - Jika permintaan tarik tercantum di halaman proyek, pilih dari daftar.
   - Jika permintaan tarik tidak tercantum di halaman proyek, pilih Lihat semua. Gunakan filter dan urutkan untuk menemukan permintaan tarik, lalu pilih dari daftar.
   - Di panel navigasi, pilih Kode, lalu pilih Tarik permintaan.
3. Pilih permintaan tarik yang ingin Anda tinjau dari daftar. Anda dapat memfilter daftar permintaan tarik dengan mengetikkan bagian dari namanya di bilah filter.
4. Dalam Ikhtisar, Anda dapat meninjau nama dan judul permintaan tarik. Anda dapat membuat dan melihat komentar yang tersisa pada permintaan tarik itu sendiri. Anda juga dapat melihat detail permintaan tarik, termasuk informasi tentang alur kerja yang berjalan, masalah terkait, pengulas, pembuat permintaan tarik, dan strategi penggabungan yang layak.

   > **Note**
   >
   > Komentar yang tersisa pada baris kode tertentu muncul di Perubahan.

5. (Opsional) Untuk menambahkan komentar yang berlaku untuk seluruh permintaan tarik, perluas Komentar pada permintaan tarik, lalu pilih Buat komentar.
6. (Opsional) Untuk melihat ringkasan semua komentar yang tersisa pada perubahan revisi permintaan tarik ini, pilih Buat ringkasan komentar.

> **Note**
>
> Fungsionalitas ini mengharuskan fitur AI generatif diaktifkan untuk ruang dan hanya tersedia di Wilayah AS Barat (Oregon). Untuk informasi selengkapnya, lihat Mengelola fitur AI generatif.

7. Di Perubahan, Anda dapat melihat perbedaan antara cabang tujuan dan revisi terbaru dari permintaan tarik. Jika ada lebih dari satu revisi, Anda dapat mengubah revisi apa yang dibandingkan dalam perbedaan di antara mereka. Untuk informasi lebih lanjut tentang revisi, lihatRevisi.

> **Tip**
>
> Anda dapat dengan cepat melihat berapa banyak file yang memiliki perubahan dalam permintaan tarik, dan file apa dalam permintaan tarik memiliki komentar pada mereka, di File diubah. Jumlah komentar yang ditampilkan di sebelah folder menunjukkan jumlah komentar pada file di folder itu. Perluas folder untuk melihat jumlah komentar untuk setiap file di folder.

8. Untuk mengubah cara perbedaan ditampilkan, pilih antara Unified dan Split.
9. Untuk menambahkan komentar ke baris dalam permintaan tarik, buka baris yang ingin Anda komentari. Pilih ikon komentar yang muncul untuk baris itu, masukkan komentar, lalu pilih Simpan.
10. Untuk melihat perubahan antara revisi dalam permintaan tarik, atau antara cabang sumber dan tujuan, pilih dari opsi yang tersedia di Membandingkan. Komentar pada baris dalam revisi dipertahankan dalam revisi tersebut.
11. Jika Anda telah mengonfigurasi alur kerja untuk menghasilkan laporan cakupan kode pada pemicu permintaan tarik, Anda dapat melihat temuan cakupan baris dan cabang dalam permintaan tarik yang relevan. Untuk menyembunyikan temuan cakupan kode, pilih Sembunyikan cakupan kode. Untuk informasi selengkapnya, lihat Laporan cakupan kode.
12. Jika Anda ingin membuat perubahan kode pada permintaan tarik, Anda dapat membuat Lingkungan Dev dari permintaan tarik. Pilih Buat Lingkungan Pengembang. Secara opsional tambahkan nama untuk Lingkungan Dev atau edit konfigurasinya dan kemudian pilih Buat.

13. Di Laporan, Anda dapat melihat laporan kualitas dalam permintaan tarik ini. Jika ada lebih dari satu revisi, Anda dapat mengubah revisi apa yang dibandingkan dalam perbedaan di antara mereka. Anda dapat memfilter laporan berdasarkan nama, status, alur kerja, tindakan, dan jenis.

    > **Note**
    >
    > Alur kerja harus dikonfigurasi untuk menghasilkan laporan agar laporan tersebut muncul dalam permintaan tarik Anda. Untuk informasi selengkapnya, lihat Mengkonfigurasi laporan kualitas dalam suatu tindakan.

14. Untuk melihat laporan tertentu, pilih dari daftar. Untuk informasi selengkapnya, lihat Pengujian dengan alur kerja.
15. Jika Anda terdaftar sebagai peninjau permintaan tarik ini dan ingin menyetujui perubahan, pastikan Anda melihat revisi terbaru, lalu pilih Menyetujui.

    > **Note**
    >
    > Semua pengulas yang diperlukan harus menyetujui permintaan tarik sebelum dapat digabungkan.

Memperbarui permintaan tarik

Anda dapat mempermudah anggota proyek lain untuk meninjau kode dengan memperbarui permintaan tarik. Anda dapat memperbarui permintaan tarik untuk mengubah pengulasnya, tautannya ke masalah, judul permintaan tarik, atau deskripsinya. Misalnya, Anda mungkin ingin mengubah pengulas yang diperlukan untuk permintaan tarik untuk menghapus seseorang yang pergi berlibur, dan menambahkan orang lain. Anda juga dapat memperbarui permintaan tarik dengan perubahan kode lebih lanjut dengan mendorong komit ke cabang sumber permintaan tarik terbuka. Setiap push ke cabang sumber permintaan tarik di repositori CodeCatalyst sumber membuat revisi. Anggota proyek dapat melihat perbedaan antara revisi dalam permintaan tarik.

Untuk memperbarui pengulas untuk permintaan tarik

1. Arahkan ke proyek tempat Anda ingin memperbarui pengulas permintaan tarik.
2. Pada halaman proyek, di bawah Buka permintaan tarik, pilih permintaan tarik tempat Anda ingin memperbarui pengulas. Atau, di panel navigasi, pilih Kode, pilih Permintaan tarik, lalu pilih permintaan tarik yang ingin Anda perbarui.

3. (Opsional) Di Ikhtisar, di area rincian permintaan tarik, pilih tanda tambah untuk menambahkan pengulas wajib atau opsional. Pilih X di sebelah reviewer untuk menghapusnya sebagai peninjau opsional atau wajib.

4. (Opsional) Di Ringkasan, di area Detail permintaan tarik, pilih Masalah tautan untuk menautkan masalah ke permintaan tarik, lalu pilih masalah dari daftar atau masukkan ID-nya. Untuk memutuskan tautan masalah, pilih ikon batalkan tautan di sebelah masalah yang ingin Anda putuskan tautannya.

Untuk memperbarui file dan kode di cabang sumber permintaan tarik

1. Untuk memperbarui beberapa file, buat Lingkungan Dev, atau kloning repositori dan cabang sumbernya dan gunakan klien Git atau lingkungan pengembangan terintegrasi (IDE) untuk membuat perubahan pada file di cabang sumber. Komit dan dorong perubahan ke cabang sumber di repositori CodeCatalyst sumber untuk memperbarui permintaan tarik secara otomatis dengan perubahan. Untuk informasi selengkapnya, lihat Mengkloning repositori sumber dan Memahami perubahan kode sumber dengan komit di Amazon CodeCatalyst.
2. Untuk memperbarui file individual di cabang sumber, Anda dapat menggunakan klien Git atau IDE seperti yang Anda lakukan untuk beberapa file. Anda juga dapat mengeditnya langsung di CodeCatalyst konsol. Untuk informasi selengkapnya, lihat Mengedit file.

Untuk memperbarui judul dan deskripsi permintaan tarik

1. Arahkan ke proyek tempat Anda ingin memperbarui judul atau deskripsi permintaan tarik.
2. Halaman proyek menampilkan permintaan tarik terbuka, termasuk informasi tentang siapa yang membuat permintaan tarik, repositori apa yang berisi cabang untuk permintaan tarik, dan kapan permintaan tarik dibuat. Anda dapat memfilter tampilan permintaan tarik terbuka berdasarkan repositori sumber. Pilih permintaan tarik yang ingin Anda ubah dari daftar.
3. Untuk melihat semua permintaan tarik, pilih Lihat semua. Atau, di panel navigasi, pilih Kode, lalu pilih Tarik permintaan. Gunakan kotak filter atau fungsi sortir untuk menemukan permintaan tarik yang ingin Anda ubah, lalu pilih.
4. Di Ikhtisar, pilih Edit.
5. Ubah judul atau deskripsi, lalu pilih Simpan.

Menggabungkan permintaan tarik

Setelah kode Anda ditinjau dan semua pengulas yang diperlukan telah menyetujuinya, Anda dapat menggabungkan permintaan tarik di CodeCatalyst konsol menggunakan strategi penggabungan yang didukung, seperti fast-forward. Tidak semua strategi gabungan yang didukung di CodeCatalyst konsol tersedia sebagai pilihan untuk semua permintaan tarik. CodeCatalyst mengevaluasi penggabungan dan hanya memungkinkan Anda memilih antara strategi gabungan yang tersedia di konsol dan mampu menggabungkan cabang sumber ke cabang tujuan. Anda juga dapat menggabungkan permintaan tarik dengan strategi gabungan Git pilihan Anda dengan menjalankan git merge perintah di komputer lokal Anda atau Lingkungan Pengembang untuk menggabungkan cabang sumber ke cabang tujuan. Anda kemudian dapat mendorong perubahan tersebut di cabang tujuan ke repositori sumber di. CodeCatalyst

> **Note**
>
> Menggabungkan cabang dan mendorong perubahan di Git tidak secara otomatis menutup permintaan tarik.

Jika Anda memiliki peran administrator Project, Anda juga dapat memilih untuk menggabungkan permintaan tarik yang belum memenuhi semua persyaratan untuk persetujuan dan aturan persetujuan.

Menggabungkan permintaan tarik (konsol)

Anda dapat menggabungkan permintaan tarik di CodeCatalyst konsol jika tidak ada konflik gabungan antara cabang sumber dan tujuan dan jika semua pengulas yang diperlukan telah menyetujui permintaan tarik. Jika ada konflik, atau jika penggabungan tidak dapat diselesaikan, tombol gabungan tidak aktif, dan label Tidak dapat digabungkan ditampilkan. Dalam hal ini, Anda harus mendapatkan persetujuan dari pemberi persetujuan yang diperlukan, menyelesaikan konflik secara lokal jika perlu, dan mendorong perubahan tersebut sebelum Anda dapat bergabung. Menggabungkan permintaan tarik akan secara otomatis mengirim email ke pembuat permintaan tarik serta pengulas yang diperlukan atau opsional. Ini tidak akan secara otomatis menutup atau mengubah status masalah apa pun yang terkait dengan permintaan tarik.

> **Tip**
>
> Anda dapat mengonfigurasi peristiwa permintaan tarik apa yang akan Anda terima email sebagai bagian dari profil Anda. Untuk informasi selengkapnya, lihat Mengelola notifikasi di Amazon CodeCatalyst.

Untuk menggabungkan permintaan tarik

1. Arahkan ke proyek tempat Anda ingin menggabungkan permintaan tarik.
2. Pada halaman proyek, di bawah Buka permintaan tarik, pilih permintaan tarik yang ingin Anda gabungkan. Jika Anda tidak melihat permintaan tarik, pilih Lihat semua permintaan tarik dan kemudian pilih dari daftar. Atau, di panel navigasi, pilih Kode, pilih Permintaan tarik, lalu pilih permintaan tarik yang ingin Anda gabungkan. Pilih Gabungkan.
3. Pilih dari strategi penggabungan yang tersedia untuk permintaan tarik. Secara opsional, pilih atau batalkan pilihan untuk menghapus cabang sumber setelah menggabungkan permintaan tarik, lalu pilih Gabung.

> **Note**
>
> Jika tombol Gabung tidak aktif, atau Anda melihat label Tidak dapat digabungkan, pengulas wajib belum menyetujui permintaan tarik, atau permintaan tarik tidak dapat digabungkan di konsol. CodeCatalyst Peninjau yang belum menyetujui permintaan tarik ditunjukkan oleh ikon jam di area Detail permintaan tarik di Ikhtisar. Jika semua pengulas yang diperlukan telah menyetujui permintaan tarik tetapi tombol Gabung masih tidak aktif, Anda mungkin memiliki konflik gabungan. Pilih label yang digarisbawahi Tidak dapat digabungkan untuk melihat detail selengkapnya tentang mengapa permintaan tarik tidak dapat digabungkan. Anda dapat menyelesaikan konflik gabungan untuk cabang tujuan di Lingkungan Pengembang atau CodeCatalyst konsol lalu menggabungkan permintaan tarik, atau Anda dapat menyelesaikan konflik dan menggabungkan secara lokal, lalu mendorong komit yang berisi penggabungan ke cabang sumber. CodeCatalyst Untuk informasi selengkapnya, lihat Menggabungkan permintaan tarik (Git) dan dokumentasi Git Anda.

Ganti persyaratan penggabungan

Jika Anda memiliki peran administrator Project, Anda dapat memilih untuk menggabungkan permintaan tarik yang belum memenuhi semua persyaratan untuk persetujuan dan aturan persetujuan yang diperlukan. Ini disebut sebagai mengesampingkan persyaratan untuk permintaan tarik. Anda dapat memilih untuk melakukan ini jika peninjau yang diperlukan tidak tersedia, atau jika kebutuhan mendesak muncul untuk menggabungkan permintaan tarik tertentu ke cabang yang memiliki aturan persetujuan yang tidak dapat dipenuhi dengan cepat.

Untuk menggabungkan permintaan tarik

1. Dalam permintaan tarik tempat Anda ingin mengganti persyaratan dan menggabungkan, pilih panah tarik-turun di sebelah tombol Gabung. Pilih Ganti persyaratan persetujuan.
2. Dalam alasan Override, berikan detail mengapa Anda menggabungkan permintaan tarik ini tanpa memenuhi aturan persetujuan dan persyaratan pengulas yang diperlukan. Meskipun ini opsional, ini sangat disarankan.
3. Secara opsional pilih strategi gabungan, atau terima default. Anda juga dapat memilih untuk memperbarui pesan komit yang dibuat secara otomatis dengan detail selengkapnya.
4. Pilih atau batal pilihan untuk menghapus cabang sumber saat bergabung. Kami menyarankan Anda mempertahankan cabang sumber saat mengganti persyaratan untuk menggabungkan permintaan tarik hingga Anda memiliki kesempatan untuk meninjau keputusan dengan anggota tim lainnya.
5. Pilih Gabungkan.

Menggabungkan permintaan tarik (Git)

Git mendukung banyak opsi untuk menggabungkan dan mengelola cabang. Perintah berikut adalah beberapa opsi yang dapat Anda gunakan. Untuk informasi selengkapnya, lihat dokumentasi yang tersedia di situs web Git. Setelah Anda menggabungkan dan mendorong perubahan Anda, tutup permintaan tarik secara manual. Untuk informasi selengkapnya, lihat Menutup permintaan tarik.

Perintah Git umum untuk menggabungkan cabang

Menggabungkan perubahan dari cabang sumber di repo lokal ke cabang tujuan di repo lokal.	`git checkout destination-branch-name`
	
	`git merge source-branch-name`

Menggabungkan cabang sumber ke cabang tujuan, menentukan penggabungan maju cepat. Ini menggabungkan cabang dan memindahkan pointer cabang tujuan ke ujung cabang sumber.	`git checkout` *`destination-branch-name`* `git merge --ff-only` *`source-branch-name`*
Menggabungkan cabang sumber ke cabang tujuan, menentukan gabungan squash. Ini menggabungkan semua komit dari cabang sumber menjadi komit gabungan tunggal di cabang tujuan.	`git checkout` *`destination-branch-name`* `git merge --squash` *`source-branch-name`*
Menggabungkan cabang sumber ke cabang tujuan, menentukan penggabungan tiga arah. Ini membuat komit gabungan dan menambahkan komit individu dari cabang sumber ke cabang tujuan.	`git checkout` *`destination-branch-name`* `git merge --no-ff` *`source-branch-name`*
Menghapus cabang sumber di repo lokal. Ini berguna untuk dilakukan sebagai pembersihan untuk repo lokal Anda setelah bergabung ke cabang tujuan dan mendorong perubahan ke repositori sumber.	`git branch -d` *`source-branch-name`*
Menghapus cabang sumber di repositori jarak jauh (repositori sumber di CodeCatalyst) menggunakan nama panggilan yang ditentukan repo lokal untuk repositori jarak jauh. (Perhatikan penggunaan titik dua (`:`).) Atau, tentukan `--delete` sebagai bagian dari perintah.	`git push` *`remote-name`* `:`*`source-branch-name`* `git push` *`remote-name`* `--delete` *`source-branch-name`*

Menutup permintaan tarik

Anda dapat menandai permintaan tarik sebagai Ditutup. Ini tidak menggabungkan permintaan tarik, tetapi dapat membantu Anda menentukan permintaan tarik mana yang memerlukan tindakan dan permintaan tarik mana yang tidak lagi relevan. Sebaiknya tutup permintaan tarik jika Anda tidak lagi

berencana untuk menggabungkan perubahan tersebut, atau jika perubahan tersebut digabungkan dengan permintaan tarik lainnya.

Menutup permintaan tarik akan secara otomatis mengirim email ke pembuat permintaan tarik serta pengulas yang diperlukan atau opsional. Ini tidak akan secara otomatis mengubah status masalah apa pun yang terkait dengan permintaan tarik.

> **Note**
>
> Anda tidak dapat membuka kembali permintaan tarik setelah ditutup.

Untuk menutup permintaan tarik

1. Arahkan ke proyek tempat Anda ingin menutup permintaan tarik.
2. Pada halaman proyek, permintaan tarik terbuka ditampilkan. Pilih permintaan tarik yang ingin Anda tutup.
3. Pilih Tutup.
4. Tinjau informasi, lalu pilih Tutup permintaan tarik.

Memahami perubahan kode sumber dengan komit di Amazon CodeCatalyst

Commit adalah snapshot dari isi dan perubahan isi repositori Anda. Setiap kali pengguna melakukan dan mendorong perubahan ke cabang, informasi itu disimpan. Informasi komit Git mencakup pembuat komit, orang yang melakukan perubahan, tanggal dan waktu, dan perubahan yang dibuat. Informasi serupa secara otomatis disertakan saat Anda membuat atau mengedit file di CodeCatalyst konsol Amazon, tetapi nama penulis adalah nama CodeCatalyst pengguna Anda. Anda juga dapat menambahkan tag Git ke commit untuk membantu Anda mengidentifikasi commit tertentu.

Di Amazon CodeCatalyst, Anda dapat:

- Lihat daftar komit untuk cabang.
- Lihat komit individu, termasuk perubahan yang dibuat dalam komit jika dibandingkan dengan orang tua atau orang tuanya.

Anda juga dapat melihat file dan folder. Untuk informasi selengkapnya, lihat Mengelola file kode sumber di Amazon CodeCatalyst.

Topik

Melihat komit ke cabang

Anda dapat melihat riwayat perubahan yang dilakukan pada cabang dengan meninjau komit cabang di CodeCatalyst konsol. Ini membantu Anda memahami siapa yang membuat perubahan pada cabang dan kapan. Anda juga dapat meninjau perubahan yang dibuat dalam komit tertentu.

> **Tip**
>
> Anda juga dapat melihat riwayat komit yang membuat perubahan pada file tertentu. Untuk mengetahui informasi selengkapnya, lihat Melihat file.

Anda juga dapat melihat commit dengan menggunakan klien Git Anda. Untuk informasi selengkapnya, lihat dokumentasi Git Anda.

Untuk melihat komit (konsol)

1. Arahkan ke proyek yang berisi repositori sumber tempat Anda ingin melihat komit.
2. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber.

   Pilih repositori tempat Anda ingin melihat komit ke cabang.
3. Cabang default repositori ditampilkan, termasuk informasi tentang komit terbaru ke cabang. Pilih Komit. Atau, pilih Lainnya, lalu pilih Lihat komit.
4. Untuk melihat komit untuk cabang yang berbeda, pilih pemilih cabang, lalu pilih nama cabang.
5. Untuk melihat detail tentang komit tertentu, pilih judulnya dari judul Komit. Detail komit ditampilkan, termasuk informasi tentang komit induknya dan perubahan yang dibuat pada kode dengan membandingkan komit induk dengan komit yang ditentukan.

> **Tip**
>
> Jika komit memiliki lebih dari satu induk, Anda dapat memilih komit induk mana untuk melihat informasi dan menampilkan perubahan dengan memilih ikon tarik-turun di sebelah ID komit induk.

Mengubah cara komit ditampilkan (CodeCatalystkonsol)

Anda dapat mengubah informasi apa yang ditampilkan dalam tampilan Komit. Anda dapat memilih untuk menyembunyikan atau menampilkan kolom seperti author dan commit ID.

Untuk mengubah cara komit ditampilkan (konsol)

1. Arahkan ke proyek yang berisi repositori sumber tempat Anda ingin melihat komit.
2. Pilih nama repositori dari daftar repositori sumber untuk proyek. Atau, di panel navigasi, pilih Kode, lalu pilih Repositori sumber.

   Pilih repositori tempat Anda ingin mengubah cara Anda melihat komit.
3. Cabang default repositori ditampilkan, termasuk informasi tentang komit terbaru ke cabang. Pilih Komit.
4. Pilih ikon roda gigi.
5. Di Preferensi, pilih jumlah komit yang akan ditampilkan, dan pilih apakah akan menampilkan informasi tentang pembuat komit, tanggal komit, dan ID komit.

   > **Note**
   >
   > Anda tidak dapat menyembunyikan judul komit dalam tampilan informasi.
6. Ketika Anda telah membuat perubahan, pilih Simpan untuk menyimpannya, atau Batal untuk membuangnya.

Kuota untuk repositori sumber di CodeCatalyst

Tabel berikut menjelaskan kuota dan batas untuk repositori sumber di Amazon. CodeCatalyst Untuk informasi selengkapnya tentang kuota di Amazon CodeCatalyst, lihatKuota untuk CodeCatalyst.

Sumber Daya	Informasi
Nama cabang	Setiap kombinasi karakter yang diizinkan antara 1 dan 256 karakter panjangnya dan harus unik dalam repositori. Nama cabang tidak boleh: • dimulai atau diakhiri dengan garis miring (`/`) atau tanda titik (`.`) • terdiri dari karakter tunggal @ • mengandung dua atau lebih tanda titik (`..`) berturut-turut, garis miring ke depan (`//`), atau kombinasi dari karakter berikut: `@{` • berisi spasi atau salah satu karakter berikut: `? ^ * [ \ ~ :` Nama cabang adalah referensi. Banyak pembatasan pada nama cabang didasarkan pada standar referensi Git. Untuk informasi selengkapnya, lihat Git Internals dan git-check-ref-format.
Komentar pada pull request	Maksimal 1.000 pada permintaan tarik.
Pesan komit	Maksimal 1024 karakter.
Jalur file	Setiap kombinasi karakter yang diperbolehkan dengan panjang antara 1 dan 4.096 karakter. Jalur file harus berupa nama yang tidak ambigu yang menentukan file dan lokasi yang tepat dari file tersebut. Kedalaman jalur file tidak boleh melebihi 20 direktori. Selain itu, jalur file tidak boleh: • berisi string kosong • berupa jalur file relatif

Sumber Daya	Informasi

	• menyertakan salah satu kombinasi karakter berikut:
	`/./`
	`/../`
	`//`
	• diakhiri dengan garis miring atau garis miring terbalik
	
	Nama file dan jalur harus memenuhi syarat. Nama dan jalur ke sebuah file pada komputer lokal Anda harus mengikuti standar untuk sistem operasi tersebut. Saat menentukan jalur ke file dalam repositori, gunakan standar untuk Amazon Linux.

Ukuran file	Maksimum 6 MB untuk setiap file individual saat menggunakan CodeCatalyst konsol.
Ukuran file dapat dilihat di konsol CodeCatalyst	Maksimum 6 MB untuk setiap file individual saat menggunakan CodeCatalyst konsol.

Sumber Daya	Informasi
Ukuran gumpalan Git	Maksimal 2 GB. **Note** Tidak ada batas pada jumlah atau ukuran total semua file dalam satu commit, selama metadata tidak melebihi 6 MB dan gumpalan tunggal tidak melebihi 2 GB. Namun, sebagai praktik terbaik, pertimbangkan untuk membuat beberapa komit yang lebih kecil daripada satu komit besar.
Metadata untuk commit	Maksimum 6 MB untuk metadata gabungan untuk komit (misalnya, kombinasi informasi penulis, tanggal, daftar komit induk, dan pesan komit). **Note** Tidak ada batas pada jumlah atau ukuran total semua file dalam satu commit, selama data tidak melebihi 20 MB, file individu tidak melebihi 6 MB, dan gumpalan tunggal tidak melebihi 2 GB.
Jumlah CodeCatalyst masalah yang dapat ditautkan ke permintaan tarik	50
Jumlah masalah Jira yang dapat ditautkan ke permintaan tarik	50
Jumlah permintaan tarik terbuka dalam satu spasi	Maksimal 1.000 untuk CodeCatalyst ruang Amazon.

Sumber Daya	Informasi
Jumlah total permintaan tarik dalam satu spasi	Maksimal 10.000 untuk CodeCatalyst ruang Amazon.
Jumlah referensi dalam satu kali push	Maksimum 4.000, termasuk membuat, menghapus, dan memperbarui. Tidak ada batas jumlah keseluruhan referensi dalam repositori.
Jumlah repositori dalam satu ruang	Maksimal 5.000 untuk CodeCatalyst ruang Amazon.
Deskripsi repositori	Kombinasi karakter apa pun dengan panjang antara 0 dan 1.000 karakter. Deskripsi repositor i adalah opsional.
Nama repositori	Nama repositori harus unik dalam sebuah proyek. Mereka dapat berisi kombinasi huruf, angka, titik, garis bawah, dan tanda hubung antara 1 dan 100 karakter panjangnya. Nama tidak peka huruf besar/kecil. Nama repositor i tidak dapat diakhiri dengan.git, tidak dapat berisi spasi, dan tidak dapat berisi salah satu karakter berikut: `! ? @ # $ % ^ & * ( ) + = { } [ ] | \ / > < ~ ` ' " ; :`
Ukuran repositori	Ukuran repositori dipengaruhi oleh batas penyimpanan keseluruhan untuk ruang Anda. Untuk informasi lebih lanjut, lihat Harga danMemecahkan masalah dengan repositori sumber.
Peninjau untuk permintaan tarik	Maksimal total 100 pengulas (opsional atau wajib) untuk permintaan tarik.

Sumber Daya	Informasi
Ringkasan tertulis untuk permintaan tarik	Jumlah maksimum ringkasan tertulis untuk permintaan tarik tergantung pada tingkat penagihan untuk ruang Anda. Untuk informasi selengkapnya, lihat Harga .

Menulis dan memodifikasi kode dengan Dev Environments di CodeCatalyst

Dev Environments adalah lingkungan pengembangan berbasis cloud. Di Amazon CodeCatalyst, Anda menggunakan Dev Environments untuk mengerjakan kode yang disimpan di repositori sumber proyek Anda. Saat membuat Lingkungan Dev, Anda memiliki beberapa opsi:

- Buat Lingkungan Dev khusus proyek CodeCatalyst untuk mengerjakan kode dengan lingkungan pengembangan terintegrasi (IDE) yang didukung.
- Buat Lingkungan Dev kosong, kloning kode ke dalamnya dari repositori sumber, dan kerjakan kode itu dengan IDE yang didukung.
- Buat Lingkungan Pengembang di IDE dan kloning repositori sumber ke Lingkungan Dev

Devfile adalah file YAMAL standar terbuka yang menstandarisasi Lingkungan Dev Anda. Dengan kata lain, file ini mengkodifikasi alat pengembangan yang diperlukan untuk Lingkungan Dev Anda. Akibatnya, Anda dapat dengan cepat mengatur Lingkungan Dev, beralih antar proyek, dan mereplikasi konfigurasi Lingkungan Dev di seluruh anggota tim. Dev Environments meminimalkan waktu yang Anda habiskan untuk membuat dan memelihara lingkungan pengembangan lokal, karena mereka menggunakan devfile yang mengonfigurasi semua alat yang Anda perlukan untuk membuat kode, menguji, dan men-debug untuk proyek tertentu.

Alat proyek dan pustaka aplikasi yang disertakan dalam Lingkungan Dev Anda ditentukan oleh devfile di repositori sumber proyek Anda. Jika Anda tidak memiliki devfile di repositori sumber Anda, CodeCatalyst secara otomatis menerapkan devfile default. Devfile default ini mencakup alat untuk bahasa dan kerangka kerja pemrograman yang paling sering digunakan. Jika proyek Anda dibuat menggunakan cetak biru, devfile secara otomatis dibuat oleh. CodeCatalyst Untuk informasi lebih lanjut tentang devfile, lihat https://devfile.io.

Setelah Anda membuat Lingkungan Dev, hanya Anda yang dapat mengaksesnya. Di Lingkungan Dev Anda, Anda dapat melihat dan mengerjakan kode repositori sumber Anda dalam IDE yang didukung.

Secara default, Dev Environment dikonfigurasi untuk memiliki prosesor 2-core, RAM 4 GB, dan penyimpanan persisten 16 GB. Jika Anda memiliki izin administrator Space, Anda dapat mengubah tingkat penagihan untuk ruang Anda untuk menggunakan opsi konfigurasi Lingkungan Dev yang berbeda dan mengelola batas komputasi dan penyimpanan.

Topik

Membuat Lingkungan Dev

Anda dapat membuat Lingkungan Pengembang dengan berbagai cara:

- Buat Lingkungan Pengembang CodeCatalyst dengan repositori sumber atau repositori CodeCatalyst sumber tertaut dari halaman Ikhtisar, Lingkungan Pengembang, atau repositori Sumber
- Buat Lingkungan Dev kosong CodeCatalyst yang tidak terhubung ke repositori sumber dari halaman Lingkungan Dev
- Buat Lingkungan Pengembang di IDE pilihan Anda dan kloning repositori sumber apa pun ke Lingkungan Dev

Anda dapat membuat satu Lingkungan Dev per cabang repositori. Sebuah proyek dapat memiliki beberapa repositori. Lingkungan Dev yang Anda buat hanya dapat dikelola dengan CodeCatalyst akun Anda, tetapi Anda dapat membuka Lingkungan Dev dan bekerja di dalamnya dengan IDE yang didukung. Anda harus AWS Toolkit menginstal untuk menggunakan Lingkungan Dev di IDE Anda. Untuk informasi selengkapnya, lihat Lingkungan pengembangan terintegrasi yang didukung untuk Lingkungan Dev. Secara default, Dev Environments dibuat dengan prosesor 2-core, RAM 4 GB, dan penyimpanan persisten 16 GB.

> **Note**
>
> Jika Anda membuat Lingkungan Pengembang yang terkait dengan repositori sumber, kolom Resource selalu menampilkan cabang yang Anda tentukan saat membuat Lingkungan Dev ini. Ini berlaku bahkan jika Anda membuat cabang lain, beralih ke cabang lain dalam Lingkungan Dev, atau mengkloning repositori tambahan. Jika Anda membuat Lingkungan Dev kosong, kolom Resource akan kosong.

Lingkungan pengembangan terintegrasi yang didukung untuk Lingkungan Dev

Anda dapat menggunakan Dev Environments dengan lingkungan pengembangan terintegrasi (IDE) yang didukung berikut ini:

- AWS Cloud9
- JetBrains IDE
  - IntelliJ IDE Ultimate
  - GoLand
  - PyCharmProfesional
- Kode Visual Studio

Menciptakan Lingkungan Pengembang di CodeCatalyst

Untuk mulai bekerja dengan Dev Environment in CodeCatalyst, autentikasi dan masuk dengan AWS Builder ID atau SSO Anda.

Untuk membuat Lingkungan Dev dari cabang

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek tempat Anda ingin membuat Lingkungan Dev.
3. Di panel navigasi, lakukan salah satu hal berikut:
   - Pilih Ikhtisar, lalu arahkan ke bagian My Dev Environments.
   - Pilih Kode, lalu pilih Dev Environments.
   - Pilih Kode, pilih repositori Sumber, dan pilih repositori yang ingin Anda buat Lingkungan Dev.

4. Pilih Buat Lingkungan Pengembang.
5. Pilih IDE yang didukung dari menu tarik-turun. Untuk informasi selengkapnya, lihat Lingkungan pengembangan terintegrasi yang didukung untuk Lingkungan Dev.
6. Pilih Kloning repositori.
7. Lakukan salah satu hal berikut ini:

   a. Pilih repositori yang akan dikloning, pilih Bekerja di cabang yang ada, lalu pilih cabang dari menu drop-down cabang yang ada.

   > **Note**
   >
   > Jika Anda memilih repositori pihak ketiga, Anda harus bekerja di cabang yang ada.

   b. Pilih repositori untuk dikloning, pilih Bekerja di cabang baru, masukkan nama cabang ke bidang Nama cabang, dan pilih cabang untuk membuat cabang baru dari Buat cabang dari menu drop-down.

   > **Note**
   >
   > Jika Anda membuat Lingkungan Dev dari halaman repositori Sumber atau dari repositori sumber tertentu, Anda tidak perlu memilih repositori. Lingkungan Dev akan dibuat dari repositori sumber yang Anda pilih dari halaman repositori Sumber.

8. (Opsional) Di Alias - opsional, masukkan alias untuk Lingkungan Pengembang.
9. (Opsional) Pilih tombol edit konfigurasi Lingkungan Dev untuk mengedit konfigurasi komputasi, penyimpanan, atau batas waktu Dev Environment.
10. (Opsional) Di Amazon Virtual Private Cloud (Amazon VPC) - opsional, pilih koneksi VPC yang ingin Anda kaitkan dengan Lingkungan Dev Anda dari menu tarik-turun.

    Jika VPC default disetel untuk ruang Anda, Lingkungan Pengembang Anda akan berjalan terhubung ke VPC tersebut. Anda dapat mengganti ini dengan mengaitkan koneksi VPC yang berbeda. Juga, perhatikan bahwa Lingkungan Dev yang terhubung dengan VPC tidak mendukung. AWS Toolkit

> **Note**
>
> Saat Anda membuat Lingkungan Dev dengan koneksi VPC, antarmuka jaringan baru dibuat di dalam VPC. CodeCatalyst berinteraksi dengan antarmuka ini menggunakan peran VPC terkait. Juga, pastikan bahwa blok CIDR IPv4 Anda tidak dikonfigurasi ke rentang `172.16.0.0/12` alamat IP.

11. Pilih Buat. Saat Dev Environment Anda sedang dibuat, kolom status Dev Environment akan menampilkan Mulai, dan kolom status akan ditampilkan Running setelah Dev Environment telah dibuat.

Untuk membuat Lingkungan Dev kosong

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek tempat Anda ingin membuat Lingkungan Dev.
3. Di panel navigasi, lakukan salah satu hal berikut:
   - Pilih Ikhtisar, lalu arahkan ke bagian My Dev Environments.
   - Pilih Kode, lalu pilih Dev Environments.
4. Pilih Buat Lingkungan Pengembang.
5. Pilih IDE yang didukung dari menu tarik-turun. Untuk informasi selengkapnya, lihat Lingkungan pengembangan terintegrasi yang didukung untuk Lingkungan Dev.
6. Pilih Buat Lingkungan Dev kosong.
7. (Opsional) Di Alias - opsional, masukkan alias untuk Lingkungan Pengembang.
8. (Opsional) Pilih tombol edit konfigurasi Lingkungan Dev untuk mengedit konfigurasi komputasi, penyimpanan, atau batas waktu Dev Environment.
9. (Opsional) Di Amazon Virtual Private Cloud (Amazon VPC) - opsional, pilih koneksi VPC yang ingin Anda kaitkan dengan Lingkungan Dev Anda dari menu tarik-turun.

   Jika VPC default disetel untuk ruang Anda, Lingkungan Pengembang Anda akan berjalan terhubung ke VPC tersebut. Anda dapat mengganti ini dengan mengaitkan koneksi VPC yang berbeda. Juga, perhatikan bahwa Lingkungan Dev yang terhubung dengan VPC tidak mendukung. AWS Toolkit

> **Note**
>
> Saat Anda membuat Lingkungan Dev dengan koneksi VPC, antarmuka jaringan baru dibuat di dalam VPC. CodeCatalyst berinteraksi dengan antarmuka ini menggunakan peran VPC terkait. Juga, pastikan bahwa blok CIDR IPv4 Anda tidak dikonfigurasi ke rentang `172.16.0.0/12` alamat IP.

10. Pilih Buat. Saat Dev Environment Anda sedang dibuat, kolom status Dev Environment akan menampilkan Mulai, dan kolom status akan ditampilkan Running setelah Dev Environment telah dibuat.

> **Note**
>
> Membuat dan membuka Lingkungan Pengembang untuk pertama kalinya mungkin memakan waktu satu hingga dua menit.

> **Note**
>
> Setelah Lingkungan Dev terbuka di IDE, Anda mungkin perlu mengubah direktori ke repositori sumber sebelum Anda melakukan dan mendorong perubahan ke kode Anda.

Membuat Lingkungan Dev dalam IDE

Anda dapat menggunakan Dev Environments untuk dengan cepat mengerjakan kode yang disimpan di repositori sumber proyek Anda. Dev Environments meningkatkan kecepatan pengembangan Anda karena Anda dapat segera memulai pengkodean di lingkungan pengembangan cloud khusus proyek yang berfungsi penuh dengan lingkungan pengembangan terintegrasi (IDE) yang didukung.

Untuk informasi tentang bekerja dengan CodeCatalyst dari IDE, lihat dokumentasi berikut.

- Amazon CodeCatalyst untuk JetBrains IDE
- Amazon CodeCatalyst untuk Kode VS
- Amazon CodeCatalyst untuk AWS Cloud9

Menghentikan Lingkungan Pengembang

`/projects`Direktori Dev Environment menyimpan file yang ditarik dari repositori sumber dan devfile yang digunakan untuk mengkonfigurasi Lingkungan Dev. `/home`Direktori, yang kosong pada pembuatan Dev Environment, menyimpan file yang Anda buat saat menggunakan Dev Environment Anda. Segala sesuatu di `/projects` dan `/home` direktori Lingkungan Dev disimpan terus-menerus, sehingga Anda dapat berhenti bekerja di Lingkungan Pengembang jika Anda perlu beralih ke Lingkungan Dev, repositori, atau proyek lain.

> **Warning**
>
> Lingkungan Pengembang tidak akan habis waktu jika ada instance, termasuk browser web, shell jarak jauh, dan IDE, tetap terhubung. Jadi, pastikan untuk menutup semua instance yang terhubung untuk menghindari biaya tambahan.

Lingkungan Pengembang akan berhenti secara otomatis jika tidak digunakan untuk jumlah waktu yang dipilih di bidang Timeout selama pembuatan Lingkungan Dev. Anda dapat menghentikan Lingkungan Pengembang sebelum menganggur. Jika Anda memilih Tidak ada batas waktu saat membuat Lingkungan Dev, Lingkungan Pengembang tidak akan berhenti secara otomatis. Sebaliknya, itu akan berjalan terus menerus.

> **Warning**
>
> Jika Anda menghentikan Lingkungan Pengembang yang terkait dengan koneksi VPC yang dihapus, itu tidak dapat dilanjutkan.

Untuk menghentikan Lingkungan Pengembang dari halaman Lingkungan Pengembang

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek tempat Anda ingin menghentikan Lingkungan Pengembang.
3. Di panel navigasi, pilih Kode.
4. Pilih Lingkungan Pengembang.
5. Pilih tombol radio untuk Lingkungan Dev yang ingin Anda hentikan.
6. Dari menu Tindakan, pilih Berhenti.

> **Note**
>
> Penggunaan komputasi hanya ditagih saat Lingkungan Dev berjalan, tetapi penggunaan penyimpanan ditagih selama Lingkungan Dev ada. Hentikan Lingkungan Pengembang Anda saat tidak digunakan untuk menghentikan penagihan komputasi.

Melanjutkan Lingkungan Pengembang

`/projects`Direktori Dev Environment menyimpan file yang ditarik dari repositori sumber dan devfile yang digunakan untuk mengkonfigurasi Lingkungan Dev. `/home`Direktori, yang kosong pada pembuatan Dev Environment, menyimpan file yang Anda buat saat menggunakan Dev Environment Anda. Segala sesuatu di `/projects` dan `/home` direktori Lingkungan Pengembang disimpan secara terus-menerus, sehingga Anda dapat berhenti bekerja di Lingkungan Pengembang jika Anda perlu beralih ke Lingkungan Dev, repositori, atau proyek lain dan melanjutkan bekerja di Lingkungan Pengembang Anda di lain waktu.

Lingkungan Pengembang akan berhenti secara otomatis jika tidak digunakan untuk jumlah waktu yang dipilih di bidang Timeout selama pembuatan Lingkungan Dev. Anda harus menutup tab AWS Cloud9 browser agar Lingkungan Dev menganggur.

> **Note**
>
> Lingkungan Dev masih tersedia dan berjalan bahkan jika Anda menghapus cabang yang Anda gunakan untuk membuat Lingkungan Dev. Jika Anda ingin melanjutkan bekerja di Lingkungan Pengembang tempat Anda menghapus cabangnya, buat cabang baru dan dorong perubahan Anda ke sana.

Untuk melanjutkan Lingkungan Pengembang dari halaman ikhtisar

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek tempat Anda ingin melanjutkan Lingkungan Pengembang, dan arahkan ke bagian My Dev Environments.
3. Pilih Resume in (IDE).

    - Untuk JetBrains IDE, pilih JetBrains GateWay-EAP saat diminta untuk Memilih aplikasi untuk membuka tautan -gateway. JetBrains Pilih Buka Tautan untuk mengonfirmasi saat diminta.

- Untuk VS Code IDE, pilih VS Code saat diminta untuk Memilih aplikasi untuk membuka tautan VS Code. Pilih Buka Tautan untuk mengonfirmasi.

Untuk melanjutkan Lingkungan Dev dari repositori sumber

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek tempat Anda ingin melanjutkan Lingkungan Pengembang.
3. Di panel navigasi, pilih Kode.
4. Pilih Source Repositories.
5. Pilih repositori sumber yang berisi Lingkungan Dev yang ingin Anda lanjutkan.
6. Pilih nama cabang untuk melihat menu drop-down cabang Anda, lalu pilih cabang Anda.
7. Pilih Lanjutkan Lingkungan Pengembang.

   - Untuk JetBrains IDE, pilih Buka Tautan untuk mengonfirmasi saat diminta Izinkan situs ini membuka tautan JetBrains -gateway dengan JetBrains Gateway? .
   - Untuk VS Code IDE, pilih Buka Tautan untuk mengonfirmasi saat diminta Izinkan situs ini membuka tautan VS Code dengan Visual Studio Code? .

Untuk melanjutkan Lingkungan Pengembang dari halaman Lingkungan Pengembang

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek tempat Anda ingin melanjutkan Lingkungan Pengembang.
3. Di panel navigasi, pilih Kode.
4. Pilih Lingkungan Pengembang.
5. Dari kolom IDE, pilih Lanjutkan di (IDE) untuk Lingkungan Dev.

   - Untuk JetBrains IDE, pilih Buka Tautan untuk mengonfirmasi saat diminta Izinkan situs ini membuka tautan JetBrains -gateway dengan JetBrains Gateway? .
   - Untuk VS Code IDE, pilih Buka Tautan untuk mengonfirmasi saat diminta Izinkan situs ini membuka tautan VS Code dengan Visual Studio Code? .

> **Note**
>
> Melanjutkan Lingkungan Pengembang mungkin memakan waktu beberapa menit.

Mengedit Lingkungan Pengembang

Saat IDE Anda berjalan, Anda dapat mengedit Lingkungan Dev. Jika Anda mengedit batas waktu komputasi atau tidak aktif, Lingkungan Pengembang akan dimulai ulang setelah Anda menyimpan perubahan.

Untuk mengedit Lingkungan Pengembang

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek tempat Anda ingin mengedit Lingkungan Dev.
3. Di panel navigasi, pilih Kode.
4. Pilih Lingkungan Pengembang.
5. Pilih Lingkungan Dev yang ingin Anda edit.
6. Pilih Edit.
7. Buat perubahan yang Anda inginkan pada batas waktu komputasi atau ketidakaktifan.
8. Pilih Simpan.

Menghapus Lingkungan Dev

Setelah selesai mengerjakan konten yang disimpan di Lingkungan Dev Anda, Anda dapat menghapus Lingkungan Pengembang. Buat Lingkungan Dev baru untuk mengerjakan konten baru. Jika Anda menghapus Lingkungan Pengembang, konten yang bertahan akan dihapus secara permanen. Sebelum Anda menghapus Dev Environment, pastikan Anda melakukan dan mendorong perubahan kode Anda ke repositori sumber asli Dev Environment. Setelah Anda menghapus Lingkungan Dev, tagihan komputasi dan penyimpanan untuk Lingkungan Pengembang akan berhenti.

Setelah Anda menghapus Lingkungan Dev Anda, mungkin perlu beberapa menit untuk memperbarui kuota penyimpanan. Jika Anda telah mencapai kuota penyimpanan, Anda tidak akan dapat membuat Lingkungan Dev baru selama waktu ini.

> **Important**
>
> Menghapus Lingkungan Pengembang tidak dapat dibatalkan. Setelah Anda menghapus Lingkungan Dev, Anda tidak lagi dapat memulihkannya.

Untuk menghapus Lingkungan Pengembang

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek tempat Anda ingin menghapus Lingkungan Dev.
3. Di panel navigasi, pilih Kode.
4. Pilih Lingkungan Pengembang.
5. Pilih Lingkungan Dev yang ingin Anda hapus.
6. Pilih Hapus.
7. Masukkan **`delete`** untuk mengonfirmasi penghapusan Lingkungan Pengembang.
8. Pilih Hapus.

> **Note**
>
> Sebelum menghapus koneksi VPC di ruang Anda, pastikan untuk menghapus Lingkungan Pengembang yang terkait dengan VPC tersebut.
> Bahkan jika Anda menghapus Lingkungan Pengembang, Anda mungkin tidak menghapus antarmuka jaringan di VPC. Pastikan untuk membersihkan sumber daya Anda sesuai kebutuhan. Jika terjadi kesalahan saat Anda menghapus Lingkungan Pengembang yang terhubung dengan VPC, Anda harus melepaskan koneksi basi Anda, dan menghapusnya setelah mengonfirmasi bahwa itu tidak digunakan.

Menghubungkan ke Lingkungan Dev menggunakan SSH

Anda dapat terhubung ke Lingkungan Dev Anda menggunakan SSH untuk melakukan tindakan tanpa batasan, seperti penerusan port, mengunggah dan mengunduh file, dan menggunakan IDE lainnya.

> **Note**
>
> Jika Anda ingin terus menggunakan SSH untuk waktu yang lama setelah menutup tab atau jendela IDE, pastikan untuk menetapkan batas waktu tinggi untuk Lingkungan Dev Anda sehingga tidak berhenti karena tidak aktif di IDE.

Prasyarat

- Anda memerlukan salah satu dari sistem operasi berikut:
  - Windows 10 atau yang lebih baru dan OpenSSH diaktifkan
  - macOS dan Bash versi 3 atau lebih tinggi
  - Linux dengan`yum`, `dpkg` atau manajer `rpm` paket dan Bash versi 3 atau lebih tinggi
- Anda juga membutuhkan AWS CLI versi 2.9.4 atau lebih tinggi.

Untuk terhubung ke Lingkungan Dev menggunakan SSH

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek tempat Anda ingin terhubung ke Lingkungan Dev menggunakan SSH.
3. Di panel navigasi, pilih Kode.
4. Pilih Lingkungan Pengembang.
5. Pilih Lingkungan Dev yang sedang berjalan yang ingin Anda sambungkan menggunakan SSH.
6. Pilih Connect via SSH, pilih sistem operasi yang Anda inginkan, dan lakukan hal berikut:

   - Jika Anda belum melakukannya, tempel dan jalankan perintah pertama di terminal yang Anda tentukan. Perintah mengunduh skrip dan menjalankan modifikasi berikut di lingkungan lokal Anda sehingga Anda dapat terhubung ke Lingkungan Dev Anda menggunakan SSH:
     - Menginstal plugin Session Manager untuk AWS CLI
     - Memodifikasi lokal Anda AWS Config dan menambahkan CodeCatalyst profil sehingga Anda dapat melakukan login SSO. Untuk informasi selengkapnya, lihat Menyiapkan untuk menggunakan AWS CLI dengan CodeCatalyst.
     - Memodifikasi konfigurasi SSH lokal Anda dan menambahkan konfigurasi yang diperlukan untuk menghubungkan ke Lingkungan Dev Anda menggunakan SSH.
     - Menambahkan skrip di `~/.aws/codecatalyst-dev-env` direktori yang digunakan oleh klien SSH untuk terhubung ke Lingkungan Dev Anda. Skrip ini memanggil CodeCatalyst StartDevEnvironmentSession API dan menggunakan AWS Systems Manager Session Manager plugin untuk membuat AWS Systems Manager sesi dengan Lingkungan Dev Anda yang digunakan oleh klien SSH lokal untuk terhubung dengan aman ke Lingkungan Dev jarak jauh.
   - Masuk ke Amazon CodeCatalyst menggunakan AWS SSO menggunakan perintah kedua. Perintah ini meminta dan mengambil kredensi sehingga skrip dalam

~/.aws/codecatalyst-dev-env direktori dapat memanggil API. CodeCatalyst StartDevEnvironmentSession Perintah ini harus dijalankan setiap kali kredensialmu kedaluwarsa. Ketika Anda menjalankan perintah terakhir di modal (ssh<destination>) Anda akan mendapatkan kesalahan jika kredensional Anda kedaluwarsa atau Anda belum melakukan login SSO seperti yang diinstruksikan dalam langkah ini.

- Connect ke Dev Environment yang Anda tentukan menggunakan SSH menggunakan perintah ketiga. Perintah ini memiliki struktur sebagai berikut:

```
ssh codecatalyst-dev-env=<space-name>=<project-name>=<dev-environment-id>
```

Anda juga dapat menggunakan perintah ini untuk melakukan tindakan lain yang diizinkan oleh klien SSH, seperti penerusan port atau mengunggah dan mengunduh file:

- Penerusan port:

```
ssh -L <local-port>:127.0.0.1:<remote-port> codecatalyst-dev-env=<space-name>=<project-name>=<dev-environment-id>
```

- Mengunggah file ke direktori home di Lingkungan Dev Anda:

```
scp -O </path-to-local-file> codecatalyst-dev-env=<space-name>=<project-name>=<dev-environment-id>:</path-to-remote-file-or-directory>
```

Mengonfigurasi devfile untuk Lingkungan Dev

Devfile adalah standar terbuka yang membantu Anda menyesuaikan Lingkungan Pengembang pengembangan di seluruh tim Anda. Devfile adalah file YAMAL yang mengkodifikasi alat pengembangan yang Anda butuhkan. Dengan mengonfigurasi devfile, Anda dapat menentukan terlebih dahulu alat proyek dan pustaka aplikasi yang Anda butuhkan dan Amazon CodeCatalyst menginstalnya ke Lingkungan Dev untuk Anda. Devfile khusus untuk repositori tempat pembuatannya, dan Anda dapat membuat devfile terpisah untuk setiap repositori. Lingkungan Dev Anda mendukung perintah dan peristiwa, dan menyediakan image devfile universal default.

Jika Anda membuat proyek menggunakan cetak biru kosong, Anda dapat membuat devfile secara manual. Jika Anda membuat proyek menggunakan cetak biru yang berbeda, CodeCatalyst buat devfile secara otomatis. `/projects`Direktori Dev Environment menyimpan file yang ditarik dari repositori sumber dan devfile. `/home`Direktori, yang kosong saat Anda pertama kali membuat

Lingkungan Dev, menyimpan file yang Anda buat saat menggunakan Lingkungan Dev Anda. Segala sesuatu di `/projects` dan `/home` direktori Lingkungan Dev disimpan terus-menerus.

> **Note**
>
> `/home`Folder hanya berubah jika Anda mengubah nama devfile atau nama komponen devfile. Jika Anda mengubah nama komponen devfile atau devfile, isi direktori diganti dan data `/home` `/home` direktori Anda sebelumnya tidak dapat dipulihkan.

Jika Anda membuat Lingkungan Dev dengan repositori sumber yang tidak berisi devfile di akarnya, atau jika Anda membuat Lingkungan Dev tanpa repositori sumber, devfile universal default diterapkan ke repositori sumber secara otomatis. Gambar devfile universal default yang sama digunakan untuk semua IDE. CodeCatalyst saat ini mendukung devfile versi 2.0.0. Untuk informasi selengkapnya tentang devfile, lihat skema Devfile - Versi 2.0.0.

> **Note**
>
> Anda hanya dapat menyertakan gambar kontainer publik di devfile Anda.

Perhatikan bahwa Lingkungan Dev yang terhubung dengan VPC hanya mendukung gambar devfile berikut:

- Gambar universal
- Gambar ECR Amazon pribadi, jika repositori berada di wilayah yang sama dengan VPC

Topik

- Komponen Devfile

Mengedit devfile repositori untuk Lingkungan Dev

Gunakan prosedur berikut untuk mengedit devfile repositori untuk Lingkungan Dev.

Mengedit devfile repositori untuk Lingkungan Dev di CodeCatalyst

Untuk mengedit devfile repositori

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek yang berisi repositori sumber yang ingin Anda edit devfile.
3. Di panel navigasi, pilih Kode.
4. Pilih Source Repositories.
5. Pilih repositori sumber yang berisi devfile yang ingin Anda edit.
6. Dari daftar file, pilih `devfile.yaml` file.
7. Pilih Edit.
8. Edit devfile.
9. Pilih Komit, atau buat permintaan tarik sehingga anggota tim dapat meninjau dan menyetujui perubahan.

> **Note**
>
> Jika Anda mengedit devfile Anda, Anda harus memulai ulang devfile agar perubahan diterapkan. Ini bisa dilakukan dengan berlari`/aws/mde/mde start --location devfile.yaml`. Jika ada masalah saat memulai devfile Anda, itu akan masuk ke mode pemulihan. Namun, jika Anda mengedit devfile yang terkait dengan Lingkungan Dev yang terhubung dengan VPC, Anda harus memulai ulang Lingkungan Dev agar perubahan diterapkan.

Anda dapat meninjau devfile mana yang digunakan dengan menjalankan`/aws/mde/mde status`. Bidang lokasi memiliki jalur devfile relatif terhadap `/projects` folder lingkungan.

```
{
            "status": "STABLE",
```

```
            "location": "devfile.yaml"
        }
```

Anda juga dapat memindahkan devfile default `/projects/devfile.yaml` ke repositori kode sumber Anda. Untuk memperbarui lokasi devfile, gunakan perintah berikut:`/aws/mde/mde start --location repository-name/devfile.yaml`.

Mengedit devfile repositori untuk Lingkungan Dev dalam IDE

Untuk mengubah konfigurasi Dev Environment, Anda harus mengedit devfile. Kami menyarankan Anda mengedit devfile dalam IDE yang didukung dan kemudian memperbarui Lingkungan Dev Anda, tetapi Anda juga dapat mengedit devfile dari root repositori sumber di. CodeCatalyst Jika Anda mengedit devfile dalam IDE yang didukung, Anda harus melakukan dan mendorong perubahan Anda ke repositori sumber atau membuat permintaan tarik sehingga anggota tim dapat meninjau dan menyetujui pengeditan devfile.

- Mengedit devfile repositori untuk Lingkungan Dev di AWS Cloud9
- Mengedit devfile repositori untuk Lingkungan Pengembang di VS Code
- Mengedit devfile repositori untuk Lingkungan Dev di JetBrains

Fitur Devfile didukung oleh CodeCatalyst

CodeCatalyst mendukung fitur devfile berikut pada versi 2.0.0. Untuk informasi selengkapnya tentang devfile, lihat skema Devfile - Versi 2.0.0.

Fitur	Tipe
`exec`	Perintah
`postStart`	Peristiwa
`container`	Komponen
`args`	Properti Komponen
`env`	Properti Komponen
`mountSources`	Properti Komponen

Fitur	Tipe
`volumeMounts`	Properti Komponen

Contoh devfile untuk Lingkungan Dev

Berikut ini adalah contoh devfile sederhana.

```
schemaVersion: 2.0.0
metadata:
  name: al2
components:
  - name: test
    container:
      image: public.ecr.aws/amazonlinux/amazonlinux:2
      mountSources: true
      command: ['sleep', 'infinity']
  - name: dockerstore
commands:
  - id: setupscript
    exec:
      component: test
      commandLine: "chmod +x script.sh"
      workingDir: /projects/devfiles
  - id: executescript
    exec:
      component: test
      commandLine: "/projects/devfiles/script.sh"
  - id: yumupdate
    exec:
      component: test
      commandLine: "yum -y update --security"
events:
  postStart:
    - setupscript
    - executescript
    - yumupdate
```

Startup Devfile, perintah, dan log peristiwa ditangkap dan disimpan di`/aws/mde/logs`. Untuk men-debug perilaku devfile, mulai Lingkungan Dev Anda menggunakan devfile yang berfungsi dan akses log.

Memecahkan masalah devfile repositori menggunakan mode pemulihan

Jika ada masalah saat memulai devfile Anda, itu akan masuk ke mode pemulihan sehingga Anda masih dapat terhubung ke lingkungan Anda dan memperbaiki devfile Anda. Saat dalam mode pemulihan, berjalan `/aws/mde/mde status` tidak akan berisi lokasi devfile Anda.

```
{
            "status": "STABLE"
        }
```

Anda dapat memeriksa kesalahan di log di bawah`/aws/mde/logs`, memperbaiki devfile, dan mencoba menjalankan `/aws/mde/mde start` lagi.

Menentukan gambar devfile universal untuk Lingkungan Dev

Gambar universal default mencakup bahasa pemrograman yang paling umum digunakan dan alat terkait yang dapat digunakan untuk IDE Anda. Jika tidak ada gambar yang ditentukan, CodeCatalyst berikan gambar ini dan berisi alat yang dikelola oleh CodeCatalyst. Untuk tetap diberitahu tentang rilis gambar baru, lihatBerlangganan notifikasi gambar universal dengan SNS.

> **Note**
>
> `public.ecr.aws/aws-mde/universal-image:latest`Gambar mendapatkan `public.ecr.aws/aws-mde/universal-image:3.0` gambar.

Amazon CodeCatalyst mendukung gambar devfile berikut.

Versi gambar	Pengidentifikasi gambar
`Universal image 1.0`	`public.ecr.aws/aws-mde/universal-image:1.0`
`Universal image 2.0`	`public.ecr.aws/aws-mde/universal-image:2.0`
`Universal image 3.0`	`public.ecr.aws/aws-mde/universal-image:3.0`

> **Note**
>
> Jika Anda menggunakan AWS Cloud9, pelengkapan otomatis tidak akan berfungsi untuk PHP, Ruby, dan CSS setelah memutakhirkan ke. `universal-image:3.0`

Topik

Berlangganan notifikasi gambar universal dengan SNS

CodeCatalyst menyediakan layanan pemberitahuan gambar universal. Anda dapat menggunakannya untuk berlangganan topik Amazon Simple Notification Service (SNS) yang memberi tahu Anda saat pembaruan gambar CodeCatalyst universal telah dirilis. Untuk informasi selengkapnya tentang topik SNS, lihat Apa itu Layanan Pemberitahuan Sederhana Amazon? .

Setiap kali gambar universal baru dirilis, kami mengirimkan pemberitahuan ke pelanggan; bagian ini menjelaskan cara berlangganan pembaruan gambar CodeCatalyst universal.

Contoh pesan

```
{
    "Type": "Notification",
    "MessageId": "123456789",
    "TopicArn": "arn:aws:sns:us-east-1:1234657890:universal-image-updates",
    "Subject": "New Universal Image Release",
    "Message": {
        "v1": {
            "Message": "A new version of the Universal Image has been released. You are
 now able to launch new DevEnvironments using this image.",
            "image ": {
                "release_type": "MAJOR VERSION",
                "image_name": "universal-image",
                "image_version": "2.0",
                "image_uri": "public.ecr.aws/amazonlinux/universal-image:2.0"
            }
```

```
        }
    },
    "Timestamp": "2021-09-03T19:05:57.882Z",
    "UnsubscribeURL": "example url"
}
```

Untuk berlangganan pembaruan gambar CodeCatalyst universal menggunakan konsol Amazon SNS

1. Buka konsol Amazon SNS ke Dasbor.
2. Di bilah navigasi, pilih Wilayah AWS.
3. Di panel navigasi, pilih Langganan, lalu pilih Buat langganan.
4. Di Topik ARN, masukkan. `arn:aws:sns:us-east-1:089793673375:universal-image-updates`
5. Di Protokol, pilih Email.
6. Di Endpoint, berikan alamat email. Alamat email ini akan digunakan untuk menerima notifikasi.
7. Pilih Buat langganan.
8. Anda akan menerima email konfirmasi dengan baris subjek "AWS Pemberitahuan - Konfirmasi Berlangganan". Buka email dan pilih Konfirmasi berlangganan.

Untuk berhenti berlangganan pembaruan gambar CodeCatalyst universal menggunakan konsol Amazon SNS

1. Buka konsol Amazon SNS ke Dasbor.
2. Di bilah navigasi, pilih Wilayah AWS.
3. Di panel navigasi, pilih Langganan lalu pilih langganan yang ingin Anda hentikan berlangganan.
4. Pilih Tindakan, lalu pilih Hapus langganan.
5. Pilih Hapus.

Versi runtime image 1.0 universal

Tabel berikut mencantumkan runtime yang tersedia untuk`universal-image:1.0`.

`universal-image:1.0`versi runtime

Nama runtime	Versi	Versi minor mayor dan terbaru tertentu
aws cli	2.11	`aws-cli: 2.x`
docker compose	2.16	`docker-compose: 2.x`
dotnet	6.0	`dotnet: 6.x`
	7.0	`dotnet: 7.x`
golang	1.19	`golang: 1.x`
jawa	corretto11	`java: corretto11.x`
	corretto17	`java: corretto17.x`
nodejs	14.20	`nodejs: 14.x`
	16.19	`nodejs: 16.x`
openssl	1.0	`openssl: 1.x`
	1.1	
php	7.2	`php: 7.x`
python	3.9	`python: 3.x`
	3.10	
ruby	3.1	`ruby: 3.x`
terraform	1.4	`terraform: 1.x`

Versi runtime image 2.0 universal

Tabel berikut mencantumkan runtime yang tersedia untuk`universal-image:2.0`.

`universal-image:2.0`versi runtime

Nama runtime	Versi	Versi minor mayor dan terbaru tertentu
aws cli	2.11	`aws-cli: 2.x`
docker compose	2.17	`docker-compose: 2.x`
dotnet	6.0	`dotnet: 6.x`
	7.0	`dotnet: 7.x`
golang	1.20	`golang: 1.x`
jawa	corretto11	`java: corretto11.x`
	corretto17	`java: corretto17.x`
nodejs	16.19	`nodejs: 16.x`
openssl	1.0	`openssl: 1.x`
	1.1	
php	7.2	`php: 7.x`
python	3.9	`python: 3.x`
	3.10	
ruby	3.2	`ruby: 3.x`
terraform	1.4	`terraform: 1.x`

Versi runtime gambar universal 3.0

Tabel berikut mencantumkan runtime yang tersedia untuk`universal-image:3.0`.

`universal-image:3.0`versi runtime

Nama runtime	Versi	Versi minor mayor dan terbaru tertentu
aws cli	2.11	`aws-cli: 2.x`
docker compose	2.17	`docker-compose: 2.x`
dotnet	6.0	`dotnet: 6.x`
	7.0	`dotnet: 7.x`
golang	1.21	`golang: 1.x`
jawa	corretto11	`java: corretto11.x`
	corretto17	`java: corretto17.x`
nodejs	18.17	`nodejs: 18.x`
	20.6	`nodejs: 20.x`
openssl	3.0	`openssl: 3.x`
php	8.2	`php: 8.x`
python	3.9	`python: 3.x`
	3.11	
ruby	3.2	`ruby: 3.x`
terraform	1.5	`terraform: 1.x`

Perintah Devfile

Saat ini, CodeCatalyst hanya mendukung `exec` perintah di devfile Anda. Untuk informasi selengkapnya, lihat Menambahkan perintah dalam dokumentasi Devfile.io.

Contoh berikut menunjukkan cara menentukan `exec` perintah di devfile Anda.

```
commands:
  - id: setupscript
    exec:
      component: test
      commandLine: "chmod +x script.sh"
      workingDir: /projects/devfiles
  - id: executescript
    exec:
      component: test
      commandLine: "./projects/devfiles/script.sh"
  - id: updateyum
    exec:
      component: test
      commandLine: "yum -y update --security"
```

Setelah Anda terhubung ke Lingkungan Dev Anda, Anda dapat menjalankan perintah yang ditentukan melalui terminal.

```
/aws/mde/mde command <command-id>
/aws/mde/mde command executescript
```

Untuk perintah yang berjalan lama, Anda dapat menggunakan bendera streaming `-s` untuk menampilkan eksekusi perintah secara real time.

```
/aws/mde/mde -s command <command-id>
```

> **Note**
>
> `command-id`harus huruf kecil.

Parameter Exec didukung oleh CodeCatalyst

CodeCatalyst mendukung `exec` parameter berikut pada devfile versi 2.0.0.

- `commandLine`

- `component`

- `id`

- `workingDir`

Acara Devfile

Saat ini, CodeCatalyst hanya mendukung `postStart` acara di devfile Anda. Untuk informasi selengkapnya, lihat postStartObjectdi dokumentasi Devfile.io.

Contoh berikut menunjukkan cara menambahkan binding `postStart` event di devfile Anda.

```
commands:
  - id: executescript
    exec:
      component: test
      commandLine: "./projects/devfiles/script.sh"
  - id: updateyum
    exec:
      component: test
      commandLine: "yum -y update --security"
events:
  postStart:
    - updateyum
    - executescript
```

Setelah startup, Dev Environment Anda akan menjalankan `postStart` perintah yang ditentukan dalam urutan yang ditentukan. Jika perintah gagal, Dev Environment akan terus berjalan dan output eksekusi disimpan dalam log di bawah`/aws/mde/logs`.

Komponen Devfile

Saat ini, CodeCatalyst hanya mendukung `container` komponen di devfile Anda. Untuk informasi selengkapnya, lihat Menambahkan komponen dalam dokumentasi Devfile.io.

Contoh berikut menunjukkan cara menambahkan perintah startup ke container Anda di devfile Anda.

```
components:
  - name: test
    container:
```

```
    image: public.ecr.aws/amazonlinux/amazonlinux:2
    command: ['sleep', 'infinity']
```

> **Note**
>
> Ketika kontainer memiliki perintah entri berumur pendek, Anda harus menyertakan `command: ['sleep', 'infinity']` agar kontainer tetap berjalan.

CodeCatalyst juga mendukung properti berikut dalam komponen kontainer Anda:`args`,`env`,`mountSources`, dan`volumeMounts`.

Mengaitkan koneksi VPC ke Lingkungan Pengembang

Koneksi VPC adalah CodeCatalyst sumber daya yang berisi semua konfigurasi yang diperlukan untuk alur kerja Anda untuk mengakses VPC. Administrator ruang angkasa dapat menambahkan koneksi VPC mereka sendiri di konsol CodeCatalyst Amazon atas nama anggota luar angkasa. Dengan menambahkan koneksi VPC, anggota ruang dapat menjalankan tindakan alur kerja dan membuat Lingkungan Pengembang yang mematuhi aturan jaringan dan dapat mengakses sumber daya di VPC terkait.

Anda hanya dapat mengaitkan Lingkungan Pengembang ke koneksi VPC setelah pembuatan Lingkungan Dev. Anda tidak dapat mengubah koneksi VPC yang terkait dengan Lingkungan Dev setelah Anda membuatnya. Jika Anda ingin menggunakan koneksi VPC yang berbeda, Anda harus menghapus Lingkungan Dev Anda saat ini dan membuat yang baru.

> **Important**
>
> Lingkungan Pengembang dengan koneksi VPC tidak mendukung repositori sumber pihak ketiga yang ditautkan. CodeCatalyst

Perhatikan bahwa Dev Environments memanfaatkan beberapa AWS sumber daya dan layanan pada saat pembuatan. Ini berarti bahwa Dev Environments terhubung ke AWS layanan berikut:

- Amazon CodeCatalyst
- AWS SSM

- AWS KMS
- Amazon ECR
- Amazon CloudWatch
- Amazon ECS

> **Note**
>
> AWS Toolkit tidak mendukung pembuatan Lingkungan Dev dengan koneksi VPC terkait. Perhatikan juga bahwa jika Anda menggunakan IDE selain AWS Cloud9, Anda mungkin mengalami waktu pemuatan sekitar lima menit.

Anda harus memiliki peran administrator Space atau peran pengguna Daya untuk mengelola koneksi VPC di tingkat ruang. Untuk informasi selengkapnya tentang VPC, lihat Mengelola VPC Amazon CodeCatalyst di Panduan CodeCatalyst Administrator.

Kuota untuk Lingkungan Pengembang di CodeCatalyst

Tabel berikut menjelaskan kuota dan batas untuk Lingkungan Pengembang di Amazon. CodeCatalyst Untuk informasi selengkapnya tentang kuota di Amazon CodeCatalyst, lihatKuota untuk CodeCatalyst.

Jumlah jam Dev Environment per bulan	Jam Dev Environment dipengaruhi oleh batas penyimpanan keseluruhan untuk ruang Anda. Untuk informasi lebih lanjut, lihat Harga danMemecahkan masalah dengan Lingkungan Dev.
Jumlah penyimpanan Dev Environment per ruang	Dev Environment menyimpan kami dipengaruhi oleh batas penyimpanan keseluruhan untuk ruang Anda. Untuk informasi lebih lanjut, lihat Harga danMemecahkan masalah dengan Lingkungan Dev.
Jumlah komputasi Dev Environment	Komputasi Dev Environment dipengaruhi oleh batas penyimpanan keseluruhan untuk ruang

	Anda. Untuk informasi lebih lanjut, lihat Harga danMemecahkan masalah dengan Lingkungan Dev.

Publikasikan dan bagikan paket perangkat lunak di CodeCatalyst

Amazon CodeCatalyst berisi layanan repositori paket yang dikelola sepenuhnya yang memudahkan tim pengembangan Anda untuk menyimpan dan berbagi paket perangkat lunak yang digunakan untuk pengembangan aplikasi dengan aman. Paket-paket ini disimpan dalam repositori paket, yang dibuat dan diatur dalam proyek di. CodeCatalyst

CodeCatalyst mendukung format paket berikut:

- npm

Paket dalam repositori paket dapat ditemukan dan dibagi antara anggota proyek yang berisi repositori.

Untuk menambahkan paket ke repositori, konfigurasikan manajer paket untuk menggunakan titik akhir repositori (URL). Anda kemudian dapat menggunakan manajer paket untuk mempublikasikan paket ke repositori.

Anda dapat mengonfigurasi CodeCatalyst alur kerja untuk mempublikasikan paket ke, dan menggunakan paket dari, repositori CodeCatalyst paket. Untuk informasi selengkapnya tentang menggunakan paket dalam alur kerja, lihatMenerbitkan dan mengimpor paket menggunakan alur kerja.

Anda dapat membuat paket dalam satu repositori paket tersedia untuk repositori lain dalam proyek yang sama dengan menambahkannya sebagai repositori upstream. Semua versi paket yang tersedia untuk repositori hulu juga tersedia untuk repositori hilir.

Anda dapat membuat paket sumber terbuka tersedia untuk repositori Anda dengan menghubungkan CodeCatalyst repositori publik dan eksternal ke sana. Untuk informasi selengkapnya tentang repositori upstream dan menghubungkan ke repositori eksternal, termasuk daftar repositori yang didukung, lihat. Mengkonfigurasi dan menggunakan repositori upstream

Topik

- Menghubungkan ke repositori eksternal publik
- Menerbitkan dan memodifikasi paket
- Menggunakan npm
- Kuota untuk paket

Konsep paket

Berikut adalah beberapa konsep dan istilah yang perlu diketahui saat mengelola, menerbitkan, atau mengkonsumsi paket CodeCatalyst.

Paket

Paket adalah bundel yang mencakup perangkat lunak dan metadata yang diperlukan untuk menginstal perangkat lunak dan menyelesaikan dependensi apa pun. CodeCatalyst mendukung format paket npm.

Paket terdiri dari:

- Nama (misalnya, `webpack` adalah nama paket npm populer)
- Namespace opsional (misalnya, `@types` di `@types/node`)
- Satu set versi (misalnya,`1.0.0`,`1.0.1`,`1.0.2`)
- Metadata tingkat paket (misalnya, tag dist npm)

Ruang nama Package

Beberapa format paket mendukung nama paket hierarkis untuk mengatur paket ke dalam grup logis dan untuk membantu menghindari tabrakan nama. Paket yang memiliki nama yang sama dapat disimpan di ruang nama yang berbeda. Misalnya, npm mendukung cakupan, dan paket npm `@types/node` memiliki cakupan `@types` dan nama. `node` Ada banyak nama paket lainnya dalam cakupan `@types`. Dalam CodeCatalyst, ruang lingkup (“tipe”) disebut sebagai namespace paket, dan nama (“node”) disebut sebagai nama paket. Jika Anda tidak memiliki cara untuk mengelompokkan nama paket, akan lebih sulit untuk menghindari tabrakan nama.

Versi Package

Versi paket mengidentifikasi versi spesifik dari sebuah paket, seperti `@types/node@12.6.9`. Format nomor versi dan semantik bervariasi untuk format paket yang berbeda. Sebagai contoh, npm

paket versi harus sesuai dengan spesifikasi Versioning Semantik. Dalam CodeCatalyst, versi paket terdiri dari pengenal versi, package-version-level metadata, dan satu set aset.

Aset

Aset adalah file individual yang disimpan di dalamnya CodeCatalyst yang terkait dengan versi paket, seperti `.tgz` file npm.

Package repositori

Sebuah repositori CodeCatalyst paket berisi satu set paket, yang berisi versi paket, yang masing-masing memetakan ke satu set aset. Setiap repositori paket menyediakan titik akhir untuk mengambil dan menerbitkan paket menggunakan alat seperti Node.js CLI (). `npm` Anda dapat membuat hingga 1.000 repositori paket di setiap ruang.

Anda dapat menautkan repositori paket ke repositori lain dengan menggunakan repositori upstream. Saat Anda menautkan repositori paket sebagai repositori upstream, Anda dapat menggunakan paket di repositori tertaut melalui repositori yang dikonfigurasi. Untuk informasi selengkapnya, lihat Repositori hulu.

Gateway repositori adalah jenis khusus dari repositori paket yang menarik dan menyimpan paket dari otoritas paket eksternal resmi. Untuk informasi selengkapnya, lihat Repositori gerbang.

Repositori gerbang

Repositori gateway adalah jenis khusus dari repositori paket yang terhubung ke otoritas paket resmi eksternal yang didukung. Saat Anda menambahkan repositori gateway sebagai repositori upstream, Anda dapat menggunakan paket dari otoritas paket resmi yang sesuai. Repositori hilir Anda tidak berkomunikasi dengan repositori publik, sebaliknya, semuanya dimediasi oleh repositori gateway. Paket yang dikonsumsi dengan cara ini disimpan di repositori gateway dan repositori hilir yang menerima permintaan asli.

Repositori gateway sudah ditentukan sebelumnya, tetapi harus dibuat di setiap proyek yang akan digunakan. Daftar berikut berisi setiap repositori gateway yang dapat dibuat CodeCatalyst dan otoritas paket yang terhubung dengannya.

- npm-public-registry-gatewaymenyediakan paket npm dari npmjs.com.

Repositori hulu

Anda dapat menggunakan CodeCatalyst untuk membuat hubungan hulu antara dua repositori paket. Repositori paket adalah hulu dari yang lain ketika versi paket yang dikandungnya dapat diakses dari titik akhir repositori paket dari repositori hilir. Dengan hubungan hulu, isi dari dua repositori paket secara efektif digabungkan dari sudut pandang klien.

Misalnya, jika manajer paket meminta versi paket yang tidak ada di repositori, kemudian CodeCatalyst akan mencari repositori upstream yang dikonfigurasi untuk versi paket. Repositori upstream dicari dalam urutan mereka dikonfigurasi, dan setelah paket ditemukan, CodeCatalyst akan berhenti mencari.

Mengkonfigurasi dan menggunakan repositori paket

Di CodeCatalyst, paket disimpan dan dikelola di dalam repositori paket. Untuk mempublikasikan paket ke CodeCatalyst atau menggunakan paket dari CodeCatalyst (atau repositori paket publik yang didukung), Anda harus membuat repositori paket dan menghubungkan manajer paket Anda ke paket tersebut.

Topik

Membuat repositori paket

Lakukan langkah-langkah berikut untuk membuat repositori paket di. CodeCatalyst

Untuk membuat repositori paket

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek di mana Anda ingin membuat repositori paket.
3. Dari panel navigasi, pilih Paket.
4. Pada halaman Package repositories, pilih Create package repository.

5. Di bagian Package repository details, tambahkan yang berikut ini:

    a. Nama repositori. Pertimbangkan untuk menggunakan nama deskriptif dengan detail seperti proyek atau nama tim Anda, atau bagaimana repositori akan digunakan.

    b. (Opsional) Deskripsi repositori. Deskripsi repositori sangat membantu ketika Anda memiliki beberapa repositori di beberapa tim dalam sebuah proyek.

6. Di bagian Edit repositori upstream, tambahkan repositori paket apa pun yang ingin Anda akses melalui repositori paket Anda. CodeCatalyst Anda dapat menambahkan repositori Gateway untuk terhubung ke repositori paket eksternal atau repositori paket lainnya. CodeCatalyst

    - Ketika sebuah paket diminta dari repositori paket, repositori upstream akan dicari dalam urutan mereka muncul dalam daftar ini. Setelah paket ditemukan, CodeCatalyst akan berhenti mencari. Untuk mengubah urutan repositori upstream, Anda dapat drag dan drop repositori dalam daftar, atau menggunakan tombol Reorder.

7. Pilih Buat untuk membuat repositori paket Anda.

Menghubungkan ke repositori paket

Untuk mempublikasikan paket ke CodeCatalyst atau untuk mengkonsumsi dari CodeCatalyst, Anda harus mengkonfigurasi manajer paket Anda dengan informasi titik akhir repositori paket dan kredenal Anda. CodeCatalyst Jika Anda belum membuat repositori, Anda dapat melakukannya dengan mengikuti instruksi di. Membuat repositori paket

Untuk petunjuk tentang cara menghubungkan manajer paket npm ke repositori CodeCatalyst paket, lihat. Mengkonfigurasi dan menggunakan npm

Mengedit repositori paket

Lakukan langkah-langkah berikut untuk mengedit deskripsi repositori paket dan repositori upstream.

Untuk mengedit repositori paket

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek yang berisi repositori paket yang ingin Anda edit.
3. Dari panel navigasi, pilih Paket.
4. Pada halaman Package repositories, pilih repositori yang ingin Anda hapus.
5. Pilih dropdown Tindakan dan pilih Edit.

6. Edit deskripsi repositori dan repositori hulu. Untuk informasi selengkapnya tentang repositori upstream, lihat. Mengkonfigurasi dan menggunakan repositori upstream
7. Pilih Simpan.

Menghapus repositori paket

Lakukan langkah-langkah berikut untuk menghapus repositori paket di. CodeCatalyst

Untuk menghapus repositori paket

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek yang berisi repositori paket yang ingin Anda hapus.
3. Dari panel navigasi, pilih Paket.
4. Pada halaman Package repositories, pilih repositori yang ingin Anda hapus.
5. Pilih dropdown Tindakan dan pilih Hapus.
6. Tinjau informasi yang diberikan tentang efek menghapus repositori paket.
7. Masuk `delete` ke kolom input dan pilih Hapus.

Mengkonfigurasi dan menggunakan repositori upstream

Anda dapat menghubungkan kedua repositori gateway, dan repositori paket lainnya, sebagai upstream ke repositori paket Anda. Hal ini memungkinkan klien manajer paket untuk mengakses paket yang terkandung dalam lebih dari satu repositori paket dengan menggunakan endpoint repositori paket tunggal. Berikut ini adalah manfaat utama menggunakan repositori hulu:

- Anda hanya perlu mengonfigurasi manajer paket Anda dengan satu titik akhir repositori untuk menarik dari berbagai sumber.
- Paket yang dikonsumsi dari repositori upstream disimpan di repositori hilir Anda, yang memastikan paket Anda tersedia meskipun repositori upstream mengalami pemadaman yang tidak terduga.

Anda dapat menambahkan repositori upstream saat Anda membuat repositori paket. Anda juga dapat menambah atau menghapus repositori upstream dari repositori paket yang ada di konsol. CodeCatalyst

Saat Anda menambahkan repositori gateway sebagai repositori upstream, repositori paket terhubung ke repositori paket publik repositori gateway yang sesuai. Untuk daftar repositori paket publik yang didukung, lihat. Repositori paket eksternal yang didukung dan repositori gateway mereka

Anda dapat menautkan beberapa repositori bersama-sama sebagai repositori upstream. Misalnya, tim Anda membuat repositori bernama `project-repo` dan sudah menggunakan repositori lain bernama yang telah npm-public-registry-gatewayditambahkan sebagai repositori upstream, yang terhubung ke repositori npm publik,. `team-repo` `npmjs.com` Anda dapat menambahkan `team-repo` sebagai repositori upstream ke. `project-repo` Dalam hal ini, Anda hanya perlu mengkonfigurasi manajer paket Anda untuk digunakan `project-repo` untuk menarik paket dari`project-repo`,`team-repo`,`npm-public-registry-gateway`, dan`npmjs.com`.

Topik

- Menambahkan repositori upstream
- Mengedit urutan pencarian repositori hulu
- Meminta versi paket dengan repositori hulu
- Menghapus repositori upstream

Menambahkan repositori upstream

Menambahkan repositori paket publik atau repositori CodeCatalyst paket lain sebagai repositori upstream ke repositori hilir Anda membuat semua paket di repositori upstream tersedia untuk manajer paket yang terhubung ke repositori hilir.

Untuk menambahkan repositori upstream

1. Di panel navigasi, pilih Paket.
2. Pada halaman Package repositories, pilih repositori paket yang ingin Anda tambahkan repositori upstream.
3. Pilih menu dropdown Actions dan pilih Edit.
4. Di bagian Repositori Upstream, pilih Add upstream repository.
5. Pilih bilah pencarian untuk membuka dan mencari daftar repositori yang tersedia. Anda dapat menambahkan repositori paket publik yang didukung atau repositori lain sebagai CodeCatalyst repositori upstream. Ketika Anda menemukan repositori yang ingin Anda tambahkan, pilih dari daftar.

> **Note**
>
> Di repositori Gateway, ada repositori. npm-public-registry-gateway Untuk terhubung ke otoritas paket eksternal publik, seperti npmjs.com, CodeCatalyst menggunakan repositori gateway sebagai repositori perantara yang mencari dan menyimpan paket yang ditarik dari repositori eksternal. Ini menghemat waktu dan transfer data karena semua repositori paket dalam proyek akan menggunakan paket dari repositori gateway.

6. Ketika Anda telah memilih semua repositori yang ingin Anda tambahkan sebagai repositori upstream, pilih Tambah.
7. Untuk informasi selengkapnya tentang mengubah urutan pencarian repositori hulu, lihat. Mengedit urutan pencarian repositori hulu

Ketika Anda telah menambahkan repositori upstream, Anda dapat menggunakan manajer paket yang terhubung ke repositori lokal Anda untuk mengambil paket dari repositori upstream. Anda tidak perlu memperbarui konfigurasi manajer paket Anda. Untuk informasi selengkapnya tentang meminta versi paket dari repositori upstream, lihat. Meminta versi paket dengan repositori hulu

Mengedit urutan pencarian repositori hulu

CodeCatalyst mencari repositori upstream dalam urutan pencarian yang dikonfigurasi. Ketika sebuah paket ditemukan, CodeCatalyst berhenti mencari. Anda dapat mengubah urutan repositori upstream yang dicari untuk paket.

Untuk mengedit urutan pencarian repositori hulu

1. Di panel navigasi, pilih Paket.
2. Pada halaman Repositori, pilih repositori paket yang urutan pencarian repositori hulu yang ingin Anda edit.
3. Pilih dropdown Tindakan dan pilih Edit.
4. Di bagian Repositori Upstream, Anda dapat melihat repositori upstream dan urutan pencariannya. Untuk mengubah urutan pencarian, seret dan lepas repositori dalam daftar, atau gunakan tombol Susun Ulang.
5. Setelah selesai mengedit urutan pencarian repositori upstream, pilih Simpan.

Meminta versi paket dengan repositori hulu

Contoh berikut menunjukkan skenario yang mungkin ketika manajer paket meminta paket dari repositori CodeCatalyst paket yang memiliki repositori upstream.

Untuk contoh ini, manajer paket, sepertinpm, meminta versi paket dari repositori paket bernama `downstream` yang memiliki beberapa repositori upstream. Ketika paket diminta, hal berikut dapat terjadi:

- Jika `downstream` berisi versi paket yang diminta, akan dikembalikan ke klien.
- Jika `downstream` tidak berisi versi paket yang diminta, CodeCatalyst cari di repositori upstream, dalam `downstream` urutan pencarian yang dikonfigurasi. Jika versi paket ditemukan, referensi untuk itu akan disalin ke `downstream`, dan versi paket dikembalikan ke klien.
- Jika tidak ada `downstream` atau repositori upstream yang berisi versi paket, `Not Found` respons HTTP 404 dikembalikan ke klien.

Jumlah maksimum repositori hulu langsung yang diizinkan untuk satu repositori adalah 10. Jumlah maksimum repositori yang CodeCatalyst dicari saat versi paket diminta adalah 25.

Retensi paket dari repositori hulu

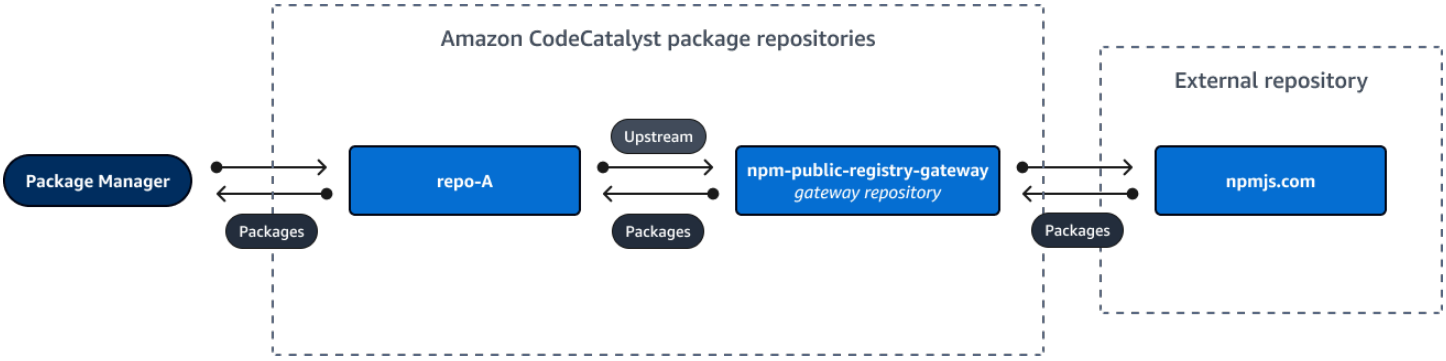
Jika versi paket yang diminta ditemukan di repositori upstream, referensi untuk itu dipertahankan dan selalu tersedia di repositori yang memintanya. Ini memastikan bahwa Anda memiliki akses ke paket Anda jika ada pemadaman tak terduga dari repositori upstream. Versi paket yang dipertahankan tidak terpengaruh oleh salah satu dari berikut ini:

- Menghapus repositori hulu.
- Memutuskan koneksi repositori hulu dari repositori hilir.
- Menghapus versi paket dari repositori hulu.
- Mengedit versi paket di repositori hulu (misalnya, dengan menambahkan aset baru ke dalamnya).

Mengambil paket melalui hubungan hulu

CodeCatalyst dapat mengambil paket melalui beberapa repositori tertaut yang disebut repositori upstream. Jika repositori CodeCatalyst paket memiliki koneksi upstream ke repositori CodeCatalyst paket lain yang memiliki koneksi upstream ke repositori gateway, permintaan paket yang tidak

ada di repositori upstream disalin dari repositori eksternal. Misalnya, pertimbangkan konfigurasi berikut: repositori bernama `repo-A` memiliki koneksi upstream ke repositori gateway,. `npm-public-registry-gateway` `npm-public-registry-gateway`memiliki koneksi upstream ke repositori paket publik, https://npmjs.com.



Jika `npm` dikonfigurasi untuk menggunakan `repo-A` repositori, menjalankan `npm install` memulai penyalinan paket dari https://npmjs.com ke. `npm-public-registry-gateway` Versi yang dipasang juga ditarik ke dalam `repo-A`. Contoh berikut menginstal `lodash`.

```
$ npm config get registry
https://packages.region.codecatalyst.aws/npm/space-name/proj-name/repo-name/
$ npm install lodash
+ lodash@4.17.20
added 1 package from 2 contributors in 6.933s
```

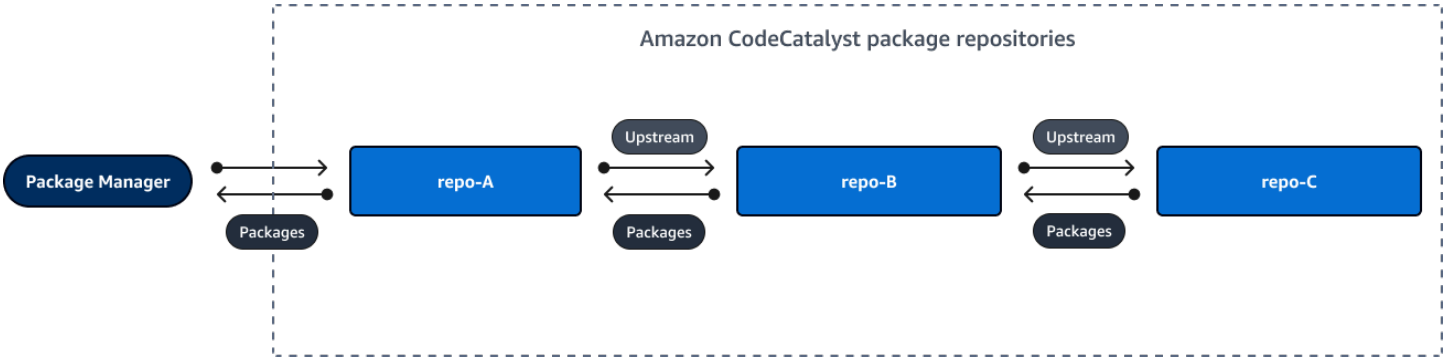
Setelah berjalan`npm install`, hanya `repo-A` berisi versi terbaru (`lodash 4.17.20`) karena itulah versi yang diambil oleh `npm` from`repo-A`.

Karena `npm-public-registry-gateway` memiliki koneksi upstream eksternal ke https://npmjs.com, semua versi paket yang diimpor dari https://npmjs.com disimpan din`npm-public-registry-gateway`. Versi paket ini dapat diambil oleh repositori hilir mana pun dengan koneksi hulu ke. `npm-public-registry-gateway`

Isi `npm-public-registry-gateway` menyediakan cara bagi Anda untuk melihat semua paket dan versi paket yang diimpor dari https://npmjs.com dari waktu ke waktu.

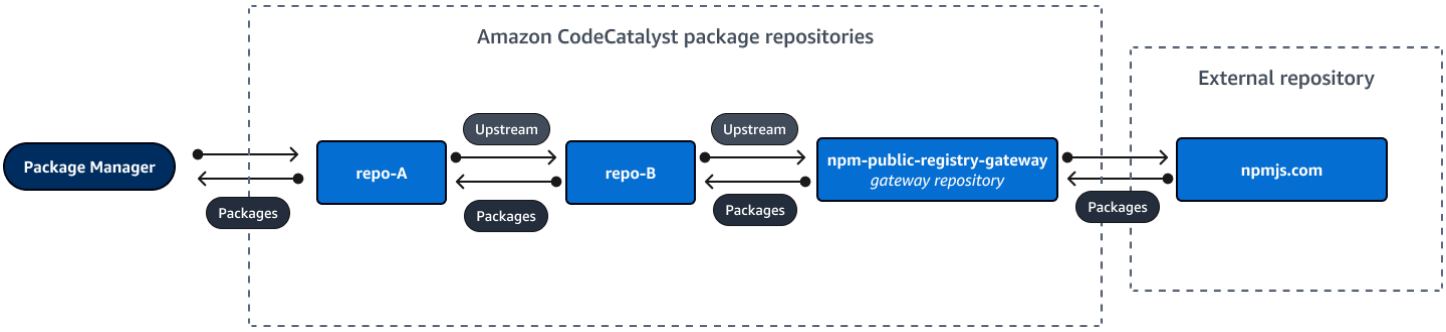
Retensi paket dalam repositori menengah

CodeCatalyst memungkinkan Anda untuk rantai repositori hulu. Misalnya, `repo-A` dapat memiliki `repo-B` sebagai repositori upstream dan `repo-B` dapat memiliki `repo-C` sebagai repositori upstream. Konfigurasi ini membuat versi paket di `repo-B` dan `repo-C` tersedia dari `repo-A`.



Ketika manajer paket terhubung ke repositori `repo-A` dan mengambil versi paket dari repositori`repo-C`, versi paket tidak dipertahankan dalam repositori. `repo-B` Versi paket hanya disimpan di repositori hilir terjauh, yang dalam contoh ini adalah. `repo-A` Itu tidak disimpan di repositori perantara apa pun. Ini juga berlaku untuk rantai yang lebih panjang; misalnya, jika ada empat repositori:`repo-A`,`repo-B`,`repo-C`, dan`repo-D`, dan manajer paket yang terhubung untuk `repo-A` mengambil versi paket dari`repo-D`, versi paket akan dipertahankan `repo-A` tetapi tidak di atau. `repo-B` `repo-C`

Perilaku retensi paket serupa saat menarik versi paket dari repositori paket publik, kecuali bahwa versi paket selalu dipertahankan di repositori gateway yang memiliki koneksi hulu langsung ke repositori publik. Misalnya, `repo-A` memiliki `repo-B` sebagai repositori hulu. `repo-B`memiliki `npm-public-registry-gateway` sebagai repositori upstream, yang memiliki koneksi upstream ke repositori publik, npmjs.com; lihat diagram di bawah ini.



Jika manajer paket terhubung untuk **`repo-A`** meminta versi paket tertentu, lodash 4.17.20 misalnya, dan versi paket tidak ada di salah satu dari tiga repositori, itu akan diambil dari npmjs.com. Ketika lodash 4.17.20 diambil, itu dipertahankan karena itu adalah repositori hilir terjauh dan **`repo-A`** **`npm-public-registry-gateway`** karena memiliki koneksi hulu ke repositori eksternal publik, npmjs.com. lodash 4.17.20 tidak dipertahankan `repo-B` karena itu adalah repositori perantara.

Menghapus repositori upstream

Jika Anda tidak lagi ingin mengakses paket dalam repositori upstream, Anda dapat menghapus repositori upstream dari repositori paket.

> **Warning**
>
> Saat Anda menghapus repositori upstream, Anda dapat memutuskan rantai hubungan hulu, yang dapat merusak proyek atau build Anda.

Untuk menghapus repositori upstream

1. Di panel navigasi, pilih Paket.
2. Pada halaman Package repositories, pilih repositori paket dari mana Anda ingin menghapus repositori upstream.
3. Pilih dropdown Tindakan dan pilih Edit.
4. Di bagian Repositori Upstream, temukan repositori upstream yang ingin Anda hapus dan pilih Hapus.
5. Setelah selesai menghapus repositori upstream, pilih Simpan.

Menghubungkan ke repositori eksternal publik

Anda dapat menghubungkan repositori CodeCatalyst paket ke repositori eksternal publik yang didukung dengan menambahkan repositori gateway yang sesuai sebagai repositori upstream. Repositori gateway bertindak sebagai repositori perantara yang mencari dan menyimpan paket yang ditarik dari repositori eksternal. Ini menghemat waktu dan transfer data karena semua repositori paket dalam proyek menggunakan paket dari repositori gateway.

Untuk terhubung ke repositori publik menggunakan repositori gateway

1. Di panel navigasi, pilih Paket.
2. Pada halaman Package repositories, di repositori Gateway, Anda dapat melihat daftar repositori gateway yang didukung dan deskripsinya. Untuk menggunakan repositori gateway, pertama-tama Anda harus membuatnya. Jika repositori gateway telah dibuat, tanggal dan waktu pembuatannya akan ditampilkan. Jika belum, pilih Buat untuk membuatnya.
3. Pilih repositori paket yang ingin Anda sambungkan ke repositori publik.

4. Pilih menu tarik-turun Tindakan dan pilih Edit.
5. Untuk terhubung ke repositori publik, tambahkan repositori gateway yang sesuai dengan repositori publik yang ingin Anda sambungkan sebagai repositori upstream.

   Di bagian Edit repositori hulu, pilih Tambahkan repositori. CodeCatalyst
6. Bagian repositori Gateway mencantumkan semua repositori gateway yang tersedia. Ketika Anda menemukan repositori gateway yang sesuai dengan repositori publik dan eksternal yang ingin Anda sambungkan, pilih dari daftar dan pilih Tambah.
7. Ketika sebuah paket diminta dari repositori, CodeCatalyst mencari repositori upstream dalam urutan yang muncul dalam daftar Edit repositori upstream. Ketika sebuah paket ditemukan, CodeCatalyst berhenti mencari. Untuk mengubah urutan repositori hulu, seret dan lepas repositori ke dalam daftar, atau gunakan panah urutan ulang.
8. Setelah selesai menambahkan dan memesan repositori upstream, pilih Simpan.

Ketika Anda telah menambahkan repositori gateway sebagai repositori upstream, Anda dapat menggunakan manajer paket yang terhubung ke repositori lokal Anda untuk mengambil paket dari repositori paket eksternal publik yang sesuai dengannya. Anda tidak perlu memperbarui konfigurasi manajer paket Anda. Paket yang dikonsumsi dengan cara ini disimpan di repositori gateway dan repositori paket lokal Anda. Untuk informasi selengkapnya tentang meminta versi paket dari repositori upstream, lihat. Meminta versi paket dengan repositori hulu

Repositori paket eksternal yang didukung dan repositori gateway mereka

CodeCatalyst mendukung penambahan koneksi hulu ke otoritas paket resmi berikut dengan repositori gateway.

Jenis paket repositori	Deskripsi	Nama repositori gateway
npm	registri publik npm	`npm-public-registry-gateway`

Menerbitkan dan memodifikasi paket

Paket dalam CodeCatalyst adalah bundel perangkat lunak dan metadata yang diperlukan untuk menyelesaikan dependensi dan menginstal perangkat lunak. CodeCatalyst mendukung format paket

npm. Bagian ini memberikan informasi tentang penerbitan, melihat, dan menghapus paket, dan memperbarui status versi paket.

Topik

Menerbitkan paket ke CodeCatalyst repositori paket

Anda dapat mempublikasikan versi dari semua jenis paket yang didukung ke repositori CodeCatalyst paket dengan menggunakan alat pengelola paket. Langkah-langkah untuk mempublikasikan versi paket adalah sebagai berikut:

Untuk mempublikasikan versi paket ke CodeCatalyst repositori paket

1. Jika belum, buat repositori paket.
2. Hubungkan manajer paket Anda ke repositori paket Anda. Untuk petunjuk tentang cara menghubungkan manajer paket npm ke repositori CodeCatalyst paket, lihat. Mengkonfigurasi dan menggunakan npm
3. Gunakan pengelola paket yang terhubung untuk mempublikasikan versi paket Anda.

Daftar Isi

Publikasi dan repositori hulu

Di CodeCatalyst, Anda tidak dapat mempublikasikan versi paket yang ada di repositori upstream yang dapat dijangkau atau repositori publik. Misalnya, misalkan Anda ingin mempublikasikan paket npm,, ke repositori paket`lodash@1.0`,`myrepo`, dan terhubung ke npmjs.com melalui repositori gateway yang `myrepo` dikonfigurasi sebagai repositori upstream. Jika `lodash@1.0` ada di repositori

upstream atau di npmjs.com, CodeCatalyst menolak setiap upaya untuk mempublikasikannya dengan mengeluarkan kesalahan konflik 409. `myrepo` Ini membantu mencegah Anda menerbitkan paket dengan nama dan versi yang sama secara tidak sengaja sebagai paket di repositori upstream, yang dapat mengakibatkan perilaku yang tidak terduga.

Anda masih dapat mempublikasikan versi berbeda dari nama paket yang ada di repositori upstream. Misalnya, jika `lodash@1.0` ada dalam repositori hulu, tapi `lodash@1.1` tidak, Anda dapat memublikasikan `lodash@1.1` ke repositori hilir.

Paket privat dan repositori publik

CodeCatalyst tidak mempublikasikan paket yang disimpan dalam CodeCatalyst repositori ke repositori publik, seperti npmjs.com. CodeCatalyst mengimpor paket dari repositori publik ke CodeCatalyst repositori, tetapi tidak memindahkan paket ke arah yang berlawanan. Paket yang Anda publikasikan ke CodeCatalyst repositori tetap pribadi dan hanya tersedia untuk CodeCatalyst proyek di mana repositori berada.

Penimpaan aset paket

Anda tidak dapat menerbitkan ulang aset paket yang sudah ada dengan konten berbeda yang terkandung di dalamnya. Karena npm hanya mendukung satu aset per versi paket, untuk memodifikasi versi paket yang diterbitkan, Anda harus menghapusnya terlebih dahulu.

Melihat detail versi paket

Anda dapat menggunakan CodeCatalyst konsol untuk melihat detail tentang versi paket tertentu.

Untuk melihat detail versi paket

1. Di panel navigasi, pilih Paket.
2. Pada halaman Package repositories, pilih repositori yang berisi versi paket yang ingin Anda lihat detailnya.
3. Cari versi paket di tabel Paket. Anda dapat menggunakan bilah pencarian untuk memfilter paket berdasarkan nama paket. Pilih paket dari daftar.
4. Di halaman Package details, pilih Versions, lalu pilih versi yang ingin Anda lihat.

Menghapus versi paket

Anda dapat menghapus versi paket dari halaman detail versi Package di CodeCatalyst konsol.

Untuk menghapus versi paket

1. Di panel navigasi, pilih Paket.
2. Pada halaman Package repositories, pilih repositori yang berisi versi paket yang ingin Anda hapus.
3. Cari dan pilih paket dari tabel.
4. Pada halaman Package details, pilih Versions dan pilih versi yang ingin Anda hapus.
5. Pada halaman Detail versi Package, pilih Tindakan versi lalu pilih Hapus.
6. Masukkan hapus ke bidang teks dan pilih Hapus.

Memperbarui status versi paket

Setiap versi paket CodeCatalyst memiliki status yang menjelaskan status saat ini dan ketersediaan versi paket. Anda dapat mengubah status versi paket di CodeCatalyst konsol. Untuk informasi selengkapnya tentang kemungkinan nilai status versi paket dan artinya, lihatStatus versi paket.

Untuk memperbarui status versi paket

1. Di panel navigasi, pilih Paket.
2. Pada halaman Package repositories, pilih repositori yang berisi versi paket yang ingin Anda perbarui statusnya.
3. Cari dan pilih paket dari tabel.
4. Pada halaman Package details, pilih Versions lalu pilih versi yang ingin Anda lihat.
5. Pada halaman detail versi Package, pilih Actions lalu pilih Unlist, Archive, atau Dispose. Untuk informasi tentang setiap status versi paket, lihatStatus versi paket.
6. Masukkan teks konfirmasi ke dalam bidang teks, lalu pilih Batalkan Daftar, Arsipkan, atau Buang, tergantung pada status yang Anda perbarui.

Status versi paket

Berikut ini adalah nilai yang mungkin untuk status versi paket. Anda dapat mengubah status versi paket di konsol. Untuk informasi selengkapnya, lihat Memperbarui status versi paket.

- Diterbitkan: Versi paket berhasil diterbitkan dan dapat diminta oleh manajer paket. Versi paket akan disertakan dalam daftar versi paket yang dikembalikan ke manajer paket; misalnya, dalam outputnpm `view <package-name> versions`. Semua aset versi paket tersedia dari repositori.

- Tidak terdaftar: Aset versi paket tersedia untuk diunduh dari repositori, tetapi versi paket tidak termasuk dalam daftar versi yang dikembalikan ke manajer paket. Misalnya, untuk paket npm, output `npm view <package-name> versions` tidak menyertakan versi paket. Ini berarti logika resolusi ketergantungan npm tidak memilih versi paket karena versi tidak muncul dalam daftar versi yang tersedia. Namun, jika versi paket Tidak Terdaftar sudah direferensikan dalam sebuah `npm package-lock.json` file, itu masih dapat diunduh dan diinstal; misalnya, saat menjalankan. `npm ci`
- Diarsipkan: Aset versi paket tidak dapat diunduh. Versi paket tidak akan dimasukkan dalam daftar versi yang dikembalikan ke manajer paket. Karena aset tidak tersedia, konsumsi versi paket oleh klien diblokir. Jika build aplikasi Anda bergantung pada versi yang diperbarui ke Archived, build akan gagal, kecuali versi paket telah di-cache secara lokal. Anda tidak dapat menggunakan pengelola paket atau alat build untuk menerbitkan ulang versi paket yang Diarsipkan karena masih ada di repositori. Namun, Anda dapat mengubah status versi paket kembali ke Tidak Terdaftar atau Diterbitkan di konsol.
- Disposed: Versi paket tidak muncul dalam daftar, dan aset tidak dapat diunduh dari repositori. Perbedaan utama antara Disposed dan Archived adalah bahwa dengan status Disposed, aset versi paket dihapus secara permanen oleh. CodeCatalyst Untuk alasan ini, Anda tidak dapat memindahkan versi paket dari Dibuang ke Diarsipkan, Tidak Terdaftar, atau Dipublikasikan. Versi paket tidak dapat digunakan karena aset telah dihapus. Ketika versi paket telah ditandai sebagai Disposed, Anda tidak ditagih untuk penyimpanan aset paket.

Selain status dalam daftar sebelumnya, versi paket juga dapat dihapus. Setelah dihapus, versi paket tidak ada di repositori dan Anda dapat dengan bebas menerbitkan ulang versi paket itu dengan menggunakan manajer paket atau alat build.

Mengedit kontrol asal paket

Di Amazon CodeCatalyst, versi paket dapat ditambahkan ke repositori paket dengan menerbitkannya secara langsung, menariknya ke bawah dari repositori upstream, atau menelannya dari repositori publik eksternal. Jika Anda mengizinkan versi paket ditambahkan baik dengan penerbitan langsung dan menelan dari repositori publik, maka Anda rentan terhadap serangan substitusi ketergantungan. Untuk informasi selengkapnya, lihat Serangan substitusi ketergantungan. Untuk melindungi diri Anda dari serangan substitusi dependensi, konfigurasikan kontrol asal paket pada paket dalam repositori untuk membatasi bagaimana versi paket itu dapat ditambahkan ke repositori.

Anda harus mempertimbangkan untuk mengonfigurasi kontrol asal paket untuk membuat versi baru dari paket yang berbeda berasal dari kedua sumber internal, seperti penerbitan langsung,

dan sumber eksternal, seperti repositori publik. Secara default, kontrol asal paket dikonfigurasi berdasarkan bagaimana versi pertama paket ditambahkan ke repositori.

Pengaturan kontrol asal paket

Dengan kontrol asal paket, Anda dapat mengonfigurasi bagaimana versi paket dapat ditambahkan ke repositori. Daftar berikut mencakup pengaturan dan nilai kontrol asal paket yang tersedia.

Publikasikan

Pengaturan ini mengonfigurasi apakah versi paket dapat dipublikasikan langsung ke repositori menggunakan manajer paket atau alat serupa.

- ALLOW: Versi Package dapat dipublikasikan secara langsung.
- BLOCK: Versi Package tidak dapat dipublikasikan secara langsung.

Hulu

Pengaturan ini mengonfigurasi apakah versi paket dapat dicerna dari eksternal, repositori publik, atau disimpan dari repositori hulu saat diminta oleh manajer paket.

- ALLOW: Setiap versi paket dapat dipertahankan dari repositori lain yang dikonfigurasi sebagai CodeCatalyst repositori upstream atau dicerna dari sumber publik dengan koneksi eksternal.
- BLOCK: Versi Package tidak dapat dipertahankan dari CodeCatalyst repositori lain yang dikonfigurasi sebagai repositori upstream atau dicerna dari sumber publik dengan koneksi eksternal.

Pengaturan kontrol asal paket default

Kontrol asal paket default untuk sebuah paket akan didasarkan pada bagaimana versi pertama paket itu ditambahkan ke repositori paket.

- Jika versi paket pertama diterbitkan secara langsung oleh manajer paket, pengaturannya adalah Publish: ALLOW dan Upstream: BLOCK.
- Jika versi paket pertama dicerna dari sumber publik, pengaturannya adalah Publish: BLOCK dan Upstream: ALLOW.

Skenario kontrol akses paket umum

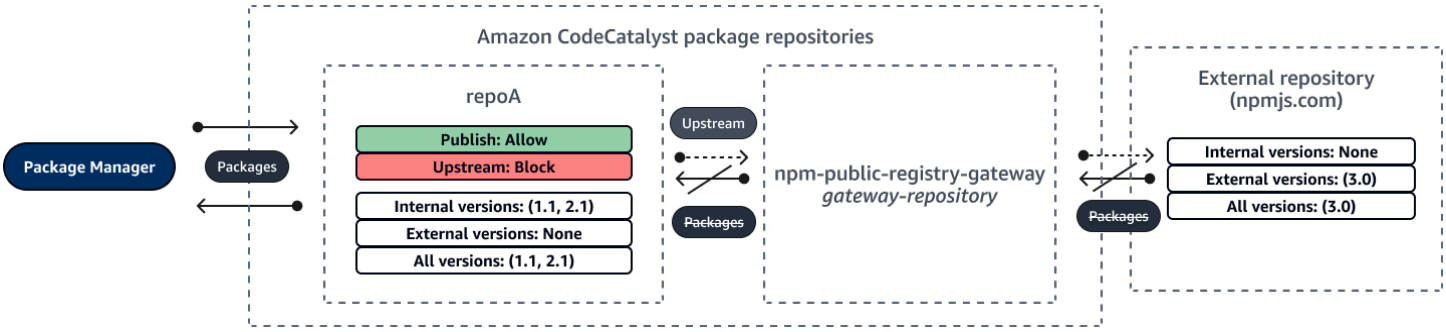
Bagian ini menjelaskan beberapa skenario umum ketika versi paket ditambahkan ke repositori CodeCatalyst paket. Pengaturan kontrol asal paket diatur untuk paket baru tergantung pada bagaimana versi paket pertama ditambahkan.

Dalam skenario berikut, paket internal diterbitkan langsung dari manajer paket ke repositori Anda, seperti paket yang Anda pertahankan. Paket eksternal adalah paket yang ada di repositori publik yang dapat dicerna ke dalam repositori Anda dengan koneksi eksternal.

Versi paket eksternal diterbitkan untuk paket internal yang ada

Dalam skenario ini, pertimbangkan paket internal, PackageA. Tim Anda menerbitkan versi paket pertama untuk PackageA ke repositori paket. CodeCatalyst Karena ini adalah versi paket pertama untuk paket itu, pengaturan kontrol asal paket secara otomatis diatur ke Publish: Allow and Upstream: Block. Setelah paket diterbitkan di repositori Anda, paket dengan nama yang sama dipublikasikan ke repositori publik yang terhubung ke repositori paket Anda. CodeCatalyst Ini bisa berupa percobaan serangan substitusi ketergantungan terhadap paket internal, atau bisa juga kebetulan. Terlepas dari itu, kontrol asal paket dikonfigurasi untuk memblokir konsumsi versi eksternal baru untuk melindungi diri dari serangan potensial.

Pada gambar berikut, RepoA adalah repositori CodeCatalyst paket Anda dengan koneksi eksternal ke repositori publik. Repositori Anda berisi versi 1.1 dan 2.1 dari PackageA, tetapi versi 3.0 dipublikasikan ke repositori publik. Biasanya, RepOA akan menelan versi 3.0 setelah paket diminta oleh manajer paket. Karena konsumsi paket disetel ke Block, versi 3.0 tidak tertelan ke dalam repositori CodeCatalyst paket Anda dan tidak tersedia untuk manajer paket yang terhubung dengannya.

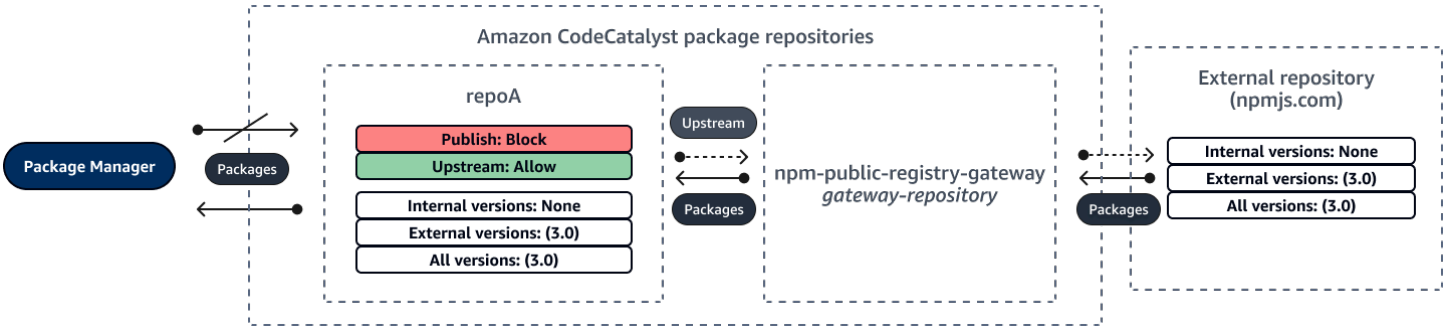


Versi paket internal diterbitkan untuk paket eksternal yang ada

Dalam skenario ini, sebuah paket, PackageB, ada secara eksternal di repositori publik yang telah Anda sambungkan ke repositori Anda. Ketika manajer paket yang terhubung ke repositori Anda

meminta PackageB, versi paket diserap ke dalam repositori Anda dari repositori publik. Karena ini adalah versi paket pertama dari PackageB yang ditambahkan ke repositori Anda, pengaturan asal paket dikonfigurasi untuk Publish: BLOCK dan Upstream: ALLOW. Kemudian, Anda mencoba mempublikasikan versi dengan nama paket yang sama ke repositori. Anda mungkin tidak mengetahui paket publik dan mencoba mempublikasikan paket yang tidak terkait dengan nama yang sama, atau Anda mungkin mencoba menerbitkan versi yang ditambal, atau Anda mungkin mencoba untuk secara langsung mempublikasikan versi paket persis yang sudah ada secara eksternal. CodeCatalyst menolak versi yang Anda coba terbitkan, tetapi Anda dapat secara eksplisit mengganti penolakan dan mempublikasikan versinya, jika perlu.

Pada gambar berikut, RepoA adalah repositori CodeCatalyst paket Anda dengan koneksi eksternal ke repositori publik. Repositori paket Anda berisi versi 3.0 yang dicerna dari repositori publik. Anda ingin mempublikasikan versi 1.2 ke repositori paket Anda. Biasanya, Anda dapat mempublikasikan versi 1.2 ke RePoA, tetapi karena penerbitan diatur ke Blokir, versi 1.2 tidak dapat dipublikasikan.



Menerbitkan versi paket yang ditambal dari paket eksternal yang ada

Dalam skenario ini, sebuah paket, PackageB, ada secara eksternal di repositori publik yang telah Anda sambungkan ke repositori paket Anda. Ketika manajer paket yang terhubung ke repositori Anda meminta PackageB, versi paket diserap ke dalam repositori Anda dari repositori publik. Karena ini adalah versi paket pertama dari PackageB yang ditambahkan ke repositori Anda, pengaturan asal paket dikonfigurasi untuk Publish: BLOCK dan Upstream: ALLOW. Tim Anda memutuskan untuk menerbitkan versi paket yang ditambal dari paket ini ke repositori. Untuk dapat mempublikasikan versi paket secara langsung, tim Anda mengubah pengaturan kontrol asal paket menjadi Publish: ALLOW dan Upstream: BLOCK. Versi paket ini sekarang dapat dipublikasikan langsung ke repositori Anda dan dicerna dari repositori publik. Setelah tim Anda menerbitkan versi paket yang ditambal, tim Anda mengembalikan setelan asal paket ke Publish: BLOCK dan Upstream: ALLOW.

Mengedit kontrol asal paket

Kontrol asal paket dikonfigurasi secara otomatis berdasarkan bagaimana versi paket pertama dari sebuah paket ditambahkan ke repositori paket. Untuk informasi selengkapnya, lihat Pengaturan kontrol asal paket default. Untuk menambah atau mengedit kontrol asal paket untuk paket dalam repositori CodeCatalyst paket, lakukan langkah-langkah dalam prosedur berikut.

Untuk menambah atau mengedit kontrol asal paket

1. Di panel navigasi, pilih Paket.
2. Pilih repositori paket yang berisi paket yang ingin Anda edit.
3. Dalam tabel Paket, cari dan pilih paket yang ingin Anda edit.
4. Dari halaman ringkasan paket, pilih Edit kontrol asal.
5. Di kontrol Origin, pilih kontrol asal paket yang ingin Anda atur untuk paket ini. Kedua pengaturan kontrol asal paket, Publish dan Upstream, harus diatur pada saat yang sama.
   - Untuk mengizinkan penerbitan versi paket secara langsung, di Publikasikan, pilih Izinkan. Untuk memblokir penerbitan versi paket, pilih Blokir.
   - Untuk memungkinkan konsumsi paket dari repositori eksternal dan menarik paket dari repositori upstream, di sumber Upstream, pilih Izinkan. Untuk memblokir semua konsumsi dan penarikan versi paket dari repositori eksternal dan upstream, pilih Blokir.
6. Pilih Simpan.

Publikasi dan repositori hulu

Di CodeCatalyst, Anda tidak dapat mempublikasikan versi paket yang ada di repositori upstream yang dapat dijangkau atau repositori publik. Misalnya, misalkan Anda ingin mempublikasikan paket `lodash@1.0` npm ke repositori,`myrepo`, dan `myrepo` memiliki repositori upstream dengan koneksi eksternal ke npmjs.com. Pertimbangkan skenario berikut.

1. Pengaturan kontrol asal paket aktif `lodash` adalah Publish: ALLOW dan Upstream: ALLOW. Jika `lodash@1.0` ada di repositori upstream atau di npmjs.com, CodeCatalyst menolak setiap upaya untuk mempublikasikannya dengan mengeluarkan kesalahan konflik 409. `myrepo` Anda masih dapat mempublikasikan versi yang berbeda, seperti`lodash@1.1`.
2. Pengaturan kontrol asal paket aktif `lodash` adalah Publish: ALLOW dan Upstream: BLOCK. Anda dapat mempublikasikan versi apa pun `lodash` ke repositori Anda yang belum ada karena versi paket tidak dapat dijangkau.

3. Pengaturan kontrol asal paket aktif `lodash` adalah Publish: BLOCK dan Upstream: ALLOW. Anda tidak dapat mempublikasikan versi paket apa pun langsung ke repositori Anda.

Serangan substitusi ketergantungan

Package manager menyederhanakan proses pengemasan dan berbagi kode yang dapat digunakan kembali. Paket-paket ini mungkin paket pribadi yang dikembangkan oleh organisasi untuk digunakan dalam aplikasi mereka, atau mereka mungkin publik, biasanya paket open-source yang dikembangkan di luar organisasi dan didistribusikan oleh repositori paket publik. Saat meminta paket, pengembang mengandalkan manajer paket mereka untuk mengambil versi baru dari dependensi mereka. Serangan substitusi dependensi, juga dikenal sebagai serangan kebingungan ketergantungan, mengeksploitasi fakta bahwa manajer paket biasanya tidak memiliki cara untuk membedakan versi paket yang sah dari versi berbahaya.

Serangan substitusi dependensi termasuk dalam subset serangan yang dikenal sebagai serangan rantai pasokan perangkat lunak. Serangan rantai pasokan perangkat lunak adalah serangan yang memanfaatkan kerentanan di mana saja dalam rantai pasokan perangkat lunak.

Serangan substitusi dependensi dapat menargetkan siapa saja yang menggunakan paket dan paket yang dikembangkan secara internal yang diambil dari repositori publik. Penyerang mengidentifikasi nama paket internal dan kemudian secara strategis menempatkan kode berbahaya dengan nama yang sama di repositori paket publik. Biasanya, kode berbahaya diterbitkan dalam paket dengan nomor versi tinggi. Package manager mengambil kode berbahaya dari feed publik ini karena mereka percaya bahwa paket berbahaya adalah versi terbaru dari paket. Hal ini menyebabkan “kebingungan” atau “substitusi” antara paket yang diinginkan dan paket berbahaya, yang menyebabkan kode dikompromikan.

Untuk mencegah serangan substitusi dependensi, Amazon CodeCatalyst menyediakan kontrol asal paket. Package origin control adalah pengaturan yang mengontrol bagaimana paket dapat ditambahkan ke repositori Anda. Kontrol dikonfigurasi secara otomatis ketika versi paket pertama dari paket baru ditambahkan ke CodeCatalyst repositori Kontrol dapat memastikan versi paket tidak dapat dipublikasikan langsung ke repositori Anda dan dicerna dari sumber publik, melindungi Anda dari serangan substitusi ketergantungan. Untuk informasi selengkapnya tentang kontrol asal paket dan cara mengubahnya, lihatMengedit kontrol asal paket.

Menggunakan npm

Topik-topik ini menjelaskan bagaimana Anda dapat menggunakannpm, manajer paket Node.js, dengan CodeCatalyst.

> **Note**
>
> CodeCatalyst mendukung `node v4.9.1` dan kemudian `npm v5.0.0` dan kemudian.

Topik

Mengkonfigurasi dan menggunakan npm

Untuk menggunakannya npm CodeCatalyst, Anda harus terhubung npm ke repositori paket Anda dan menyediakan token akses pribadi (PAT) untuk otentikasi. Anda dapat melihat instruksi untuk menghubungkan npm ke repositori paket Anda di konsol. CodeCatalyst

Daftar Isi

Mengkonfigurasi npm dengan CodeCatalyst

Petunjuk berikut menjelaskan cara mengautentikasi dan menghubungkan npm ke repositori CodeCatalyst paket Anda. Untuk informasi selengkapnya tentang npm, lihat dokumentasi resmi npm.

Untuk terhubung **npm** ke repositori CodeCatalyst paket Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda.
3. Di panel navigasi, pilih Paket.
4. Pilih repositori paket Anda dari daftar.
5. Pilih Connect to repository.
6. Dalam detail Konfigurasi, di klien Package manager, pilih klien npm.
7. Pilih sistem operasi Anda untuk melihat langkah-langkah konfigurasi yang sesuai.
8. Token akses pribadi (PAT) diperlukan untuk mengautentikasi npm dengan. CodeCatalyst Jika Anda sudah memiliki token, Anda dapat menggunakannya. Jika tidak, Anda dapat membuatnya menggunakan langkah-langkah berikut.

   a. (Opsional): Perbarui nama PAT dan Tanggal kedaluwarsa.

   b. Pilih Buat token.

   c. Salin dan simpan PAT Anda di lokasi yang aman.

   > ⚠ Warning
   >
   > Anda tidak akan dapat melihat atau menyalin PAT Anda lagi setelah Anda menutup kotak dialog. Kredensyal harus berumur pendek untuk meminimalkan lamanya waktu penyerang dapat menggunakan kredensil setelah menyalahgunakannya.

9. Jalankan perintah berikut dari direktori root proyek Anda untuk mengonfigurasi npm dengan repositori paket Anda. Perintah akan melakukan hal berikut:

   - Buat `.npmrc` file tingkat proyek jika proyek Anda tidak memilikinya.
   - Tambahkan informasi titik akhir repositori paket ke file tingkat proyek Anda. `.npmrc`
   - Tambahkan kredensyal Anda (PAT) ke file tingkat pengguna `.npmrc` Anda.

   Ganti nilai-nilai berikut.

> **Note**
>
> Jika Anda menyalin dari instruksi konsol, nilai-nilai dalam perintah berikut diperbarui untuk Anda dan tidak perlu diubah.

- Ganti *nama pengguna* dengan nama CodeCatalyst pengguna Anda.
- Ganti *PAT* dengan CodeCatalyst PAT Anda.
- Ganti *space_name dengan nama* spasi Anda CodeCatalyst .
- Ganti *proj_name dengan nama* CodeCatalyst proyek Anda.
- Ganti *repo_name dengan nama* repositori CodeCatalyst paket Anda.

```
npm set registry=https://packages.region.codecatalyst.aws/npm/space-name/proj-name/repo-name/ --location project
npm set //packages.region.codecatalyst.aws/npm/space-name/proj-name/repo-name/:_authToken=username:PAT
```

Untuk npm 6 atau lebih rendah: Untuk membuat npm selalu meneruskan token auth ke CodeCatalyst, bahkan untuk GET permintaan, setel variabel konfigurasi always-auth dengan sebagai berikut. `npm config set`

```
npm set //packages.region.codecatalyst.aws/npm/space-name/proj-name/repo-name/:always-auth=true --location project
```

Menginstal paket npm dari repositori CodeCatalyst paket

Setelah Anda menghubungkan npm ke repositori Anda dengan mengikuti langkah-langkah di[Mengkonfigurasi npm dengan CodeCatalyst](), Anda dapat menjalankan `npm` perintah di repositori Anda.

Anda dapat menginstal paket npm yang ada di repositori CodeCatalyst paket Anda atau salah satu repositori hulu dengan perintah. `npm install`

```
npm install lodash
```

Menginstal paket npm dari npmjs melalui CodeCatalyst

Anda dapat menginstal paket npm dari npmjs.com melalui repositori dengan mengonfigurasi CodeCatalyst repositori dengan koneksi upstream ke repositori gateway yang terhubung ke npmjs.com,. npm-public-registry-gateway Paket yang diinstal dari npmjs dicerna dan disimpan di repositori gateway, dan repositori paket hilir terjauh.

Untuk menginstal paket dari npmjs

1. Jika Anda belum melakukannya, konfigurasikan `npm` dengan repositori CodeCatalyst paket Anda dengan mengikuti langkah-langkah di. Mengkonfigurasi npm dengan CodeCatalyst
2. Periksa apakah repositori Anda telah menambahkan repositori gateway, npm-public-registry-gateway, sebagai koneksi upstream. Anda dapat memeriksa sumber hulu mana yang ditambahkan atau ditambahkan npm-public-registry-gatewaysebagai sumber hulu dengan mengikuti instruksi Menambahkan repositori upstream dan memilih repositori. npm-public-registry-gateway
3. Instal paket dengan `npm install` perintah.

   ```
   npm install package_name
   ```

Untuk informasi selengkapnya tentang meminta paket dari repositori upstream, lihat. Meminta versi paket dengan repositori hulu

Menerbitkan paket npm ke repositori CodeCatalyst paket Anda

Setelah selesaiMengkonfigurasi npm dengan CodeCatalyst, Anda dapat menjalankan `npm` perintah.

Anda dapat mempublikasikan paket npm ke repositori CodeCatalyst paket dengan perintah. `npm publish`

```
npm publish
```

Untuk selengkapnya tentang cara membuat paket npm, lihat Membuat Modul Node.js di npm Docs.

dukungan perintah npm

Bagian berikut merangkum `npm` perintah yang didukung oleh repositori CodeCatalyst paket, selain mencantumkan perintah tertentu yang tidak didukung.

Topik

Perintah yang didukung yang berinteraksi dengan repositori paket

Bagian ini mencantumkan npm perintah di mana npm klien membuat satu atau lebih permintaan ke registri yang dikonfigurasi (misalnya,`npm config set registry`). Perintah-perintah ini telah diverifikasi untuk berfungsi dengan benar ketika dipanggil terhadap repositori CodeCatalyst paket.

Perintah	Deskripsi
bug	Menebak lokasi URL pelacak bug paket, dan kemudian mencoba membukanya.
ci	Menginstal proyek dari awal.
mencela	Menghentikan penggunaan versi paket.
dist-tag	Memodifikasi tanda distribusi paket.
dokumen	Menebak lokasi URL dokumentasi paket, dan kemudian mencoba membukanya dengan menggunakan parameter `--browser` konfigurasi.
dokter	Menjalankan serangkaian pemeriksaan untuk memvalidasi bahwa instalasi npm Anda dapat mengelola paket Anda JavaScript .
menginstal	Menginstal paket.
install-ci-test	Menginstal proyek dari awal dan menjalank an pengujian. Alias:`npm cit`. Perintah ini menjalankan`npm ci`, diikuti segera oleh`npm test`.

Perintah	Deskripsi
uji instalasi	Menginstal paket dan menjalankan tes. Menjalankannpm `install`, diikuti segera olehnpm `test`.
ketinggalan zaman	Memeriksa registri yang dikonfigurasi untuk menentukan apakah ada paket yang diinstal sudah usang.
ping	Ping registri npm yang dikonfigurasi atau diberikan dan memverifikasi autentikasi.
mempublikasikan	Memublikasikan versi paket ke registri.
perbarui	Menebak lokasi URL repositori paket, dan kemudian mencoba membukanya dengan menggunakan parameter konfigurasi. `--browser`
melihat	Menampilkan metadata paket. Dapat juga digunakan untuk mencetak properti metadata.

Perintah sisi klien yang didukung

Perintah ini tidak memerlukan interaksi langsung dengan repositori paket, jadi CodeCatalyst tidak memerlukan apa pun untuk mendukungnya.

Perintah	Deskripsi
bin (warisan)	Menampilkan `bin` direktori npm.
membangun	Membangun paket.
cache	Memanipulasi cache paket.
penyelesaian	Memungkinkan penyelesaian tab di semua perintah npm.

Perintah	Deskripsi
konfigurasi	Memperbarui isi pengguna dan file `npmrc` global.
dedupe	Mencari pohon paket lokal dan mencoba menyederhanakan struktur dengan memindahkan dependensi lebih jauh ke atas pohon di mana mereka dapat dibagikan secara lebih efektif oleh beberapa paket dependen.
sunting	Mengedit paket yang diinstal. Memilih ketergantungan di direktori kerja saat ini dan membuka direktori paket di editor default.
jelajahi	Menelusuri paket yang diinstal. Memunculkan subshell di direktori paket terinstal yang ditentukan. Jika perintah ditentukan, maka itu dijalankan di subshell, yang kemudian segera dimatikan.
membantu	Mendapat bantuan mengenai npm.
bantuan-pencarian	Menelusuri dokumentasi bantuan npm.
init	Membuat file `package.json` .
tautan	Symlink direktori paket.
ls	Daftar paket yang diinstal.
paket	Membuat tarball dari sebuah paket.
prefix	Menampilkan awalan. Ini adalah direktori induk terdekat untuk berisi `package.json` file, kecuali juga `-g` ditentukan.
memangkas	Menghapus paket yang tidak tercantum pada daftar dependensi paket induk.

Perintah	Deskripsi
membangun kembali	Menjalankan perintah `npm build` pada folder yang cocok.
mulai ulang	Menjalankan skrip stop, restart, dan start paket serta pra-skrip dan post-script terkait.
akar	Mencetak `node_modules` direktori efektif ke standar.
run-skrip	Menjalankan skrip paket arbitrer.
shrinkwrap	Mengunci versi dependensi untuk publikasi.
hapus instalan	Meng-uninstall paket.

Perintah tidak didukung

`npm`Perintah ini tidak didukung oleh repositori CodeCatalyst paket.

Perintah	Deskripsi	Catatan
akses	Menetapkan tingkat akses pada paket yang dipublika sikan.	CodeCatalyst menggunakan model izin yang berbeda dari repositori npmjs publik.
penambah	Menambahkan akun pengguna registri	CodeCatalyst menggunak an model pengguna yang berbeda dari repositori npmjs publik.
audit	Menjalankan audit keamanan.	CodeCatalyst saat ini tidak menjual data kerentanan keamanan.
kait	Mengelola kait npm, termasuk menambahkan, menghapus, mendaftar, dan memperbarui.	CodeCatalyst saat ini tidak mendukung mekanisme

Perintah	Deskripsi	Catatan
		pemberitahuan perubahan apa pun.
Login	Mengautentikasi pengguna. Ini adalah nama lain untuk `npm adduser`.	CodeCatalyst menggunakan model otentikasi yang berbeda dari repositori npmjs publik. Untuk informasi, lihat Mengkonfigurasi npm dengan CodeCatalyst.
logout	Keluar dari registri.	CodeCatalyst menggunakan model otentikasi yang berbeda dari repositori npmjs publik. Tidak ada cara untuk keluar dari CodeCatalyst repositori, tetapi token otentikasi kedaluwarsa setelah waktu kedaluwarsa yang dapat dikonfigurasi. Durasi token default adalah 12 jam.
pemilik	Mengelola pemilik paket.	CodeCatalyst menggunakan model izin yang berbeda dari repositori npmjs publik.
profil	Mengubah pengaturan pada profil registri Anda.	CodeCatalyst menggunakan model pengguna yang berbeda dari repositori npmjs publik.
pencarian	Mencari registri untuk paket yang cocok dengan istilah pencarian.	CodeCatalyst tidak mendukung `search` perintah.

Perintah	Deskripsi	Catatan
bintang	Menandai paket favorit Anda.	CodeCatalyst saat ini tidak mendukung mekanisme favorit apa pun.
bintang	Melihat paket yang ditandai sebagai favorit.	CodeCatalyst saat ini tidak mendukung mekanisme favorit apa pun.
tim	Mengelola tim dan keanggotaan tim.	CodeCatalyst menggunakan model keanggotaan pengguna dan grup yang berbeda dari repositori npmjs publik.
token	Mengelola token autentikasi Anda.	CodeCatalyst menggunakan model yang berbeda untuk mendapatkan token otentikasi. Untuk informasi, lihat Mengkonfigurasi npm dengan CodeCatalyst.
batalkan publikasi	Menghapus paket dari registri.	CodeCatalyst tidak mendukung penghapusan versi paket dari repositori dengan menggunakan klien npm. Anda dapat menghapus paket di konsol.
whoami	Menampilkan nama pengguna npm.	CodeCatalyst menggunakan model pengguna yang berbeda dari repositori npmjs publik.

penanganan tanda npm

registri npm mendukung tanda, yang merupakan alias string untuk versi paket. Anda dapat menggunakan tag untuk memberikan alias alih-alih menggunakan nomor versi. Misalnya, Anda memiliki proyek dengan beberapa aliran pengembangan dan Anda menggunakan tag yang berbeda untuk setiap aliran (misalnya,,`stable`, `betadev`,`canary`). Untuk informasi lebih lanjut, lihat dist-tag di npm Docs.

Secara default, npm menggunakan tanda `latest` untuk mengidentifikasi versi paket saat ini. `npm install` *`pkg`* (tanpa penentu `@`*`version`* atau `@`*`tag`*) menginstal tanda terbaru. Biasanya, proyek hanya menggunakan tag terbaru untuk versi rilis stabil. Tanda lain digunakan untuk versi yang tidak stabil atau pra-rilis.

Mengedit tag dengan klien npm

Tiga `npm dist-tag` perintah (`add`,`rm`, dan`ls`) berfungsi dengan cara yang sama di repositori CodeCatalyst paket seperti yang berfungsi di registri npm default.

tanda npm dan repositori hulu

Saat `npm` meminta tag untuk paket dan versi paket itu juga ada di repositori upstream, CodeCatalyst gabungkan tag sebelum mengembalikannya ke klien. Misalnya, repositori bernama `R` memiliki repositori upstream bernama. `U` Tabel berikut menunjukkan tag untuk paket bernama `web-helper` yang ada di kedua repositori.

Repositori	Nama paket	Tanda paket
R	`web-helper`	terbaru (alias untuk versi 1.0.0)
U	`web-helper`	alfa (alias untuk versi 1.0.1)

Dalam hal ini, ketika klien npm mengambil tag untuk *`web-helper`* paket dari repositori*`R`*, ia menerima tag terbaru dan alpha. Versi yang ditunjukkan oleh tanda tidak akan berubah.

Ketika tag yang sama hadir pada paket yang sama di repositori upstream dan lokal, CodeCatalyst gunakan tag yang terakhir diperbarui. Misalnya, anggap tanda pada webhelper telah dimodifikasi agar terlihat seperti berikut ini.

Repositori	Nama paket	Tanda paket	Terakhir diperbarui
R	`web-helper`	terbaru (alias untuk versi 1.0.0)	1 Januari 2023
U	`web-helper`	terbaru (alias untuk versi 1.0.1)	1 Juni 2023

Dalam hal ini, ketika klien npm mengambil tag untuk paket web-helper dari repositori*R*, tag terbaru akan alias versi 1.0.1 karena diperbarui terakhir. Ini memudahkan untuk menggunakan versi paket baru di repositori upstream yang belum ada di repositori lokal dengan menjalankannya. `npm update`

Kuota untuk paket

Tabel berikut menjelaskan kuota dan batas untuk paket di Amazon CodeCatalyst. Untuk informasi selengkapnya tentang kuota di Amazon CodeCatalyst, lihatKuota untuk CodeCatalyst.

Sumber Daya	Kuota bawaan
Package repositori	Maksimal 1000 per ruang.
Repositori hulu langsung	Maksimal 10 per repositori paket.
Repositori paket upstream dicari	Maksimal 25 repositori upstream dicari per versi paket yang diminta.
Ukuran file aset Package	Maksimal 5GB per paket aset.
Aset paket	Maksimal 150 per versi paket.

Bangun, uji, dan terapkan dengan alur kerja di CodeCatalyst

Setelah menulis kode aplikasi Anda di Lingkungan CodeCatalyst Dev dan mendorongnya ke repositori CodeCatalyst sumber Anda, Anda siap untuk menerapkannya. Cara untuk melakukannya secara otomatis adalah melalui alur kerja.

Alur kerja adalah prosedur otomatis yang menjelaskan cara membangun, menguji, dan menyebarkan kode Anda sebagai bagian dari sistem integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD). Alur kerja mendefinisikan serangkaian langkah, atau tindakan, yang harus diambil selama menjalankan alur kerja. Alur kerja juga mendefinisikan peristiwa, atau pemicu, yang menyebabkan alur kerja dimulai. Untuk menyiapkan alur kerja, Anda membuat file definisi alur kerja menggunakan editor visual atau YAMAL CodeCatalyst konsol.

> **Tip**
>
> Untuk melihat sekilas bagaimana Anda dapat menggunakan alur kerja dalam sebuah proyek, buat proyek dengan cetak biru. Setiap cetak biru menerapkan alur kerja yang berfungsi yang dapat Anda tinjau, jalankan, dan bereksperimen.

Tentang file definisi alur kerja

File definisi alur kerja adalah file YAMAL yang menjelaskan alur kerja Anda. File disimpan dalam `~/.codecatalyst/workflows/` folder di root repositori sumber Anda. File dapat memiliki ekstensi.yml.yaml.

Berikut ini adalah contoh file definisi alur kerja sederhana. Kami menjelaskan setiap baris contoh ini dalam tabel berikut.

```
Name: MyWorkflow
SchemaVersion: 1.0
RunMode: QUEUED
Triggers:
  - Type: PUSH
    Branches:
      - main
Actions:
  Build:
    Identifier: aws/build@v1
```

```
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      Steps:
        - Run: docker build -t MyApp:latest .
```

Garis	Deskripsi
`Name: MyWorkflow`	Menentukan nama alur kerja. Untuk informasi lebih lanjut tentang `Name` properti, lihatProperti tingkat atas.
`SchemaVersion: 1.0`	Menentukan versi skema alur kerja. Untuk informasi lebih lanjut tentang `SchemaVersion` properti, lihatProperti tingkat atas.
`RunMode: QUEUED`	Menunjukkan bagaimana CodeCatalyst menangani beberapa proses. Untuk informasi selengkapnya tentang mode lari, lihatMengonfigurasi perilaku antrian run.
`Triggers:`	Menentukan logika yang akan menyebabkan alur kerja berjalan untuk memulai. Untuk informasi lebih lanjut tentang menggunakan pemicu, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.

`- Type: PUSH`
`  Branches:`
`    - main`
	Menunjukkan bahwa alur kerja harus dimulai setiap kali Anda mendorong kode ke `main` cabang repositori sumber default. Untuk informasi selengkapnya tentang sumber alur kerja, lihatMenghubungkan alur kerja ke repositori sumber.

`Actions:`	Mendefinisikan tugas yang akan dilakukan selama menjalankan alur kerja. Dalam contoh ini, `Actions` bagian mendefinisikan satu tindakan yang disebut`Build`. Untuk informasi

Garis	Deskripsi
	selengkapnya tentang tindakan, lihatMengkonfigurasi tindakan yang dilakukan alur kerja.
`Build:`	Mendefinisikan properti untuk `Build` tindakan. Untuk informasi selengkapnya tentang tindakan build, lihatMembangun dengan alur kerja.
`Identifier: aws/build@v1`	Menentukan identifier hard-code yang unik untuk tindakan build.

`Inputs:`
`  Sources:`
`  - WorkflowSource`
	Menunjukkan bahwa tindakan build harus terlihat di repositori `WorkflowSource` sumber untuk menemukan file yang dibutuhkan untuk menyelesaikan pemrosesannya. Untuk informasi selengkapnya, lihat Menghubungkan alur kerja ke repositori sumber.

`Configuration:`	Berisi properti konfigurasi yang khusus untuk tindakan build.
`Steps:` `  - Run: docker build -t MyApp:latest .`	Memberitahu tindakan build untuk membangun image Docker yang disebut `MyApp` dan menandainya`latest`.

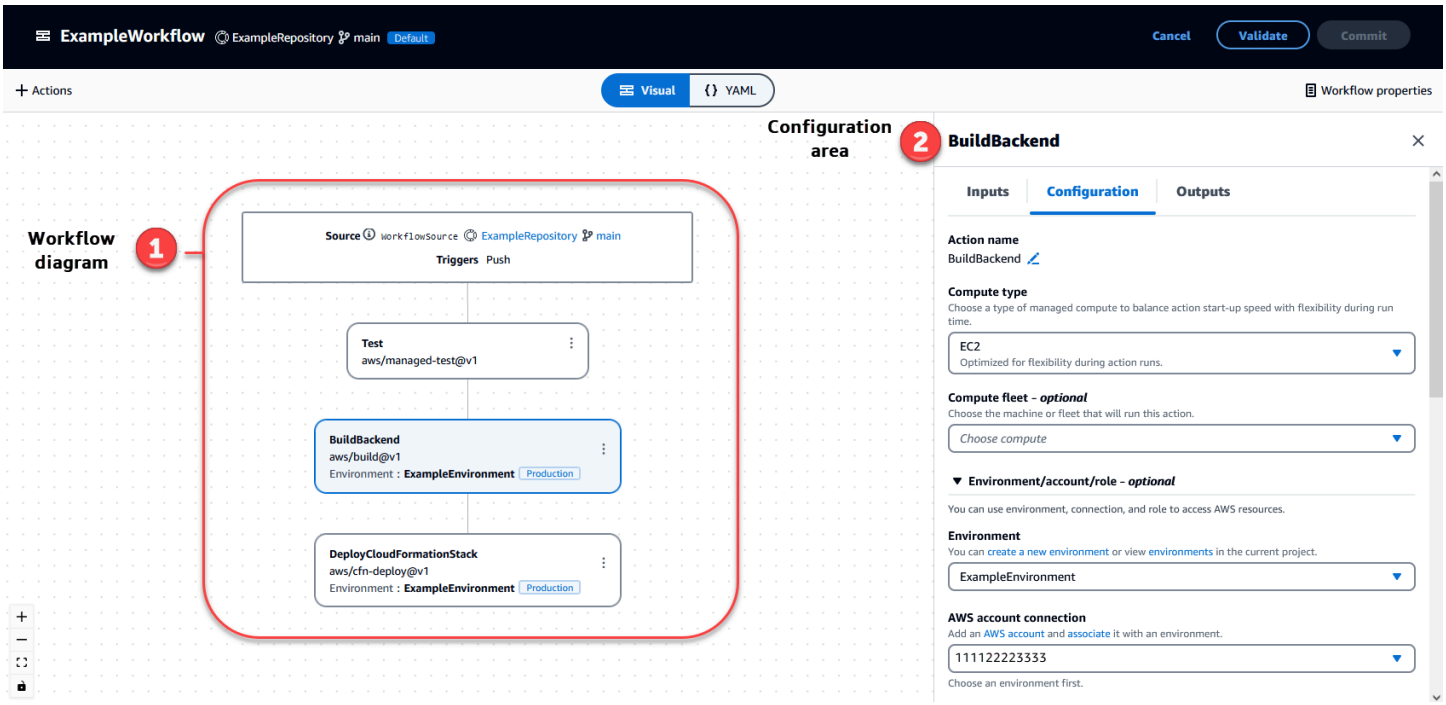
Untuk daftar lengkap semua properti yang tersedia dalam file definisi alur kerja, lihat. Alur kerja definisi YAMAL

Menggunakan editor visual dan YAMAL CodeCatalyst konsol

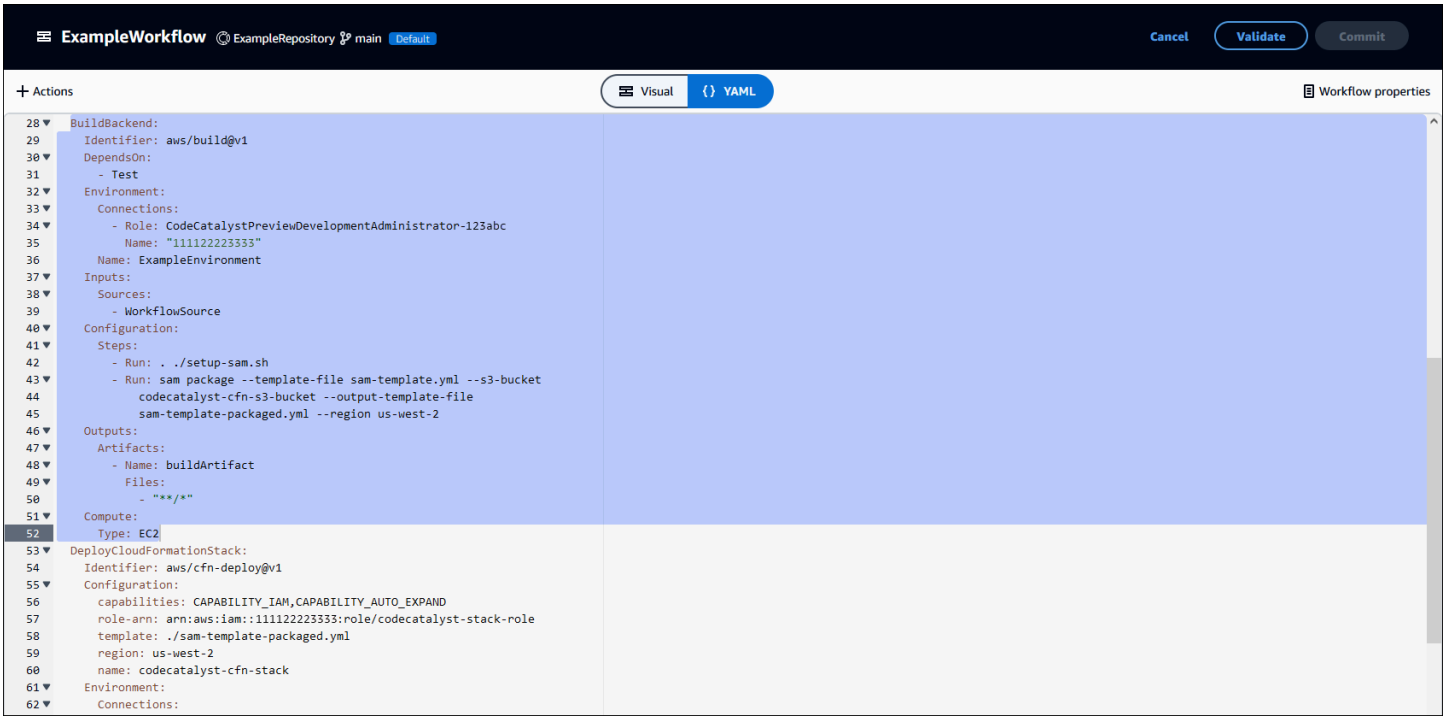
Untuk membuat dan mengedit file definisi alur kerja, Anda dapat menggunakan editor pilihan Anda, tetapi sebaiknya gunakan editor visual CodeCatalyst konsol atau editor YAMAL. Editor ini menawarkan validasi file yang bermanfaat untuk membantu memastikan nama properti YAMAL, nilai, penyarangan, spasi, kapitalisasi, dan sebagainya, benar.

Gambar berikut menunjukkan alur kerja di editor visual. Editor visual menawarkan antarmuka pengguna lengkap untuk membuat dan mengkonfigurasi file definisi alur kerja Anda. Editor visual

menyertakan diagram alur kerja (1) yang menunjukkan komponen utama alur kerja, dan area konfigurasi (2).



Atau, Anda dapat menggunakan editor YAMAL, yang ditunjukkan pada gambar berikutnya. Gunakan editor YAMAL untuk menempelkan blok kode besar (dari tutorial, misalnya), atau untuk menambahkan properti lanjutan yang tidak ditawarkan melalui editor visual.

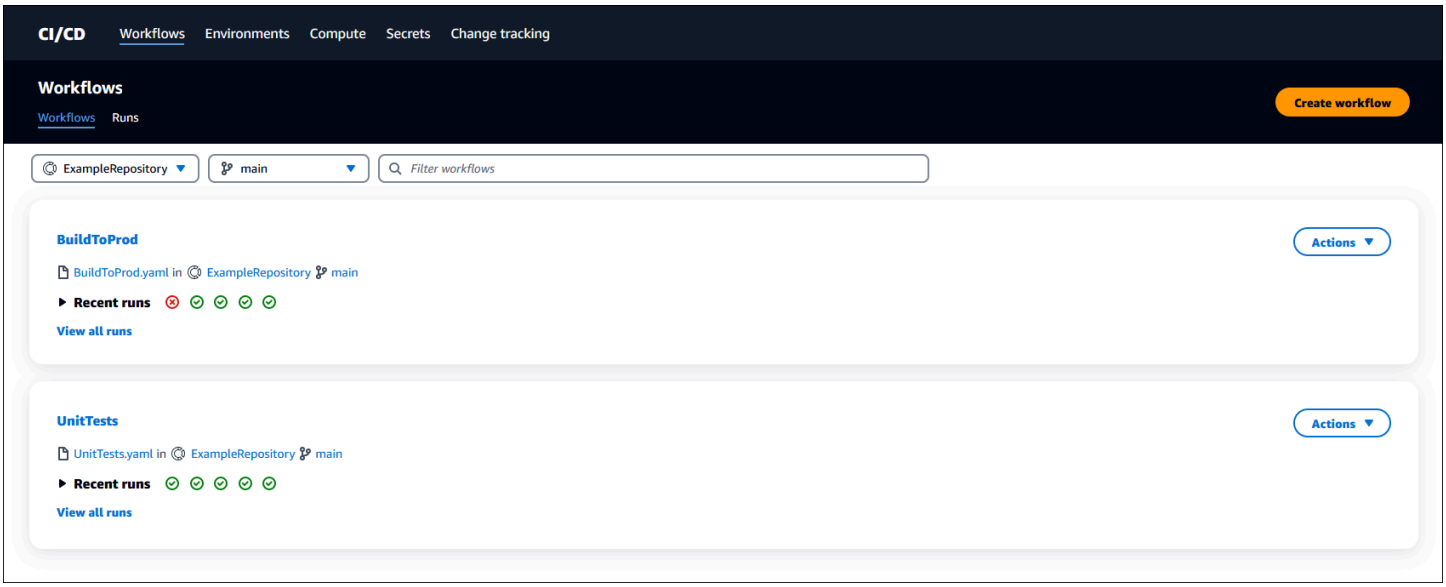


Anda dapat beralih dari editor visual ke editor YAMAL untuk melihat efek konfigurasi Anda pada kode YAMAL yang mendasarinya.

Menemukan alur kerja

Anda dapat melihat alur kerja Anda di halaman ringkasan Alur kerja, bersama dengan alur kerja lain yang telah Anda atur dalam proyek yang sama.

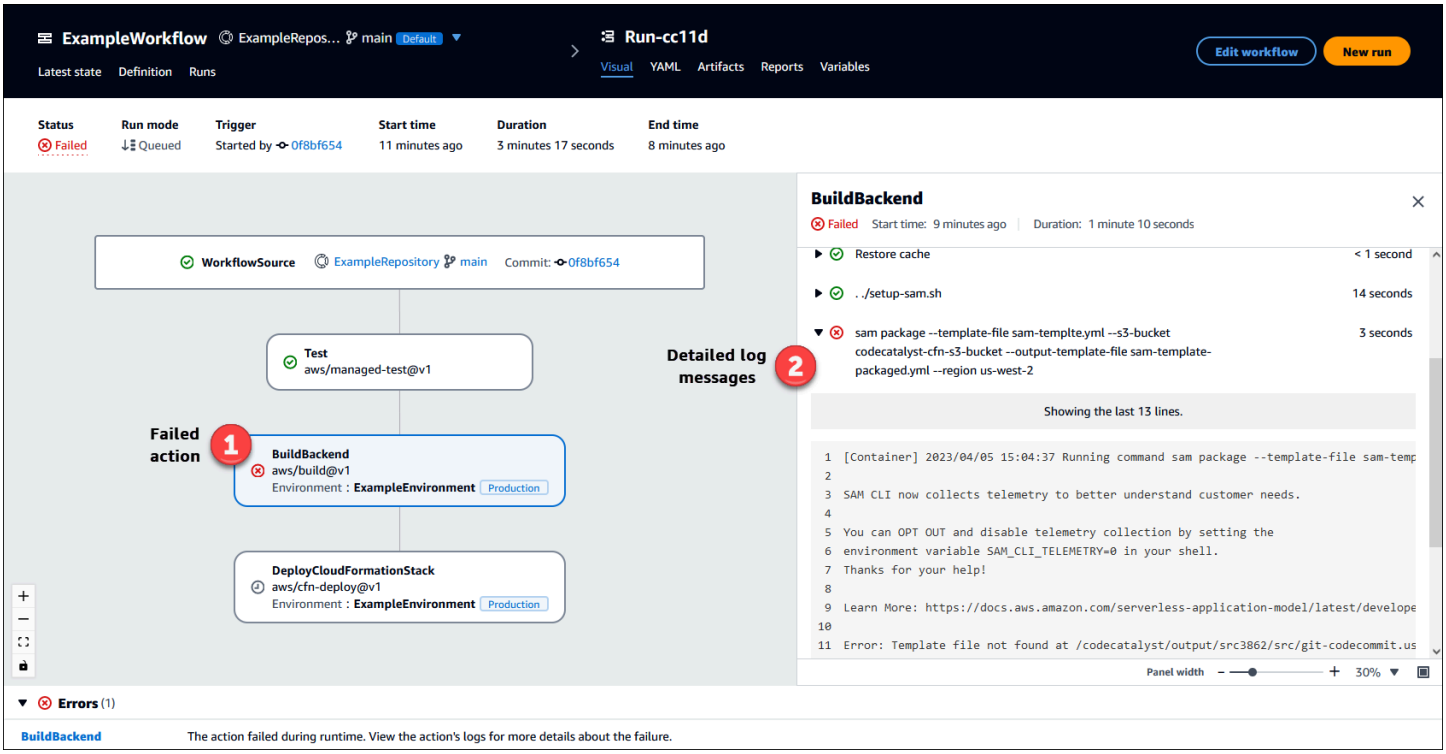
Gambar berikut menunjukkan halaman ringkasan Alur kerja. Itu diisi dengan dua alur kerja: BuildToProddan. UnitTests Anda dapat melihat bahwa keduanya telah dijalankan beberapa kali. Anda dapat memilih Runs terbaru untuk melihat riwayat run dengan cepat, atau memilih nama alur kerja untuk melihat kode YAMAL alur kerja dan informasi terperinci lainnya.



Melihat detail alur kerja

Anda dapat melihat detail alur kerja yang dijalankan dengan memilih run di halaman ringkasan Alur kerja.

Gambar berikut menunjukkan rincian alur kerja yang disebut Run-CC11d yang dimulai secara otomatis pada komit ke sumber. Diagram alur kerja menunjukkan bahwa suatu tindakan telah gagal (1). Anda dapat menavigasi ke log (2) untuk melihat pesan log terperinci dan memecahkan masalah. Untuk informasi selengkapnya tentang alur kerja berjalan, lihatMenjalankan alur kerja.



Langkah selanjutnya

Untuk mempelajari lebih lanjut tentang konsep alur kerja, lihatKonsep alur kerja.

Untuk membuat alur kerja pertama Anda, lihatMemulai dengan alur kerja.

Konsep alur kerja

Berikut adalah beberapa konsep dan istilah yang perlu diketahui saat membangun, menguji, atau menerapkan kode Anda dengan alur kerja. CodeCatalyst

Alur Kerja

Alur kerja adalah prosedur otomatis yang menjelaskan cara membangun, menguji, dan menyebarkan kode Anda sebagai bagian dari sistem integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD). Alur kerja mendefinisikan serangkaian langkah, atau tindakan, yang harus diambil selama menjalankan alur kerja. Alur kerja juga mendefinisikan peristiwa, atau pemicu, yang menyebabkan alur kerja dimulai. Untuk menyiapkan alur kerja, Anda membuat file definisi alur kerja menggunakan editor visual atau YAMAL CodeCatalyst konsol.

> **Tip**
>
> Untuk melihat sekilas bagaimana Anda dapat menggunakan alur kerja dalam sebuah proyek, buat proyek dengan cetak biru. Setiap cetak biru menerapkan alur kerja yang berfungsi yang dapat Anda tinjau, jalankan, dan bereksperimen.

File definisi alur kerja

File definisi alur kerja adalah file YAMAL yang menjelaskan alur kerja Anda. File disimpan dalam `~/.codecatalyst/workflows/` folder di root repositori sumber Anda. File dapat memiliki ekstensi.yml.yaml.

Untuk informasi selengkapnya tentang file definisi alur kerja, lihatAlur kerja definisi YAMAL.

Tindakan

Tindakan adalah blok bangunan utama alur kerja, dan mendefinisikan unit logis kerja, atau tugas, untuk dilakukan selama alur kerja dijalankan. Biasanya, alur kerja mencakup beberapa tindakan yang berjalan secara berurutan atau paralel tergantung pada cara Anda mengonfigurasinya.

Untuk informasi selengkapnya tentang tindakan, lihatMengkonfigurasi tindakan yang dilakukan alur kerja.

Kelompok aksi

Grup aksi berisi satu atau lebih tindakan. Mengelompokkan tindakan ke dalam grup tindakan membantu Anda menjaga alur kerja tetap teratur, dan juga memungkinkan Anda mengonfigurasi dependensi di antara grup yang berbeda.

Untuk informasi selengkapnya tentang grup aksi, lihatMengelompokkan tindakan ke dalam kelompok aksi.

Artifacts

Artefak adalah output dari tindakan alur kerja, dan biasanya terdiri dari folder atau arsip file. Artefak penting karena memungkinkan Anda berbagi file dan informasi antar tindakan.

Untuk informasi lebih lanjut tentang artifact, lihat Berbagi data antar tindakan dalam alur kerja menggunakan artefak.

Hitung

Compute mengacu pada mesin komputasi (CPU, memori, dan sistem operasi) yang dikelola dan dikelola oleh CodeCatalyst untuk menjalankan tindakan alur kerja.

Untuk informasi selengkapnya tentang komputasi, lihatMengkonfigurasi gambar Docker lingkungan komputasi dan runtime untuk alur kerja.

Lingkungan

Lingkungan, jangan dikelirukan dengan Lingkungan Pengembang, adalah tempat kode digunakan. Biasanya berisi instance aplikasi yang sedang berjalan bersama dengan infrastruktur yang terkait. Anda dapat memberi nama lingkungan Anda seperti pengembangan, pengujian, pementasan, atau produksi. Setiap penerapan yang dihasilkan oleh CodeCatalyst ke lingkungan akan muncul di halaman Lingkungan. Untuk mengatur lingkungan, Anda memberinya nama, sepertimy-`production-environment`, dan kemudian mengaitkannya dengan Anda Akun AWS.

Selain menampilkan informasi penyebaran, lingkungan juga berfungsi sebagai mekanisme untuk menetapkan peran AWS IAM untuk tindakan alur kerja.

Untuk informasi selengkapnya tentang lingkungan, lihatMenyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst.

Gerbang

Gate adalah komponen alur kerja yang dapat Anda gunakan untuk mencegah alur kerja berjalan kecuali kondisi tertentu terpenuhi. Contoh gerbang adalah gerbang Persetujuan tempat pengguna harus mengirimkan persetujuan di CodeCatalyst konsol sebelum proses alur kerja diizinkan untuk dilanjutkan.

Anda dapat menambahkan gerbang di antara urutan tindakan dalam alur kerja, atau sebelum tindakan pertama (yang berjalan segera setelah Sumber diunduh). Anda juga dapat menambahkan gerbang setelah tindakan terakhir, jika Anda memiliki kebutuhan untuk melakukannya.

Untuk informasi lebih lanjut tentang gerbang, lihatMengembangkan alur kerja.

Gerbang

Gate adalah komponen alur kerja yang dapat Anda gunakan untuk mencegah alur kerja berjalan kecuali kondisi tertentu terpenuhi. Contoh gerbang adalah gerbang Persetujuan tempat pengguna

harus mengirimkan persetujuan di CodeCatalyst konsol sebelum proses alur kerja diizinkan untuk dilanjutkan.

Anda dapat menambahkan gerbang di antara urutan tindakan dalam alur kerja, atau sebelum tindakan pertama (yang berjalan segera setelah Sumber diunduh). Anda juga dapat menambahkan gerbang setelah tindakan terakhir, jika Anda memiliki kebutuhan untuk melakukannya.

Untuk informasi lebih lanjut tentang gerbang, lihatMengembangkan alur kerja.

Laporan

Laporan berisi rincian tentang pengujian yang terjadi selama menjalankan alur kerja. Anda dapat membuat laporan seperti laporan pengujian, laporan cakupan kode, laporan analisis komposisi perangkat lunak, dan laporan analisis statis. Anda dapat menggunakan laporan untuk membantu memecahkan masalah selama alur kerja. Jika memiliki banyak laporan dari beberapa alur kerja, Anda dapat menggunakan laporan untuk melihat tren dan tingkat kegagalan untuk membantu mengoptimalkan aplikasi dan konfigurasi penerapan.

Untuk informasi selengkapnya tentang laporan, lihatJenis laporan kualitas.

Berjalan

Run adalah iterasi tunggal dari alur kerja. Selama menjalankan, CodeCatalyst melakukan tindakan yang ditentukan dalam file konfigurasi alur kerja dan mengeluarkan log, artefak, dan variabel terkait.

Untuk informasi selengkapnya tentang lari, lihatMenjalankan alur kerja.

Sumber

Sumber, juga disebut sumber input, adalah repositori sumber yang menghubungkan tindakan alur kerja untuk mendapatkan file yang dibutuhkan untuk menjalankan operasinya. Misalnya, tindakan alur kerja mungkin terhubung ke repositori sumber untuk mendapatkan file sumber aplikasi untuk membangun aplikasi.

Untuk informasi selengkapnya tentang sumber, lihat Menghubungkan alur kerja ke repositori sumber.

Variabel

Variabel adalah pasangan kunci-nilai yang berisi informasi yang dapat Anda referensikan dalam alur kerja Anda CodeCatalyst .

Untuk informasi lebih lanjut tentang variabel, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

Pemicu alur kerja

Pemicu alur kerja, atau hanya pemicu, memungkinkan Anda memulai alur kerja yang berjalan secara otomatis ketika peristiwa tertentu terjadi, seperti push kode. Anda mungkin ingin mengonfigurasi pemicu untuk membebaskan pengembang perangkat lunak Anda dari keharusan memulai alur kerja yang berjalan secara manual melalui konsol. CodeCatalyst

Anda dapat menggunakan tiga jenis pemicu:

- Push - Pemicu push kode menyebabkan alur kerja dijalankan setiap kali komit didorong.
- Permintaan tarik - Pemicu permintaan tarik menyebabkan alur kerja dijalankan setiap kali permintaan tarik dibuat, direvisi, atau ditutup.
- Jadwal - Pemicu jadwal menyebabkan alur kerja berjalan dimulai pada jadwal yang Anda tentukan. Pertimbangkan untuk menggunakan pemicu jadwal untuk menjalankan build malam perangkat lunak Anda sehingga build terbaru siap untuk pengembang perangkat lunak Anda untuk bekerja keesokan paginya.

Anda dapat menggunakan push, pull request, dan schedule trigger sendiri atau dalam kombinasi dalam alur kerja yang sama.

Untuk informasi lebih lanjut tentang menggunakan pemicu, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.

Memulai dengan alur kerja

Dalam tutorial ini, Anda akan belajar cara membuat dan mengkonfigurasi alur kerja pertama Anda.

> **Tip**
>
> Lebih suka memulai dengan alur kerja yang telah dikonfigurasi sebelumnya? LihatMembuat proyek dengan cetak biru, yang mencakup instruksi untuk menyiapkan proyek dengan alur kerja yang berfungsi, contoh aplikasi, dan sumber daya lainnya.

Topik

Prasyarat

Sebelum Anda memulai:

- Anda membutuhkan CodeCatalyst ruang. Untuk informasi selengkapnya, lihat Menciptakan ruang.
- Di CodeCatalyst ruang Anda, Anda memerlukan CodeCatalyst proyek kosong, Mulai dari awal yang disebut:

  ```
  codecatalyst-project
  ```

  Untuk informasi selengkapnya, lihat Membuat proyek kosong di Amazon CodeCatalyst.
- Dalam proyek Anda, Anda memerlukan CodeCatalyst repositori yang disebut:

  ```
  codecatalyst-source-repository
  ```

  Untuk informasi selengkapnya, lihat Membuat repositori sumber.

> **Note**
>
> Jika Anda memiliki proyek dan repositori sumber yang ada, Anda dapat menggunakannya; Namun, membuat yang baru membuat pembersihan lebih mudah di akhir tutorial ini.

Langkah 1: Buat dan konfigurasikan alur kerja Anda

Pada langkah ini, Anda membuat dan mengonfigurasi alur kerja yang secara otomatis membangun dan menguji kode sumber Anda saat perubahan dilakukan.

Untuk membuat alur kerja Anda

1. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih Buat alur kerja.

   File definisi alur kerja muncul di editor YAMAL CodeCatalyst konsol.

Untuk mengonfigurasi alur kerja Anda

Anda dapat mengonfigurasi alur kerja Anda di editor Visual, atau editor YAMAL. Mari kita mulai dengan editor YAMAL dan kemudian beralih ke editor visual.

1. Pilih + Tindakan untuk melihat daftar tindakan alur kerja yang dapat Anda tambahkan ke alur kerja Anda.
2. Dalam tindakan Build, pilih + untuk menambahkan YAMAL tindakan ke file definisi alur kerja Anda. Alur kerja Anda sekarang terlihat mirip dengan yang berikut ini.

```
Name: Workflow_fe47
SchemaVersion: "1.0"

# Optional - Set automatic triggers.
Triggers:
  - Type: Push
    Branches:
      - main

# Required - Define action configurations.
Actions:
  Build_f0:
    Identifier: aws/build@v1

    Inputs:
      Sources:
        - WorkflowSource # This specifies that the action requires this workflow as
 a source

    Outputs:
      AutoDiscoverReports:
        Enabled: true
        # Use as prefix for the report files
        ReportNamePrefix: rpt
```

```
   Configuration:
     Steps:
       - Run: echo "Hello, World!"
       - Run: echo "<?xml version=\"1.0\" encoding=\"UTF-8\" ?>" >> report.xml
       - Run: echo "<testsuite tests=\"1\" name=\"TestAgentJunit\" >" >>
report.xml
       - Run: echo "<testcase classname=\"TestAgentJunit\" name=\"Dummy
           Test\"/></testsuite>" >> report.xml
```

Alur kerja menyalin file dalam repositori **`WorkflowSource`** sumber ke mesin komputasi yang menjalankan **`Build_f0`** tindakan, mencetak **`Hello, World!`** ke log, menemukan laporan pengujian di mesin komputasi, dan mengeluarkannya ke halaman Laporan konsol. CodeCatalyst

3. Pilih Visual untuk melihat file definisi alur kerja di editor visual. Bidang di editor visual memungkinkan Anda mengonfigurasi properti YAMAL yang ditampilkan di editor YAMAL.

Langkah 2: Simpan alur kerja Anda dengan komit

Pada langkah ini, Anda menyimpan perubahan Anda. Karena alur kerja disimpan sebagai `.yaml` file di repositori, Anda menyimpan perubahan dengan komit.

Untuk melakukan perubahan alur kerja Anda

1. (Opsional) Pilih Validasi untuk memastikan kode YAMAL alur kerja valid.
2. Pilih Terapkan.
3. Dalam nama file Workflow, masukkan nama untuk file konfigurasi alur kerja Anda, seperti. **`my-first-workflow`**
4. Dalam pesan Komit, masukkan pesan untuk mengidentifikasi komit Anda, seperti**`create my-first-workflow.yaml`**.
5. Di Repositori, pilih repositori yang ingin Anda simpan alur kerja di (). `codecatalyst-repository`
6. Di Nama cabang, pilih cabang yang ingin Anda simpan alur kerja di (`main`).
7. Pilih Terapkan.

Alur kerja baru Anda muncul dalam daftar alur kerja. Mungkin perlu beberapa saat untuk muncul.

Karena alur kerja disimpan dengan komit, dan karena alur kerja memiliki pemicu push kode yang dikonfigurasi, menyimpan alur kerja memulai alur kerja berjalan secara otomatis.

Langkah 3: Lihat hasil run

Pada langkah ini, Anda menavigasi ke proses yang dimulai dari komit Anda dan melihat hasilnya.

Untuk melihat hasil yang dijalankan

1. Pilih nama alur kerja Anda, misalnya,`Workflow_fe47`.

   Diagram alur kerja yang menunjukkan label repositori sumber Anda (WorkflowSource) dan tindakan build (misalnya, build_F0).
2. Dalam diagram alur kerja, pilih tindakan build (misalnya, Build_f0).
3. Tinjau isi tab Log, Laporan, Konfigurasi, dan Variabel. Tab ini menunjukkan hasil tindakan build Anda.

   Untuk informasi selengkapnya, lihat Melihat hasil tindakan build.

(Opsional) Langkah 4: Bersihkan

Pada langkah ini, Anda membersihkan sumber daya yang Anda buat dalam tutorial ini.

Untuk menghapus sumber daya

- Jika Anda membuat proyek baru untuk tutorial ini, hapuslah. Untuk petunjuk, lihat Menghapus proyek. Menghapus proyek juga menghapus repositori sumber dan alur kerja.

Membangun dengan alur kerja

Menggunakan CodeCatalyst alur kerja, Anda dapat membangun aplikasi dan sumber daya lainnya.

Topik

- Bagaimana cara membangun aplikasi?
- Manfaat dari tindakan membangun
- Alternatif untuk aksi build
- Menambahkan aksi build

- Melihat hasil tindakan build
- Tutorial: Unggah artefak ke Amazon S3
- Membangun dan menguji definisi YAMAL tindakan

Bagaimana cara membangun aplikasi?

Untuk membuat aplikasi atau sumber daya CodeCatalyst, pertama-tama Anda membuat alur kerja, lalu menentukan tindakan build di dalamnya.

Tindakan build adalah blok pembangun alur kerja yang mengkompilasi kode sumber Anda, menjalankan pengujian unit, dan menghasilkan artefak yang siap digunakan.

Anda menambahkan tindakan build ke alur kerja menggunakan editor visual CodeCatalyst konsol atau editor YAMAL.

Langkah-langkah tingkat tinggi untuk membangun aplikasi atau sumber daya adalah sebagai berikut.

Untuk membangun aplikasi (tugas tingkat tinggi)

1. Di CodeCatalyst, Anda menambahkan kode sumber untuk aplikasi yang ingin Anda bangun. Untuk informasi selengkapnya, lihat Menyimpan kode sumber di repositori untuk proyek di CodeCatalyst.
2. Di CodeCatalyst, Anda membuat alur kerja. Alur kerja adalah tempat Anda menentukan cara membangun, menguji, dan menerapkan aplikasi Anda. Untuk informasi selengkapnya, lihat Memulai dengan alur kerja.
3. (Opsional) Dalam alur kerja, Anda menambahkan pemicu yang menunjukkan peristiwa yang akan menyebabkan alur kerja dimulai secara otomatis. Untuk informasi selengkapnya, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu
4. Dalam alur kerja, Anda menambahkan tindakan build yang mengkompilasi dan mengemas kode sumber aplikasi atau sumber daya Anda. Secara opsional, Anda juga dapat meminta tindakan build menjalankan pengujian unit, menghasilkan laporan, dan menerapkan aplikasi jika Anda tidak ingin menggunakan tindakan pengujian atau penerapan untuk tujuan ini. Untuk informasi lebih lanjut tentang tindakan pengujian dan penerapan, lihatMenambahkan aksi build.
5. (Opsional) Dalam alur kerja, Anda menambahkan tindakan pengujian dan tindakan penerapan untuk menguji dan menerapkan aplikasi atau sumber daya Anda. Anda dapat memilih dari beberapa tindakan yang telah dikonfigurasi sebelumnya untuk menerapkan aplikasi Anda ke

target yang berbeda, seperti Amazon ECS. Untuk informasi selengkapnya, lihat Pengujian dengan alur kerja, dan Menerapkan dengan alur kerja.

6. Anda memulai alur kerja baik secara manual atau otomatis melalui pemicu. Alur kerja menjalankan tindakan build, test, dan deploy secara berurutan untuk membangun, menguji, dan menerapkan aplikasi dan resource Anda ke target. Untuk informasi selengkapnya, lihat Memulai alur kerja berjalan secara manual.

Manfaat dari tindakan membangun

Menggunakan tindakan build dalam alur kerja memiliki manfaat sebagai berikut:

- Dikelola sepenuhnya — Tindakan build menghilangkan kebutuhan untuk menyiapkan, menambal, memperbarui, dan mengelola server build Anda sendiri.
- Sesuai permintaan — Tindakan build menskalakan sesuai permintaan untuk memenuhi kebutuhan build Anda. Anda hanya membayar untuk jumlah menit build yang Anda konsumsi. Untuk informasi selengkapnya, lihat Mengkonfigurasi gambar Docker lingkungan komputasi dan runtime untuk alur kerja.
- Di luar kotak — CodeCatalyst termasuk gambar Docker lingkungan runtime yang dikemas sebelumnya yang digunakan untuk menjalankan semua tindakan alur kerja Anda, termasuk tindakan build. Gambar-gambar ini telah dikonfigurasi sebelumnya dengan alat yang berguna untuk membangun aplikasi seperti AWS CLI dan Node.js. Anda dapat mengonfigurasi CodeCatalyst untuk menggunakan image build yang Anda berikan dari registri publik atau pribadi. Untuk informasi selengkapnya, lihat Menentukan gambar Docker lingkungan runtime.

Alternatif untuk aksi build

Jika Anda menggunakan tindakan build untuk menerapkan aplikasi Anda, pertimbangkan untuk menggunakan tindakan CodeCatalyst penerapan sebagai gantinya. Tindakan penerapan melakukan behind-the-scenes konfigurasi yang seharusnya Anda tulis secara manual jika Anda menggunakan tindakan build. Untuk informasi selengkapnya tentang tindakan penerapan yang tersedia, lihatDaftar tindakan penerapan.

Anda juga dapat menggunakan AWS CodeBuild untuk membangun aplikasi Anda. Untuk informasi lebih lanjut, lihat Apa itu CodeBuild? .

Menambahkan aksi build

Gunakan prosedur berikut untuk menambahkan tindakan build ke CodeCatalyst alur kerja Anda.

Visual

Untuk menambahkan tindakan build menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
3. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
4. Pilih Edit.
5. Pilih Visual.
6. Pilih Tindakan.
7. Di Actions, pilih Build.
8. Pada tab Input dan Konfigurasi, lengkapi bidang sesuai dengan kebutuhan Anda. Untuk deskripsi setiap bidang, lihatMembangun dan menguji definisi YAMAL tindakan. Referensi ini memberikan informasi rinci tentang setiap bidang (dan nilai properti YAMAL yang sesuai) seperti yang muncul di YAMAL dan editor visual.
9. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
10. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menambahkan tindakan build menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
3. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
4. Pilih Edit.
5. Pilih YAMAL.
6. Pilih Tindakan.
7. Di Actions, pilih Build.

8. Ubah properti dalam kode YAMAL sesuai dengan kebutuhan Anda. Penjelasan tentang setiap properti yang tersedia disediakan diMembangun dan menguji definisi YAMAL tindakan.
9. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
10. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Membangun definisi tindakan

Tindakan build didefinisikan sebagai sekumpulan properti YAMAL di dalam file definisi alur kerja Anda. Untuk informasi tentang properti ini, lihat Membangun dan menguji definisi YAMAL tindakan diAlur kerja definisi YAMAL.

Melihat hasil tindakan build

Gunakan petunjuk berikut untuk melihat hasil tindakan build, termasuk log, laporan, dan variabel yang dihasilkan.

Untuk melihat hasil tindakan build

1. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
3. Dalam diagram alur kerja, pilih nama tindakan build Anda, misalnya, Build.
4. Untuk melihat log untuk build run, pilih Log. Log untuk berbagai fase build ditampilkan. Anda dapat memperluas atau menciutkan log sesuai kebutuhan.
5. Untuk melihat laporan pengujian yang dihasilkan oleh tindakan build, pilih Laporan, atau di panel navigasi, pilih Laporan. Untuk informasi selengkapnya, lihat Jenis laporan kualitas.
6. Untuk melihat konfigurasi yang digunakan untuk tindakan build, pilih Configuration. Untuk informasi selengkapnya, lihat Menambahkan aksi build.
7. Untuk melihat variabel yang digunakan oleh aksi build, pilih Variables. Untuk informasi selengkapnya, lihat Mengkonfigurasi dan menggunakan variabel dalam alur kerja.

Tutorial: Unggah artefak ke Amazon S3

Dalam tutorial ini, Anda mempelajari cara mengunggah artefak ke bucket Amazon S3 menggunakan alur kerja CodeCatalyst yang menyertakan beberapa tindakan build. Tindakan ini berjalan

secara seri saat alur kerja dimulai. Tindakan build pertama menghasilkan dua file, `Hello.txt` dan`Goodbye.txt`, dan menggabungkannya menjadi artefak build. Tindakan build kedua mengunggah artefak ke Amazon S3. Anda akan mengonfigurasi alur kerja untuk dijalankan setiap kali Anda mendorong komit ke repositori sumber Anda.

Topik

Prasyarat

Sebelum memulai, Anda perlu melakukan hal berikut:

- Anda membutuhkan CodeCatalyst ruang dengan AWS akun yang terhubung. Untuk informasi selengkapnya, lihat Menciptakan ruang.
- Di ruang Anda, Anda memerlukan CodeCatalyst proyek kosong, Mulai dari awal yang disebut:

  ```
  codecatalyst-artifact-project
  ```

  Untuk informasi selengkapnya, lihat Membuat proyek kosong di Amazon CodeCatalyst.
- Dalam proyek Anda, Anda memerlukan CodeCatalyst lingkungan yang disebut:

  ```
  codecatalyst-artifact-environment
  ```

  Konfigurasikan lingkungan ini sebagai berikut:

  - Pilih jenis apa saja, seperti Pengembangan.
  - Hubungkan AWS akun Anda ke sana.

  Untuk informasi selengkapnya, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst.

Langkah 1: Buat AWS peran

Pada langkah ini, Anda membuat peran AWS IAM yang nantinya akan Anda tetapkan ke tindakan build dalam alur kerja Anda. Peran ini memberikan izin tindakan CodeCatalyst build untuk mengakses AWS akun Anda dan menulis ke Amazon S3 tempat artefak Anda akan disimpan. Peran tersebut disebut peran Build.

> **Note**
>
> Jika Anda sudah memiliki peran build yang Anda buat untuk tutorial lain, Anda dapat menggunakannya untuk tutorial ini juga. Pastikan itu memiliki izin dan kebijakan kepercayaan yang ditunjukkan dalam prosedur berikut.

Untuk informasi selengkapnya tentang peran IAM, lihat peran IAM di AWS AWS Identity and Access Management Panduan Pengguna.

Untuk membuat peran build

1. Buat kebijakan untuk peran tersebut, sebagai berikut:

    a. Masuk ke AWS.
    b. Buka konsol IAM di https://console.aws.amazon.com/iam/.
    c. Di panel navigasi, pilih Kebijakan.
    d. Pilih Buat kebijakan.
    e. Pilih tab JSON.
    f. Hapus kode yang ada.
    g. Tempel kode berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:ListBucket"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

h. Pilih Berikutnya: Tanda.

i. Pilih Berikutnya: Tinjau.

j. Dalam Nama, masukkan:

```
codecatalyst-s3-build-policy
```

k. Pilih Buat kebijakan.

Anda sekarang telah membuat kebijakan izin.

2. Buat peran build, sebagai berikut:

a. Di panel navigasi, pilih Peran, lalu pilih Buat peran.

b. Pilih Kebijakan kepercayaan khusus.

c. Hapus kebijakan kepercayaan kustom yang ada.

d. Tambahkan kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
```

```
                    "codecatalyst.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

e. Pilih Selanjutnya.

f. Di Kebijakan izin, cari `codecatalyst-s3-build-policy` dan pilih kotak centang.

g. Pilih Selanjutnya.

h. Untuk nama Peran, masukkan:

```
codecatalyst-s3-build-role
```

i. Untuk deskripsi Peran, masukkan:

```
CodeCatalyst build role
```

j. Pilih Buat peran.

Anda sekarang telah membuat peran build dengan kebijakan kepercayaan dan kebijakan izin.

Langkah 2: Buat ember Amazon S3

Pada langkah ini, Anda membuat ember Amazon S3 tempat `Goodbye.txt` artefak `Hello.txt` dan artefak akan diunggah.

Untuk membuat bucket Amazon S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel utama, pilih Buat ember.
3. Untuk nama Bucket, masukkan:

```
codecatalyst-artifact-bucket
```

4. Untuk Wilayah AWS , pilih Wilayah. Tutorial ini mengasumsikan Anda memilih US West (Oregon) us-west-2. Untuk informasi tentang Wilayah yang didukung oleh Amazon S3, lihat titik akhir dan kuota Amazon Simple Storage Service di. Referensi Umum AWS

5. Di bagian bawah halaman, pilih Buat ember.
6. Salin nama bucket yang baru saja Anda buat, misalnya:

```
codecatalyst-artifact-bucket
```

Anda sekarang telah membuat ember yang disebut **`codecatalyst-artifact-bucket`** di Wilayah AS Barat (Oregon) us-west-2.

Langkah 3: Buat repositori sumber

Pada langkah ini, Anda membuat repositori sumber di. CodeCatalyst Repositori ini digunakan untuk menyimpan file definisi alur kerja tutorial.

Untuk informasi lebih lanjut tentang repositori sumber, lihat. Membuat repositori sumber

Untuk membuat repositori sumber

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda,`codecatalyst-artifact-project`.
3. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
4. Pilih Tambahkan repositori, lalu pilih Buat repositori.
5. Dalam nama Repositori, masukkan:

```
codecatalyst-artifact-source-repository
```

6. Pilih Buat.

Anda sekarang telah membuat repositori yang disebut. `codecatalyst-artifact-source-repository`

Langkah 4: Buat alur kerja

Pada langkah ini, Anda membuat alur kerja yang terdiri dari blok bangunan berikut yang berjalan secara berurutan:

- Pemicu - Pemicu ini memulai alur kerja yang dijalankan secara otomatis saat Anda mendorong perubahan ke repositori sumber Anda. Untuk informasi lebih lanjut tentang pemicu, lihatMemulai alur kerja berjalan secara otomatis dengan pemicu.

- Tindakan build yang disebut `GenerateFiles` — Pada pemicu, `GenerateFiles` aksi membuat dua file, `Hello.txt` dan`Goodbye.txt`, dan mengemasnya ke dalam artefak keluaran yang disebut`codecatalystArtifact`.
- Tindakan build lain yang disebut `Upload` — Setelah menyelesaikan `GenerateFiles` tindakan, `Upload` tindakan menjalankan AWS CLI perintah `aws s3 sync` untuk mengunggah file di `codecatalystArtifact` dan di repositori sumber Anda ke bucket Amazon S3 Anda. Ini sudah AWS CLI diinstal sebelumnya dan dikonfigurasi sebelumnya pada platform CodeCatalyst komputasi, jadi Anda tidak perlu menginstal atau mengkonfigurasinya.

  Untuk informasi selengkapnya tentang perangkat lunak pra-paket pada platform CodeCatalyst komputasi, lihat. Menentukan gambar Docker lingkungan runtime Untuk informasi selengkapnya tentang `aws s3 sync` perintah, lihat sinkronisasi di Referensi AWS CLI Perintah. AWS CLI

Untuk informasi selengkapnya tentang tindakan build, lihatMembangun dengan alur kerja.

Untuk membuat alur kerja

1. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih Buat alur kerja.
3. Hapus kode sampel YAMAL.
4. Tambahkan kode YAMAL berikut:

```
Name: codecatalyst-artifact-workflow
SchemaVersion: 1.0

Triggers:
  - Type: Push
    Branches:
      - main
Actions:
  GenerateFiles:
    Identifier: aws/build@v1
    Configuration:
      Steps:
        # Create the output files.
        - Run: echo "Hello, World!" > "Hello.txt"
        - Run: echo "Goodbye!" > "Goodbye.txt"
    Outputs:
      Artifacts:
```

```
        - Name: codecatalystArtifact
          Files:
            - "**/*"
  Upload:
    Identifier: aws/build@v1
    DependsOn:
      - GenerateFiles
    Environment:
      Name: codecatalyst-artifact-environment
      Connections:
        - Name: codecatalyst-account-connection
          Role: codecatalyst-s3-build-role
    Inputs:
      Artifacts:
        - codecatalystArtifact
    Configuration:
      Steps:
        # Upload the output artifact to the S3 bucket.
        - Run: aws s3 sync . s3://codecatalyst-artifact-bucket
```

Pada kode di atas, ganti:

- *codecatalyst-artifact-environment*dengan nama lingkungan yang Anda buatPrasyarat.
- *codecatalyst-account-connection*dengan nama koneksi akun yang Anda buatPrasyarat.
- *codecatalyst-s3-build-role dengan nama peran* build yang Anda buat. Langkah 1: Buat AWS peran
- *codecatalyst-artifact-bucket*dengan nama Amazon S3 yang Anda buat. Langkah 2: Buat ember Amazon S3

Untuk informasi tentang properti dalam file ini, lihatMembangun dan menguji definisi YAMAL tindakan.

5. (Opsional) Pilih Validasi untuk memastikan kode YAMAL valid sebelum melakukan.
6. Pilih Terapkan.
7. Pada kotak dialog Commit workflow, masukkan yang berikut ini:

   a. Untuk nama file Workflow, biarkan default,`codecatalyst-artifact-workflow`.

b. Untuk pesan Commit, masukkan:

```
add initial workflow file
```

c. Untuk Repositori, pilih. codecatalyst-artifact-source-repository

d. Untuk nama cabang, pilih main.

e. Pilih Terapkan.

Anda sekarang telah membuat alur kerja. Jalankan alur kerja dimulai secara otomatis karena pemicu yang ditentukan di bagian atas alur kerja. Khususnya, ketika Anda melakukan (dan mendorong) `codecatalyst-artifact-workflow.yaml` file ke repositori sumber Anda, pemicu memulai alur kerja dijalankan.

Untuk melihat alur kerja yang sedang berjalan

1. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih alur kerja yang baru saja Anda buat:`codecatalyst-artifact-workflow`.
3. Pilih GenerateFilesuntuk melihat progres tindakan build pertama.
4. Pilih Unggah untuk melihat progres tindakan build kedua.
5. Setelah tindakan Upload selesai, lakukan hal berikut:
   - Jika alur kerja berjalan berhasil, pergi ke prosedur berikutnya.
   - Jika alur kerja gagal, pilih Log untuk memecahkan masalah.

Langkah 5: Verifikasi hasilnya

Setelah alur kerja berjalan, buka layanan Amazon S3 dan lihat di *`codecatalyst-artifact-bucket`*bucket Anda. Sekarang harus menyertakan file dan folder berikut:

```
.
|— .aws/
|— .git/
|Goodbye.txt
|Hello.txt
|REAME.md
```

`Hello.txt`File `Goodbye.txt` dan diunggah karena mereka adalah bagian dari `codecatalystArtifact` artefak. `README.md`File`.aws/`,`.git/`, dan diunggah karena berada di repositori sumber Anda.

Bersihkan

Bersihkan CodeCatalyst dan AWS untuk menghindari biaya untuk layanan ini.

Untuk membersihkan di CodeCatalyst

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Hapus repositori `codecatalyst-artifact-source-repository` sumber.
3. Hapus `codecatalyst-artifact-workflow` alur kerja.

Untuk membersihkan di AWS

1. Bersihkan di Amazon S3, sebagai berikut:
   a. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
   b. Hapus file di `codecatalyst-artifact-bucket` ember.
   c. Hapus `codecatalyst-artifact-bucket` ember.
2. Bersihkan di IAM, sebagai berikut:
   a. Buka konsol IAM di https://console.aws.amazon.com/iam/.
   b. Hapus`codecatalyst-s3-build-policy`.
   c. Hapus`codecatalyst-s3-build-role`.

Membangun dan menguji definisi YAMAL tindakan

Berikut ini adalah definisi YAMAL dari tindakan build dan test. Ada satu referensi untuk dua tindakan karena properti YAML mereka sangat mirip.

Definisi tindakan ini ada sebagai bagian dalam file definisi alur kerja yang lebih luas. Untuk informasi selengkapnya tentang file ini, lihatAlur kerja definisi YAMAL.

Pilih properti YAMB dalam kode berikut untuk melihat deskripsi jika itu.

> **Note**
>
> Sebagian besar properti YAMAL yang mengikuti memiliki elemen UI yang sesuai di editor visual. Untuk mencari elemen UI, gunakan Ctrl+F. Elemen akan terdaftar dengan properti YAMLnya yang terkait.

```
# The workflow definition starts here.
# See Properti tingkat atas for details.

Name: MyWorkflow
SchemaVersion: 1.0
Actions:

# The action definition starts here.
  action-name:
    Identifier: aws/build@v1 | aws/managed-test@v1
    DependsOn:
      - dependent-action-name-1
    Compute:
      Type: EC2 | Lambda
      Fleet: fleet-name
    Timeout: timeout-minutes
    Environment:
      Name: environment-name
      Connections:
        - Name: account-connection-name
          Role: iam-role-name
    Caching:
      FileCaching:
        key-name-1:
          Path: file1.txt
          RestoreKeys:
            - restore-key-1
    Inputs:
      Sources:
        - source-name-1
        - source-name-2
      Artifacts:
        - artifact-name
      Variables:
        - Name: variable-name-1
```

```
          Value: variable-value-1
        - Name: variable-name-2
          Value: variable-value-2
    Outputs:
      Artifacts:
        - Name: output-artifact-1
          Files:
            - build-output/artifact-1.jar
            - "build-output/build*"
        - Name: output-artifact-2
          Files:
            - build-output/artifact-2.1.jar
            - build-output/artifact-2.2.jar
      Variables:
        - variable-name-1
        - variable-name-2
      AutoDiscoverReports:
        Enabled: true | false
        ReportNamePrefix: AutoDiscovered
        IncludePaths:
          - "**/*"
        ExcludePaths:
          - node_modules/cdk/junit.xml
        SuccessCriteria:
          PassRate: percent
          LineCoverage: percent
          BranchCoverage: percent
          Vulnerabilities:
            Severity: CRITICAL | HIGH | MEDIUM | LOW | INFORMATIONAL
            Number: whole-number
          StaticAnalysisBug:
            Severity: CRITICAL | HIGH | MEDIUM | LOW | INFORMATIONAL
            Number: whole-number
          StaticAnalysisSecurity:
            Severity: CRITICAL | HIGH | MEDIUM | LOW | INFORMATIONAL
            Number: whole-number
          StaticAnalysisQuality:
            Severity: CRITICAL | HIGH | MEDIUM | LOW | INFORMATIONAL
            Number: whole-number
          StaticAnalysisFinding:
            Severity: CRITICAL | HIGH | MEDIUM | LOW | INFORMATIONAL
            Number: whole-number
      Reports:
        report-name-1:
```

```
        Format: format
        IncludePaths:
          - "*.xml"
        ExcludePaths:
          - report2.xml
          - report3.xml
        SuccessCriteria:
          PassRate: percent
          LineCoverage: percent
          BranchCoverage: percent
          Vulnerabilities:
            Severity: CRITICAL | HIGH | MEDIUM | LOW | INFORMATIONAL
            Number: whole-number
          StaticAnalysisBug:
              Severity: CRITICAL | HIGH | MEDIUM | LOW | INFORMATIONAL
              Number: whole-number
          StaticAnalysisSecurity:
              Severity: CRITICAL | HIGH | MEDIUM | LOW | INFORMATIONAL
              Number: whole-number
          StaticAnalysisQuality:
              Severity: CRITICAL | HIGH | MEDIUM | LOW | INFORMATIONAL
              Number: whole-number
          StaticAnalysisFinding:
              Severity: CRITICAL | HIGH | MEDIUM | LOW | INFORMATIONAL
              Number: whole-number
    Configuration:
      Container:
        Registry: registry
        Image: image
      Steps:
        - Run: "step 1"
        - Run: "step 2"
      Packages:
        NpmConfiguration:
          PackageRegistries:
            - PackagesRepository: package-repository
              Scopes:
                - "@scope"
```

nama-tindakan

(Diperlukan)

Tentukan nama tindakan. Semua nama tindakan harus unik dalam alur kerja. Nama aksi terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama tindakan.

UI yang sesuai: Tab konfigurasi/Nama tindakan

Identifier

(*nama-tindakan/*) Identifier

Mengidentifikasi tindakan. Jangan mengubah properti ini kecuali Anda ingin mengubah versi. Untuk informasi selengkapnya, lihat Menentukan versi mayor, minor, atau patch dari suatu tindakan.

Gunakan `aws/build@v1` untuk membangun tindakan.

Gunakan `aws/managed-test@v1` untuk tindakan pengujian.

UI yang sesuai: Diagram alur kerja/nama-tindakan/aws/build @v1 |aws/managed-test @v1 label

DependsOn

(*nama-tindakan/*) DependsOn

(Opsional)

Tentukan tindakan, grup tindakan, atau gerbang yang harus berjalan dengan sukses agar tindakan ini berjalan.

Untuk informasi selengkapnya tentang fungsionalitas 'tergantung pada', lihat. Mengkonfigurasi tindakan untuk bergantung pada tindakan lain

UI yang sesuai: Tab masukan/Tergantung pada - opsional

Compute

(*nama-tindakan/*) Compute

(Opsional)

Mesin komputasi yang digunakan untuk menjalankan tindakan alur kerja Anda. Anda dapat menentukan komputasi baik di tingkat alur kerja atau di tingkat tindakan, tetapi tidak keduanya. Ketika ditentukan pada tingkat alur kerja, konfigurasi komputasi berlaku untuk semua tindakan

yang ditentukan dalam alur kerja. Pada tingkat alur kerja, Anda juga dapat menjalankan beberapa tindakan pada instance yang sama. Untuk informasi selengkapnya, lihat Berbagi komputasi di seluruh tindakan.

UI yang sesuai: tidak ada

Type

(***`Tipe nama-tindakan`***/Compute/)

(Diperlukan Compute jika disertakan)

Jenis mesin komputasi. Anda dapat menggunakan salah satu nilai berikut:

- `EC2` (editor visual) atau `EC2` (editor YAMAL)

  Dioptimalkan untuk fleksibilitas selama aksi berjalan.
- `Lambda` (editor visual) atau `Lambda` (editor YAMAL)

  Kecepatan start-up aksi yang dioptimalkan.

Untuk informasi selengkapnya tentang jenis komputasi, lihatJenis komputasi.

UI yang sesuai: Tab konfigurasi/Jenis komputasi

Fleet

(*`nama-tindakan) /Compute/`* ***`Fleet`***

(Opsional)

Tentukan mesin atau armada yang akan menjalankan alur kerja atau tindakan alur kerja Anda. Dengan armada sesuai permintaan, ketika suatu tindakan dimulai, alur kerja menyediakan sumber daya yang dibutuhkan, dan mesin dihancurkan ketika tindakan selesai. Contoh armada sesuai permintaan:`Linux.x86-64.Large`,. `Linux.x86-64.XLarge` Untuk informasi lebih lanjut tentang armada sesuai permintaan, lihat. Properti armada sesuai permintaan

Dengan armada yang disediakan, Anda mengonfigurasi satu set mesin khusus untuk menjalankan tindakan alur kerja Anda. Mesin-mesin ini tetap menganggur, siap untuk memproses tindakan segera. Untuk informasi selengkapnya tentang armada yang disediakan, lihat. Properti armada yang disediakan

Jika `Fleet` dihilangkan, defaultnya adalah. `Linux.x86-64.Large`

UI yang sesuai: Tab konfigurasi/Armada komputasi

Timeout

(*nama-tindakan/*) Timeout

(Opsional)

Tentukan jumlah waktu dalam menit (editor YAMM), atau jam dan menit (editor visual), bahwa tindakan dapat berjalan sebelum CodeCatalyst mengakhiri tindakan. Minimal adalah 5 menit dan maksimum dijelaskan dalamKuota untuk alur kerja. Batas waktu default sama dengan batas waktu maksimum.

UI yang sesuai: Tab konfigurasi/Timeout - opsional

Environment

(*nama-tindakan/*) Environment

(Opsional)

Tentukan CodeCatalyst lingkungan yang akan digunakan dengan tindakan.

Untuk informasi lebih lanjut tentang lingkungan, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst danPembuatan lingkungan.

UI yang sesuai: Tab konfigurasi/Lingkungan

Name

(*nama-tindakan) /Environment/* ***Name***

(Opsional)

Tentukan nama lingkungan yang ada yang ingin Anda kaitkan dengan tindakan.

UI yang sesuai: Tab konfigurasi/Nama Lingkungan

Connections

(*nama-tindakan) /Environment/* ***Connections***

(Opsional)

Tentukan koneksi akun untuk dikaitkan dengan tindakan. Anda dapat menentukan maksimum satu koneksi akun di bawah`Environment`.

Untuk informasi selengkapnya tentang koneksi akun, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS. Untuk informasi tentang cara mengaitkan koneksi akun dengan lingkungan Anda, lihatPembuatan lingkungan.

UI yang sesuai: Tab konfigurasi/

Name

(*nama-tindakan*) */Environment/Connections/* ***Name***

(Opsional)

Tentukan nama koneksi akun.

UI yang sesuai: Tab konfigurasi/

Role

(*nama-tindakan*) */Environment/Connections/* ***Role***

(Opsional)

Tentukan nama peran IAM yang digunakan tindakan ini untuk mengakses dan beroperasi di AWS layanan seperti Amazon S3 dan Amazon ECR. Pastikan peran ini ditambahkan ke koneksi akun Anda. Untuk menambahkan peran IAM ke koneksi akun, lihatMenambahkan peran IAM ke koneksi akun.

> **Note**
>
> Anda mungkin dapat menentukan nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran di sini, asalkan memiliki izin yang cukup. Untuk informasi selengkapnya tentang peran ini, silakan lihat Membuat CodeCatalystWorkflowDevelopmentRole-***`spaceName`***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami menyarankan

Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian.

⚠ Warning

Batasi izin ke yang diperlukan oleh tindakan build dan test. Menggunakan peran dengan izin yang lebih luas dapat menimbulkan risiko keamanan.

UI yang sesuai: Tab konfigurasi/

Caching

(*nama-tindakan/*) Caching

(Opsional)

Bagian di mana Anda dapat menentukan cache untuk menyimpan file pada disk dan mengembalikannya dari cache itu dalam alur kerja berikutnya.

Untuk informasi selengkapnya tentang cache file, lihatCaching file di antara alur kerja berjalan.

UI yang sesuai: Tab konfigurasi/File caching - opsional

FileCaching

(*nama-tindakan) /Caching/* ***FileCaching***

(Opsional)

Bagian yang menentukan konfigurasi untuk urutan cache.

UI yang sesuai: Tab konfigurasi/File caching - opsional/Tambahkan cache

nama-kunci-1

(nama-aksi-nama-kunci-1/Caching/FileCaching/)

(Opsional)

Tentukan nama nama properti cache utama Anda. Nama properti cache harus unik dalam alur kerja Anda. Setiap tindakan dapat memiliki hingga lima entri. `FileCaching`

UI yang sesuai: Tab konfigurasi/File caching - Opsional/Tambahkan cache/Kunci

Path

(*/Caching/FileCaching/nama-aksi-nama-kunci-1/*) Path

(Opsional)

Tentukan jalur terkait untuk cache Anda.

UI yang sesuai: Tab konfigurasi/File caching - Opsional/Tambahkan cache/Path

RestoreKeys

(*/Caching/FileCaching/nama-aksi-nama-kunci-1/*) RestoreKeys

(Opsional)

Tentukan kunci pemulihan yang akan digunakan sebagai fallback ketika properti cache utama tidak dapat ditemukan. Kembalikan nama kunci harus unik dalam alur kerja Anda. Setiap cache dapat memiliki hingga lima entri. `RestoreKeys`

UI yang sesuai: Tab konfigurasi/File caching - Opsional/Tambahkan cache/Restore keys - opsional

Inputs

(*nama-tindakan/*) Inputs

(Opsional)

`Inputs`Bagian ini mendefinisikan data yang dibutuhkan tindakan selama menjalankan alur kerja.

> **Note**
>
> Maksimal empat input (satu sumber dan tiga artefak) diizinkan per tindakan build atau tindakan pengujian. Variabel tidak dihitung terhadap total ini.

Jika Anda perlu merujuk ke file yang berada di input yang berbeda (katakanlah sumber dan artefak), input sumber adalah input utama, dan artefak adalah input sekunder. Referensi ke file dalam input sekunder mengambil awalan khusus untuk menyisihkannya dari primer. Lihat perinciannya di Contoh: Merujuk file dalam beberapa artefak.

UI yang sesuai: Tab input

Sources

(nama-tindakan) /Inputs/ ***Sources***

(Opsional)

Tentukan label yang mewakili repositori sumber yang akan dibutuhkan oleh tindakan. Saat ini, satu-satunya label yang didukung adalah`WorkflowSource`, yang mewakili repositori sumber tempat file definisi alur kerja Anda disimpan.

Jika Anda menghilangkan sumber, maka Anda harus menentukan setidaknya satu artefak input di bawah. *action-name*`/Inputs/Artifacts`

Untuk informasi selengkapnya tentang sumber, lihat Menghubungkan alur kerja ke repositori sumber.

UI yang sesuai: tidak ada

Artifacts - input

(nama-tindakan) /Inputs/ ***Artifacts***

(Opsional)

Tentukan artefak dari tindakan sebelumnya yang ingin Anda berikan sebagai masukan untuk tindakan ini. Artefak ini harus sudah didefinisikan sebagai artefak keluaran dalam tindakan sebelumnya.

Jika Anda tidak menentukan artefak input apa pun, maka Anda harus menentukan setidaknya satu repositori sumber di bawah. *action-name*`/Inputs/Sources`

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

> **Note**
>
> Jika Artefacts - daftar drop-down opsional tidak tersedia (editor visual), atau jika Anda mendapatkan kesalahan saat memvalidasi YAMG (editor YAMM), itu mungkin karena tindakan hanya mendukung satu input. Dalam hal ini, coba hapus input sumber.

UI yang sesuai: Tab masukan/Artefak - opsional

Variables - input

(nama-tindakan) /Inputs/ ***Variables***

(Opsional)

Tentukan urutan pasangan nama/nilai yang menentukan variabel input yang ingin Anda sediakan untuk tindakan. Nama variabel terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama variabel.

Untuk informasi selengkapnya tentang variabel, termasuk contoh, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

UI yang sesuai: Input tab/Variabel - opsional

Outputs

(nama-tindakan/) Outputs

(Opsional)

Mendefinisikan data yang dihasilkan oleh tindakan selama menjalankan alur kerja.

UI yang sesuai: Tab keluaran

Artifacts - output

(nama-tindakan) /Outputs/ ***Artifacts***

(Opsional)

Tentukan nama artefak yang dihasilkan oleh tindakan. Nama artifak harus unik dalam alur kerja, dan terbatas pada karakter alfanumerik (a-z, A-Z, 0-9) dan garis bawah (_). Spasi, tanda hubung (-), dan karakter khusus lainnya tidak diperbolehkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan spasi, tanda hubung, dan karakter khusus lainnya dalam nama artefak keluaran.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab keluaran/Artefak

Name

(nama-tindakan) /Outputs/Artifacts/ ***Name***

(Diperlukan Artifacts - output jika disertakan)

Tentukan nama artefak yang dihasilkan oleh tindakan. Nama artifak harus unik dalam alur kerja, dan terbatas pada karakter alfanumerik (a-z, A-Z, 0-9) dan garis bawah (_). Spasi, tanda hubung (-), dan karakter khusus lainnya tidak diperbolehkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan spasi, tanda hubung, dan karakter khusus lainnya dalam nama artefak keluaran.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab keluaran/Artefak/Output baru/Bangun nama artefak

Files

(nama-tindakan) /Outputs/Artifacts/ ***Files***

(Diperlukan Artifacts - output jika disertakan)

Tentukan file yang CodeCatalyst termasuk dalam artefak yang dihasilkan oleh tindakan. File-file ini dihasilkan oleh tindakan alur kerja saat dijalankan, dan juga tersedia di repositori sumber Anda. Jalur file dapat berada di repositori sumber atau artefak dari tindakan sebelumnya, dan relatif terhadap repositori sumber atau root artefak. Anda dapat menggunakan pola glob untuk menentukan jalur. Contoh:

- Untuk menentukan satu file yang ada di root lokasi build atau lokasi repositori sumber, gunakan. `my-file.jar`
- Untuk menentukan satu file dalam subdirektori, gunakan `directory/my-file.jar` atau`directory/subdirectory/my-file.jar`.
- Untuk menentukan semua file, gunakan`"**/*"`. Pola `**` glob menunjukkan untuk mencocokkan sejumlah subdirektori.
- Untuk menentukan semua file dan direktori dalam direktori bernama`directory`, gunakan`"directory/**/*"`. Pola `**` glob menunjukkan untuk mencocokkan sejumlah subdirektori.

- Untuk menentukan semua file dalam direktori bernama`directory`, tetapi tidak salah satu subdirektorinya, gunakan. `"directory/*"`

> **Note**
>
> Jika jalur file Anda menyertakan satu atau beberapa tanda bintang (*) atau karakter khusus lainnya, lampirkan jalur dengan tanda kutip ganda (). "" Untuk informasi selengkapnya tentang karakter khusus, lihatPedoman dan konvensi sintaks.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

> **Note**
>
> Anda mungkin perlu menambahkan awalan ke jalur file untuk menunjukkan artefak atau sumber mana yang akan menemukannya. Untuk informasi selengkapnya, lihat Mereferensikan file dalam repositori sumber dan Merujuk file dalam artefak.

UI yang sesuai: Tab keluaran/Artefak/Output baru/File yang diproduksi oleh build

Variables - output

(nama-tindakan) /Outputs/ ***Variables***

(Opsional)

Tentukan variabel yang ingin Anda ekspor tindakan sehingga tersedia untuk digunakan oleh tindakan selanjutnya.

Untuk informasi selengkapnya tentang variabel, termasuk contoh, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

UI yang sesuai: Tab keluaran/Variabel/Tambahkan variabel

variabel-nama-1

(nama-aksi-variabel-nama-1/Outputs/Variables/)

(Opsional)

Tentukan nama variabel yang ingin Anda ekspor tindakan. Variabel ini harus sudah didefinisikan dalam `Inputs` atau `Steps` bagian dari tindakan yang sama.

Untuk informasi selengkapnya tentang variabel, termasuk contoh, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

UI yang sesuai: Tab keluaran/variabel/Tambahkan variabel/Nama

AutoDiscoverReports

(nama-tindakan) /Outputs/ ***AutoDiscoverReports***

(Opsional)

Mendefinisikan konfigurasi untuk fitur auto-discovery.

Saat Anda mengaktifkan penemuan otomatis, semua CodeCatalyst pencarian `Inputs` diteruskan ke tindakan serta semua file yang dihasilkan oleh tindakan itu sendiri, mencari laporan pengujian, cakupan kode, dan analisis komposisi perangkat lunak (SCA). Untuk setiap laporan yang ditemukan, CodeCatalyst mengubahnya menjadi CodeCatalyst laporan. CodeCatalyst Laporan adalah laporan yang sepenuhnya terintegrasi ke dalam CodeCatalyst layanan dan dapat dilihat dan dimanipulasi melalui CodeCatalyst konsol.

> **Note**
>
> Secara default, fitur auto-discover memeriksa semua file. Anda dapat membatasi file mana yang diperiksa menggunakan ExcludePaths properti IncludePaths atau.

UI yang sesuai: Tab Keluaran/Laporan/Laporan temukan otomatis

Enabled

(nama-tindakan) /Outputs/AutoDiscoverReports/ ***Enabled***

(Opsional)

Aktifkan atau nonaktifkan fitur penemuan otomatis.

Nilai-nilai yang valid adalah `true` atau `false`.

Jika `Enabled` dihilangkan, defaultnya adalah. `true`

UI yang sesuai: Tab Keluaran/Laporan/Laporan temukan otomatis

ReportNamePrefix

(nama-tindakan) /Outputs/AutoDiscoverReports/ ***ReportNamePrefix***

(Diperlukan AutoDiscoverReports jika disertakan dan diaktifkan)

Tentukan CodeCatalyst awalan yang ditambahkan ke semua laporan yang ditemukannya untuk memberi nama laporan terkait. CodeCatalyst Misalnya, jika Anda menentukan awalan`AutoDiscovered`, dan CodeCatalyst otomatis menemukan dua laporan pengujian, `TestSuiteOne.xml` dan`TestSuiteTwo.xml`, maka CodeCatalyst laporan terkait akan diberi nama dan. `AutoDiscoveredTestSuiteOne` `AutoDiscoveredTestSuiteTwo`

UI yang sesuai: Tab Keluaran/Laporan/Nama awalan

IncludePaths

(nama-tindakan) /Outputs/AutoDiscoverReports/ ***IncludePaths***

Atau

(aksi-nama-laporan-nama-1 /Outputs/Reports//) ***IncludePaths***

(Diperlukan jika AutoDiscoverReports disertakan dan diaktifkan, atau Reports jika disertakan)

Tentukan file dan jalur file yang CodeCatalyst disertakan saat mencari laporan mentah. Misalnya, jika Anda menentukan`"/test/report/*"`, akan CodeCatalyst mencari seluruh image build yang digunakan oleh tindakan mencari `/test/report/*` direktori. Ketika menemukan direktori itu, CodeCatalyst maka cari laporan di direktori itu.

> **Note**
>
> Jika jalur file Anda menyertakan satu atau beberapa tanda bintang (*) atau karakter khusus lainnya, lampirkan jalur dengan tanda kutip ganda (). "" Untuk informasi selengkapnya tentang karakter khusus, lihatPedoman dan konvensi sintaks.

Jika properti ini dihilangkan, defaultnya adalah`"**/*"`, artinya pencarian mencakup semua file di semua jalur.

> **Note**
>
> Untuk laporan yang dikonfigurasi secara manual, `IncludePaths` harus berupa pola glob yang cocok dengan satu file.

UI yang sesuai:

- Tab Keluaran/Laporan/Auto-temukan laporan/Sertakan jalur/Kecualikan jalur/Sertakan jalur
- ***Tab keluaran/Laporan/Konfigurasikan laporan/nama-laporan/nama-laporan/ Sertakan jalur secara manual/Sertakan jalur***

ExcludePaths

(nama-tindakan) /Outputs/AutoDiscoverReports/ ***ExcludePaths***

Atau

(aksi-nama-laporan-nama-1 /Outputs/Reports//) ***ExcludePaths***

(Opsional)

Tentukan file dan jalur file yang CodeCatalyst dikecualikan saat mencari laporan mentah. Misalnya, jika Anda menentukan`"/test/my-reports/**/*"`, tidak CodeCatalyst akan mencari file di `/test/my-reports/` direktori. Untuk mengabaikan semua file dalam direktori, gunakan pola `**/*` glob.

> **Note**
>
> Jika jalur file Anda menyertakan satu atau beberapa tanda bintang (*) atau karakter khusus lainnya, lampirkan jalur dengan tanda kutip ganda (). "" Untuk informasi selengkapnya tentang karakter khusus, lihatPedoman dan konvensi sintaks.

UI yang sesuai:

- Tab Keluaran/Laporan/Auto-temukan laporan/Sertakan/Kecualikan jalur/Kecualikan jalur
- ***Tab keluaran/Laporan/Konfigurasikan laporan/nama-laporan/nama-laporan-1/sertakan/Kecualikan jalur/Kecualikan jalur secara manual***

SuccessCriteria

(nama-tindakan) /Outputs/AutoDiscoverReports/ ***SuccessCriteria***

Atau

(aksi-nama-laporan-nama-1 /Outputs/Reports//) ***SuccessCriteria***

(Opsional)

Tentukan kriteria keberhasilan untuk pengujian, cakupan kode, analisis komposisi perangkat lunak (SCA), dan laporan analisis statis (SA).

Untuk informasi selengkapnya, lihat Mengkonfigurasi kriteria keberhasilan untuk laporan.

UI yang sesuai: Tab keluaran/Laporan/Kriteria keberhasilan

PassRate

(nama-tindakan) /Outputs/AutoDiscoverReports/SuccessCriteria/ ***PassRate***

Atau

(aksi-nama-laporan-nama-1/Outputs/Reports/) /SuccessCriteria/ ***PassRate***

(Opsional)

Tentukan persentase pengujian dalam laporan pengujian yang harus lulus agar CodeCatalyst laporan terkait ditandai sebagai lulus. Nilai yang valid termasuk angka desimal. Misalnya:`50`,`60.5`. Kriteria tingkat kelulusan hanya diterapkan pada laporan pengujian. Untuk informasi selengkapnya tentang laporan pengujian, lihatLaporan pengujian.

UI yang sesuai: Tab keluaran/Laporan/Kriteria sukses/Tingkat kelulusan

LineCoverage

(nama-tindakan) /Outputs/AutoDiscoverReports/SuccessCriteria/ ***LineCoverage***

Atau

(aksi-nama-laporan-nama-1/Outputs/Reports/) /SuccessCriteria/ ***LineCoverage***

(Opsional)

Tentukan persentase baris dalam laporan cakupan kode yang harus dicakup agar CodeCatalyst laporan terkait ditandai sebagai lulus. Nilai yang valid termasuk angka desimal. Misalnya:`50`,`60.5`. Kriteria cakupan baris hanya diterapkan pada laporan cakupan kode. Untuk informasi selengkapnya tentang laporan cakupan kode, lihatLaporan cakupan kode.

UI yang sesuai: Tab keluaran/Laporan/Kriteria sukses/Cakupan garis

BranchCoverage

(*nama-tindakan) /Outputs/AutoDiscoverReports/SuccessCriteria/* ***BranchCoverage***

Atau

(aksi-nama-laporan-nama-1/Outputs/Reports/) /SuccessCriteria/ ***BranchCoverage***

(Opsional)

Tentukan persentase cabang dalam laporan cakupan kode yang harus dicakup agar CodeCatalyst laporan terkait ditandai sebagai lulus. Nilai yang valid termasuk angka desimal. Misalnya:`50`,`60.5`. Kriteria cakupan cabang hanya diterapkan pada laporan cakupan kode. Untuk informasi selengkapnya tentang laporan cakupan kode, lihatLaporan cakupan kode.

UI yang sesuai: Tab keluaran/Laporan/Kriteria sukses/Cakupan cabang

Vulnerabilities

(*nama-tindakan) /Outputs/AutoDiscoverReports/SuccessCriteria/* ***Vulnerabilities***

Atau

(aksi-nama-laporan-nama-1/Outputs/Reports/) /SuccessCriteria/ ***Vulnerabilities***

(Opsional)

Tentukan jumlah maksimum dan tingkat keparahan kerentanan yang diizinkan dalam laporan SCA agar CodeCatalyst laporan terkait ditandai sebagai lulus. Untuk menentukan kerentanan, Anda harus menentukan:

- Tingkat keparahan minimum kerentanan yang ingin Anda sertakan dalam hitungan. Nilai yang valid, dari yang paling parah hingga yang paling parah, adalah: `CRITICALHIGH,MEDIUM,,LOW,INFORMATIONAL`.

  Misalnya, jika Anda memilih`HIGH`, maka `HIGH` dan `CRITICAL` kerentanan akan dihitung.
- Jumlah maksimum kerentanan dari tingkat keparahan yang ditentukan yang Anda inginkan memungkinkan. Melebihi angka ini menyebabkan CodeCatalyst laporan ditandai sebagai gagal. Nilai yang valid adalah bilangan bulat.

Kriteria kerentanan hanya diterapkan pada laporan SCA. Untuk informasi lebih lanjut tentang laporan SCA, lihatLaporan analisis komposisi perangkat lunak.

Untuk menentukan tingkat keparahan minimum, gunakan `Severity` properti. Untuk menentukan jumlah kerentanan maksimum, gunakan `Number` properti.

UI yang sesuai: Tab Keluaran/Laporan/Kriteria sukses/Kerentanan

StaticAnalysisBug

(nama-tindakan) /Outputs/AutoDiscoverReports/SuccessCriteria/ ***StaticAnalysisBug***

Atau

(aksi-nama-laporan-nama-1/Outputs/Reports/) /SuccessCriteria/ ***StaticAnalysisBug***

(Opsional)

Tentukan jumlah maksimum dan tingkat keparahan bug yang diizinkan dalam laporan SA agar CodeCatalyst laporan terkait ditandai sebagai lulus. Untuk menentukan bug, Anda harus menentukan:

- Tingkat keparahan minimum bug yang ingin Anda sertakan dalam hitungan. Nilai yang valid, dari yang paling parah hingga yang paling parah, adalah: `CRITICALHIGH,MEDIUM,,LOW,INFORMATIONAL`.

Misalnya, jika Anda memilihHIGH, maka HIGH dan CRITICAL bug akan dihitung.

- Jumlah maksimum bug dari tingkat keparahan yang ditentukan yang Anda inginkan izin. Melebihi angka ini menyebabkan CodeCatalyst laporan ditandai sebagai gagal. Nilai yang valid adalah bilangan bulat.

Kriteria bug hanya diterapkan pada PyLint dan laporan ESLint SA. Untuk informasi lebih lanjut tentang laporan SA, lihatLaporan analisis statis.

Untuk menentukan tingkat keparahan minimum, gunakan Severity properti. Untuk menentukan jumlah kerentanan maksimum, gunakan Number properti.

UI yang sesuai: Tab keluaran/Laporan/Kriteria sukses/Bug

StaticAnalysisSecurity

(nama-tindakan) /Outputs/AutoDiscoverReports/SuccessCriteria/ ***StaticAnalysisSecurity***

Atau

(aksi-nama-laporan-nama-1/Outputs/Reports/) /SuccessCriteria/ ***StaticAnalysisSecurity***

(Opsional)

Tentukan jumlah maksimum dan tingkat keparahan kerentanan keamanan yang diizinkan dalam laporan SA agar CodeCatalyst laporan terkait ditandai sebagai lulus. Untuk menentukan kerentanan keamanan, Anda harus menentukan:

- Tingkat keparahan minimum kerentanan keamanan yang ingin Anda sertakan dalam hitungan. Nilai yang valid, dari yang paling parah hingga yang paling parah, adalah: CRITICALHIGH,MEDIUM,,LOW,INFORMATIONAL.

  Misalnya, jika Anda memilihHIGH, maka HIGH dan kerentanan CRITICAL keamanan akan dihitung.
- Jumlah maksimum kerentanan keamanan dari tingkat keparahan tertentu yang Anda inginkan memungkinkan. Melebihi angka ini menyebabkan CodeCatalyst laporan ditandai sebagai gagal. Nilai yang valid adalah bilangan bulat.

Kriteria kerentanan keamanan hanya diterapkan PyLint pada laporan ESLint SA. Untuk informasi lebih lanjut tentang laporan SA, lihatLaporan analisis statis.

Untuk menentukan tingkat keparahan minimum, gunakan `Severity` properti. Untuk menentukan jumlah kerentanan maksimum, gunakan `Number` properti.

UI yang sesuai: Tab keluaran/Laporan/Kriteria sukses/Kerentanan keamanan

StaticAnalysisQuality

(nama-tindakan) /Outputs/AutoDiscoverReports/SuccessCriteria/ ***StaticAnalysisQuality***

Atau

(aksi-nama-laporan-nama-1/Outputs/Reports/) /SuccessCriteria/ ***StaticAnalysisQuality***

(Opsional)

Tentukan jumlah maksimum dan tingkat keparahan masalah kualitas yang diizinkan dalam laporan SA agar CodeCatalyst laporan terkait ditandai sebagai lulus. Untuk menentukan masalah kualitas, Anda harus menentukan:

- Tingkat keparahan minimum dari masalah kualitas yang ingin Anda sertakan dalam hitungan. Nilai yang valid, dari yang paling parah hingga yang paling parah, adalah: `CRITICALHIGH`,`MEDIUM`,,`LOW`,`INFORMATIONAL`.

  Misalnya, jika Anda memilih`HIGH`, maka `HIGH` dan masalah `CRITICAL` kualitas akan dihitung.
- Jumlah maksimum masalah kualitas dari tingkat keparahan tertentu yang Anda inginkan izin. Melebihi angka ini menyebabkan CodeCatalyst laporan ditandai sebagai gagal. Nilai yang valid adalah bilangan bulat.

Kriteria masalah kualitas hanya diterapkan pada PyLint laporan ESLint SA. Untuk informasi lebih lanjut tentang laporan SA, lihatLaporan analisis statis.

Untuk menentukan tingkat keparahan minimum, gunakan `Severity` properti. Untuk menentukan jumlah kerentanan maksimum, gunakan `Number` properti.

UI yang sesuai: Tab keluaran/Laporan/Kriteria sukses/Masalah kualitas

StaticAnalysisFinding

(nama-tindakan) /Outputs/AutoDiscoverReports/SuccessCriteria/ ***StaticAnalysisFinding***

Atau

(aksi-nama-laporan-nama-1/Outputs/Reports/) /SuccessCriteria/ ***StaticAnalysisFinding***

(Opsional)

Tentukan jumlah maksimum dan tingkat keparahan temuan yang diizinkan dalam laporan SA untuk CodeCatalyst laporan terkait yang akan ditandai sebagai lulus. Untuk menentukan temuan, Anda harus menentukan:

- Tingkat keparahan minimum dari temuan yang ingin Anda sertakan dalam hitungan. Nilai yang valid, dari yang paling parah hingga yang paling parah, adalah: `CRITICALHIGH,MEDIUM,,LOW,INFORMATIONAL`.

  Misalnya, jika Anda memilih`HIGH`, maka `HIGH` dan `CRITICAL` temuan akan dihitung.
- Jumlah maksimum temuan dari tingkat keparahan yang ditentukan yang Anda inginkan memungkinkan. Melebihi angka ini menyebabkan CodeCatalyst laporan ditandai sebagai gagal. Nilai yang valid adalah bilangan bulat.

Temuan hanya diterapkan pada laporan SARIF SA. Untuk informasi lebih lanjut tentang laporan SA, lihatLaporan analisis statis.

Untuk menentukan tingkat keparahan minimum, gunakan `Severity` properti. Untuk menentukan jumlah kerentanan maksimum, gunakan `Number` properti.

UI yang sesuai: Tab Keluaran/Laporan/Kriteria sukses/Temuan

Reports

(nama-tindakan) /Outputs/ ***Reports***

(Opsional)

Bagian yang menentukan konfigurasi untuk laporan pengujian.

UI yang sesuai: Tab keluaran/Laporan

laporan-nama-1

(***aksi-nama-laporan-nama-1***/Outputs/Reports/)

(Diperlukan Reports jika disertakan)

Nama yang ingin Anda berikan ke CodeCatalyst laporan yang akan dihasilkan dari laporan mentah Anda.

UI yang sesuai: Tab keluaran/Laporan/Konfigurasi laporan/Nama laporan secara manual

Format

(*aksi-nama-laporan-nama-1 /Outputs/Reports//)* ***Format***

(Diperlukan Reports jika disertakan)

Tentukan format file yang Anda gunakan untuk laporan Anda. Nilai yang mungkin adalah sebagai berikut.

- Untuk laporan pengujian:
  - Untuk Cucumber JSON, tentukan Cucumber (editor visual) atau `CUCUMBERJSON` (editor YAMM).
  - Untuk JUnit XHTML, tentukan JUnit (editor visual) atau `JUNITXML` (editor YAMG).
  - Untuk NUnit XHTML, tentukan NUnit (editor visual) atau `NUNITXML` (editor YAMG).
  - Untuk NUnit 3 XHTML, tentukan nUnit3 (editor visual) atau `NUNIT3XML` (editor YAMG).
  - Untuk Visual Studio TRX, tentukan Visual Studio TRX (editor visual) atau `VISUALSTUDIOTRX` (editor YAMM).
  - Untuk TestNG XHTML, tentukan TestNG (editor visual) atau `TESTNGXML` (editor YAMG).
- Untuk laporan cakupan kode:
  - Untuk XHTML Clover, tentukan Clover (editor visual) atau `CLOVERXML` (editor YAMG).
  - Untuk Cobertura XHTML, tentukan Cobertura (editor visual) atau `COBERTURAXML` (editor YAMM).
  - Untuk JaCoCo XHTML, tentukan JaCoCo(editor visual) atau `JACOCOXML` (editor YAMM).
  - Untuk SimpleCov JSON yang dihasilkan oleh simplecov, bukan simplecov-json, tentukan Simplecov (editor visual) atau (editor YAMG). `SIMPLECOV`
- Untuk laporan analisis komposisi perangkat lunak (SCA):

- Untuk SARIF, tentukan SARIF (editor visual) atau `SARIFSCA` (editor YAMG).

UI yang sesuai: Tab Keluaran/Laporan/Konfigurasi laporan secara manual/Tambah/konfigurasikan laporan/nama-laporan-1/Jenis laporan dan format Laporan

Configuration

(*nama-tindakan/*) Configuration

(Wajib) Bagian di mana Anda dapat menentukan properti konfigurasi tindakan.

UI yang sesuai: Tab konfigurasi

Container

(*nama-tindakan) /Configuration/* ***Container***

(Opsional)

Tentukan gambar Docker, atau wadah, yang digunakan tindakan untuk menyelesaikan pemrosesannya. Anda dapat menentukan salah satu gambar aktif yang disertakan CodeCatalyst, atau Anda dapat menggunakan gambar Anda sendiri. Jika Anda memilih untuk menggunakan gambar Anda sendiri, itu dapat berada di Amazon ECR, Docker Hub, atau registri lain. Jika Anda tidak menentukan gambar Docker, tindakan menggunakan salah satu gambar aktif untuk pemrosesannya. Untuk informasi tentang gambar aktif yang digunakan secara default, lihatGambar aktif.

Untuk informasi selengkapnya tentang menentukan image Docker Anda sendiri, lihat. Menetapkan image Docker lingkungan runtime kustom ke suatu tindakan

UI yang sesuai: Gambar Docker lingkungan runtime - opsional

Registry

(*nama-tindakan) /Configuration/Container/* ***Registry***

(Diperlukan `Container` jika disertakan)

Tentukan registri tempat gambar Anda disimpan. Nilai yang valid meliputi:

- `CODECATALYST`(Editor YAMB)

  Gambar disimpan dalam CodeCatalyst registri.

- Docker Hub (editor visual) atau `DockerHub` (editor YAMB)

  Gambar disimpan dalam registri gambar Docker Hub.

- Registri lain (editor visual) atau `Other` (editor YAMB)

  Gambar disimpan dalam registri gambar khusus. Registri apa pun yang tersedia untuk umum dapat digunakan.

- Amazon Elastic Container Registry (editor visual) atau `ECR` (editor YAMM)

  Gambar disimpan dalam repositori gambar Amazon Elastic Container Registry. Untuk menggunakan gambar di repositori Amazon ECR, tindakan ini memerlukan akses ke Amazon ECR. Untuk mengaktifkan akses ini, Anda harus membuat peran IAM yang mencakup izin berikut dan kebijakan kepercayaan khusus. (Anda dapat mengubah peran yang ada untuk menyertakan izin dan kebijakan, jika Anda mau.)

  Peran IAM harus menyertakan izin berikut dalam kebijakan perannya:

  - `ecr:BatchCheckLayerAvailability`
  - `ecr:BatchGetImage`
  - `ecr:GetAuthorizationToken`
  - `ecr:GetDownloadUrlForLayer`

  Peran IAM harus menyertakan kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
```

```
        }
    ]
}
```

Untuk informasi selengkapnya tentang membuat peran IAM, lihat Membuat peran menggunakan kebijakan kepercayaan khusus (konsol) di Panduan Pengguna IAM.

Setelah Anda membuat peran, Anda harus menetapkannya ke tindakan melalui lingkungan. Untuk informasi selengkapnya, lihat Mengaitkan lingkungan, koneksi akun, dan peran IAM dengan tindakan alur kerja.

UI yang sesuai: Amazon Elastic Container Registry, Docker Hub, dan opsi registri lainnya

Image

(nama-tindakan) /Configuration/Container/ ***Image***

(Diperlukan `Container` jika disertakan)

Tentukan satu dari yang berikut ini:

- Jika Anda menggunakan `CODECATALYST` registri, atur gambar ke salah satu gambar aktif berikut:
  - `CodeCatalystLinux_x86_64:2024_03`
  - `CodeCatalystLinux_x86_64:2022_11`
  - `CodeCatalystLinux_Arm64:2024_03`
  - `CodeCatalystLinux_Arm64:2022_11`
  - `CodeCatalystLinuxLambda_x86_64:2024_03`
  - `CodeCatalystLinuxLambda_x86_64:2022_11`
  - `CodeCatalystLinuxLambda_Arm64:2024_03`
  - `CodeCatalystLinuxLambda_Arm64:2022_11`
  - `CodeCatalystWindows_x86_64:2022_11`
- Jika Anda menggunakan registri Docker Hub, atur gambar ke nama gambar Docker Hub dan tag opsional.

  Contoh: `postgres:latest`

- Jika Anda menggunakan registri Amazon ECR, atur gambar ke URI registri Amazon ECR.

Contoh: `111122223333.dkr.ecr.us-west-2.amazonaws.com/codecatalyst-ecs-image-repo`

- Jika Anda menggunakan registri kustom, atur gambar ke nilai yang diharapkan oleh registri kustom.

UI yang sesuai: Runtime environment docker image (jika registri **`CODECATALYST`** adalah), image Docker Hub (jika registri adalah Docker Hub), URL gambar ECR (jika registri adalah Amazon Elastic Container Registry), dan Image URL (jika registri adalah Registri lain).

Steps

(nama-tindakan) /Configuration/ ***Steps***

(Diperlukan)

Tentukan perintah shell yang ingin Anda jalankan selama tindakan untuk menginstal, mengonfigurasi, dan menjalankan alat build Anda.

Berikut adalah contoh cara membangun proyek npm:

```
Steps:
  - Run: npm install
  - Run: npm run build
```

Berikut adalah contoh cara menentukan jalur file:

```
Steps:
  - Run: cd $ACTION_BUILD_SOURCE_PATH_WorkflowSource/app  && cat file2.txt
  - Run: cd $ACTION_BUILD_SOURCE_PATH_MyBuildArtifact/build-output/  && cat file.txt
```

Untuk informasi selengkapnya tentang menentukan jalur file, lihat Mereferensikan file dalam repositori sumber danMerujuk file dalam artefak.

UI yang sesuai: Perintah tab konfigurasi/Shell

Packages

(nama-tindakan) /Configuration/ ***Packages***

(Opsional)

Bagian di mana Anda dapat menentukan repositori paket yang digunakan tindakan untuk menyelesaikan dependensi. Paket memungkinkan Anda menyimpan dan berbagi paket perangkat lunak yang digunakan untuk pengembangan aplikasi dengan aman.

Untuk informasi selengkapnya tentang paket, lihatPublikasikan dan bagikan paket perangkat lunak di CodeCatalyst.

UI yang sesuai: Tab konfigurasi/Paket

NpmConfiguration

(nama-tindakan) /Configuration/Packages/ ***NpmConfiguration***

(Diperlukan Packages jika disertakan)

Bagian yang mendefinisikan konfigurasi untuk format paket npm. Konfigurasi ini digunakan oleh tindakan selama menjalankan alur kerja.

Untuk informasi selengkapnya tentang konfigurasi paket npm, lihatMenggunakan npm.

UI yang sesuai: Tab konfigurasi/paket/Tambahkan konfigurasi/npm

PackageRegistries

(nama-tindakan) /Configuration/Packages/NpmConfiguration/ ***PackageRegistries***

(Diperlukan Packages jika disertakan)

Bagian di mana Anda dapat menentukan properti konfigurasi dari urutan repositori paket.

UI yang sesuai: Tab konfigurasi/paket/Tambahkan konfigurasi/npm/Tambahkan repositori paket

PackagesRepository

(nama-tindakan) /Configuration/Packages/NpmConfiguration/PackageRegistries/ ***PackagesRepository***

(Diperlukan Packages jika disertakan)

Tentukan nama repositori CodeCatalyst paket Anda yang ingin digunakan tindakan.

Jika Anda menentukan beberapa repositori default, repositori terakhir akan diprioritaskan.

Untuk informasi selengkapnya tentang repositori paket, lihat. Package repositori

UI yang sesuai: Tab konfigurasi/Paket/Tambah Konfigurasi/NPM/Tambah repositori paket/Package repository

Scopes

*(nama-tindakan) /Configuration/Packages/NpmConfiguration/PackageRegistries/***Scopes**

(Opsional)

Tentukan urutan cakupan yang ingin Anda tentukan dalam registri paket Anda. Saat mendefinisikan cakupan, repositori paket yang ditentukan dikonfigurasi sebagai registri untuk semua cakupan yang terdaftar. Jika paket dengan cakupan diminta melalui klien npm, itu akan menggunakan repositori itu alih-alih default. Setiap nama lingkup harus diawali dengan “@”.

Jika Anda menyertakan cakupan utama, repositori terakhir akan diprioritaskan.

Jika `Scopes` dihilangkan, maka repositori paket yang ditentukan dikonfigurasi sebagai registri default untuk semua paket yang digunakan oleh tindakan.

Untuk informasi selengkapnya tentang cakupan, lihat Ruang nama Package dan paket Scoped.

UI yang sesuai: Tab Konfigurasi/Paket/Tambah Konfigurasi/NPM/Tambah repositori paket/Lingkup - opsional

Pengujian dengan alur kerja

Di CodeCatalyst, Anda dapat menjalankan pengujian sebagai bagian dari tindakan alur kerja yang berbeda, seperti build dan test. Tindakan alur kerja ini semuanya dapat menghasilkan laporan berkualitas. Tindakan uji adalah tindakan alur kerja yang menghasilkan pengujian, cakupan kode, analisis komposisi perangkat lunak, dan laporan analisis statis. Laporan ini ditampilkan di CodeCatalyst konsol.

Topik

- Jenis laporan kualitas
- Menambahkan tindakan pengujian

- Melihat hasil tindakan uji
- Melewatkan tes yang gagal dalam suatu tindakan
- Mengintegrasikan universal-test-runner ke dalam tindakan uji
- Mengkonfigurasi laporan kualitas dalam suatu tindakan
- Mencoba kembali kasus uji laporan
- Praktik terbaik untuk pengujian di CodeCatalyst
- Properti SARIF didukung dalam analisis komposisi perangkat lunak dan laporan analisis statis

Jenis laporan kualitas

Tindakan CodeCatalyst pengujian Amazon mendukung jenis laporan kualitas berikut. Untuk contoh tentang cara memformat laporan ini di YAMAL Anda, lihatContoh laporan kualitas YAMAL.

Topik

- Laporan pengujian
- Laporan cakupan kode
- Laporan analisis komposisi perangkat lunak
- Laporan analisis statis

Laporan pengujian

Di CodeCatalyst, Anda dapat mengonfigurasi pengujian unit, pengujian integrasi, dan pengujian sistem yang berjalan selama build. Kemudian CodeCatalyst dapat membuat laporan yang berisi hasil tes Anda.

Anda dapat menggunakan laporan pengujian untuk membantu memecahkan masalah dengan pengujian Anda. Jika memiliki banyak laporan pengujian dari beberapa build, Anda dapat menggunakan laporan pengujian untuk melihat tingkat kegagalan guna membantu mengoptimalkan build.

Anda dapat menggunakan format file laporan pengujian berikut:

- Mentimun JSON (.json)
- JUnit XML (.xml)

- NUnit XML (.xml)
- NUnit3 XML (.xml.xml)
- TestNG XML (.xml.xl)
- Studio Visual TRX (.trx, .xml)

Laporan cakupan kode

Di CodeCatalyst, Anda dapat membuat laporan cakupan kode untuk pengujian Anda. CodeCatalyst menyediakan metrik cakupan kode berikut:

Cakupan garis

Mengukur berapa banyak pernyataan yang dicakup tes Anda. Pernyataan adalah instruksi tunggal, tidak termasuk komentar.

```
line coverage = (total lines covered)/(total number of lines)
```

Cakupan cabang

Mengukur berapa banyak cabang yang dicakup pengujian Anda dari setiap cabang yang mungkin dari struktur kontrol seperti `case` pernyataan `if` atau pernyataan.

```
branch coverage = (total branches covered)/(total number of branches)
```

Format file laporan cakupan kode berikut didukung:

- JaCoCo XML (.xml)
- SimpleCov JSON (dihasilkan oleh simplecov, bukan simplecov-json, .json)
- XHTML semanggi (versi 3, .xml.)
- XML (.xml)
- LCOV (.info)

Laporan analisis komposisi perangkat lunak

Di CodeCatalyst, Anda dapat menggunakan alat analisis komposisi perangkat lunak (SCA) untuk menganalisis komponen aplikasi Anda dan memeriksa kerentanan keamanan yang diketahui. Anda dapat menemukan dan mengurai laporan SARIF yang merinci kerentanan dengan berbagai tingkat

keparahan dan cara untuk memperbaikinya. Nilai keparahan yang valid, dari yang paling parah hingga yang paling parah, adalah:`CRITICAL`,`HIGH`,`MEDIUM`,,`LOW`,`INFORMATIONAL`.

Format file laporan SCA berikut didukung:

- SARIF (.sarif, .json)

Laporan analisis statis

Anda dapat menggunakan laporan analisis statis (SA) untuk mengidentifikasi cacat kode tingkat sumber. Di CodeCatalyst, Anda dapat membuat laporan SA untuk membantu menyelesaikan masalah dalam kode Anda sebelum Anda menerapkannya. Masalah ini termasuk bug, kerentanan keamanan, masalah kualitas, dan kerentanan lainnya. Nilai keparahan yang valid, dari yang paling parah hingga yang paling parah, adalah: `CRITICALHIGH`,`MEDIUM`,`LOW`,, dan`INFORMATIONAL`.

CodeCatalyst menyediakan metrik SA berikut:

Bug

Mengidentifikasi sejumlah kemungkinan bug yang ditemukan dalam kode sumber Anda. Bug ini dapat mencakup masalah terkait keamanan memori. Berikut ini adalah contoh bug.

```
// The while loop will inadvertently index into array x out-of-bounds
int x[64];
while (int n = 0; n <= 64; n++) {
  x[n] = 0;
}
```

Kerentanan keamanan

Mengidentifikasi sejumlah kemungkinan kerentanan keamanan yang ditemukan dalam kode sumber Anda. Kerentanan keamanan ini dapat mencakup masalah seperti menyimpan token rahasia Anda dalam teks biasa.

Masalah kualitas

Mengidentifikasi sejumlah kemungkinan masalah kualitas yang ditemukan dalam kode sumber Anda. Masalah kualitas ini dapat mencakup masalah mengenai konvensi gaya. Berikut ini adalah contoh masalah kualitas.

```
// The function name doesn't adhere to the style convention of camelCase
```

```
int SUBTRACT(int x, int y) {
  return x-y
}
```

Kerentanan lainnya

Mengidentifikasi sejumlah kemungkinan kerentanan lain yang ditemukan dalam kode sumber Anda.

CodeCatalyst mendukung format file laporan SA berikut:

- PyLint (.py)
- ESLint (.js, .jsx, .ts, .tsx)
- SARIF (.sarif, .json)

Menambahkan tindakan pengujian

Gunakan prosedur berikut untuk menambahkan tindakan pengujian ke CodeCatalyst alur kerja Anda.

Visual

Untuk menambahkan tindakan pengujian menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
3. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
4. Pilih Edit.
5. Pilih Visual.
6. Pilih Tindakan.
7. Dalam Tindakan, pilih Uji.
8. Pada tab Input dan Konfigurasi, lengkapi bidang sesuai dengan kebutuhan Anda. Untuk deskripsi setiap bidang, lihatMembangun dan menguji definisi YAMAL tindakan. Referensi ini memberikan informasi rinci tentang setiap bidang (dan nilai properti YAMAL yang sesuai) seperti yang muncul di YAMAL dan editor visual.
9. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.

10. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menambahkan tindakan build menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
3. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
4. Pilih Edit.
5. Pilih YAMAL.
6. Pilih Tindakan.
7. Dalam Tindakan, pilih Uji.
8. Ubah properti dalam kode YAMAL sesuai dengan kebutuhan Anda. Penjelasan tentang setiap properti yang tersedia disediakan diMembangun dan menguji definisi YAMAL tindakan.
9. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
10. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Definisi tindakan uji

Tindakan pengujian didefinisikan sebagai sekumpulan properti YAMAL di dalam file definisi alur kerja Anda. Untuk informasi tentang properti ini, lihat Membangun dan menguji definisi YAMAL tindakan diAlur kerja definisi YAMAL.

Melihat hasil tindakan uji

Gunakan petunjuk berikut untuk melihat hasil tindakan pengujian, termasuk log, laporan, dan variabel yang dihasilkan.

Untuk melihat hasil tindakan uji

1. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.

3. Dalam diagram alur kerja, pilih nama tindakan pengujian Anda, misalnya, Uji.
4. Untuk melihat log yang dihasilkan oleh tindakan, pilih Log. Log untuk berbagai fase tindakan ditampilkan. Anda dapat memperluas atau menciutkan log sesuai kebutuhan.
5. Untuk melihat laporan pengujian yang dihasilkan oleh tindakan pengujian, pilih Laporan, atau di panel navigasi, pilih Laporan. Untuk informasi selengkapnya, lihat Jenis laporan kualitas.
6. Untuk melihat konfigurasi yang digunakan untuk tindakan pengujian, pilih Konfigurasi. Untuk informasi selengkapnya, lihat Menambahkan tindakan pengujian.
7. Untuk melihat variabel yang digunakan oleh tindakan pengujian, pilih Variabel. Untuk informasi selengkapnya, lihat Mengkonfigurasi dan menggunakan variabel dalam alur kerja.

Melewatkan tes yang gagal dalam suatu tindakan

Jika tindakan Anda memiliki lebih dari satu perintah pengujian, Anda mungkin ingin mengizinkan perintah pengujian berikutnya dalam tindakan untuk dijalankan meskipun perintah sebelumnya gagal. Misalnya, dalam perintah berikut, Anda mungkin `test2` ingin menjalankan selalu, bahkan jika `test1` gagal.

```
Steps:
- Run: npm install
- Run: npm run test1
- Run: npm run test2
```

Biasanya, ketika sebuah langkah mengembalikan kesalahan, Amazon CodeCatalyst menghentikan tindakan alur kerja dan menandainya sebagai gagal. Anda dapat mengizinkan langkah-langkah tindakan untuk terus berjalan dengan mengarahkan output kesalahan ke. `null` Anda dapat melakukan ini dengan `2>/dev/null` menambahkan perintah. Dengan modifikasi ini, contoh sebelumnya akan terlihat seperti berikut.

```
Steps:
- Run: npm install
- Run: npm run test1 2>/dev/null
- Run: npm run test2
```

Dalam cuplikan kode kedua, status `npm install` perintah akan dihormati, tetapi kesalahan apa pun yang dikembalikan oleh `npm run test1` perintah akan diabaikan. Akibatnya `npm run test2` perintah dijalankan. Dengan melakukan ini, Anda dapat melihat kedua laporan sekaligus terlepas dari apakah terjadi kesalahan.

Mengintegrasikan universal-test-runner ke dalam tindakan uji

Tindakan uji terintegrasi dengan alat `universal-test-runner` baris perintah sumber terbuka. Alat ini menyediakan fitur pengujian lanjutan, seperti mencoba kembali satu atau lebih kasus uji dari laporan pengujian. `universal-test-runner`menggunakan Protokol Eksekusi Uji untuk menjalankan pengujian Anda untuk bahasa apa pun dalam kerangka kerja tertentu. `universal-test-runner`mendukung kerangka kerja berikut:

- Gradle
- Lelucon
- Maven
- pytest
- .NET

`universal-test-runner`diinstal hanya pada gambar yang dikuratori untuk tindakan pengujian. Jika Anda mengonfigurasi tindakan pengujian untuk menggunakan Docker Hub kustom atau Amazon ECR, Anda harus menginstal secara manual `universal-test-runner` untuk mengaktifkan fitur pengujian lanjutan. Untuk melakukannya, instal Node.js (14 atau lebih tinggi) pada gambar, lalu instal `universal-test-runner` melalui `npm` menggunakan perintah shell`- Run: npm install -g @aws/universal-test-runner`. Untuk informasi selengkapnya tentang menginstal Node.js di container Anda melalui perintah shell, lihat Menginstal dan Memperbarui Node Version Manager.

Untuk informasi lebih lanjut tentang`universal-test-runner`, lihat Apa itu universal-test-runner?

Visual

Untuk digunakan universal-test-runner dalam editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
3. Pilih nama alur kerja Anda.
4. Pilih Edit.
5. Pilih Visual.
6. Pilih Tindakan.
7. Dalam Tindakan, pilih Uji.

8. Pada tab Konfigurasi, lengkapi bidang perintah Shell dengan memperbarui kode sampel dengan kerangka kerja yang didukung pilihan Anda. Misalnya, untuk menggunakan kerangka kerja yang didukung, Anda akan menggunakan Run perintah yang mirip dengan berikut ini.

   ```
   - Run: run-tests <framework>
   ```

   Jika kerangka kerja yang Anda inginkan tidak didukung, pertimbangkan untuk menyumbangkan adaptor atau pelari khusus. Untuk deskripsi bidang perintah Shell, lihatSteps.
9. (Opsional) Pilih Validasi untuk memvalidasi kode YAMM alur kerja sebelum melakukan.
10. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk digunakan universal-test-runner di editor YAMM

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
3. Pilih nama alur kerja Anda.
4. Pilih Edit.
5. Pilih YAMAL.
6. Pilih Tindakan.
7. Dalam Tindakan, pilih Uji.
8. Ubah kode YAMM sesuai dengan kebutuhan Anda. Misalnya, untuk menggunakan kerangka kerja yang didukung, Anda akan menggunakan Run perintah yang mirip dengan berikut ini.

   ```
   Configuration:
     Steps:
       - Run: run-tests <framework>
   ```

   Jika kerangka kerja yang Anda inginkan tidak didukung, pertimbangkan untuk menyumbangkan adaptor atau pelari khusus. Untuk deskripsi properti Steps, lihatSteps.
9. (Opsional) Pilih Validasi untuk memvalidasi kode YAMM alur kerja sebelum melakukan.
10. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Mengkonfigurasi laporan kualitas dalam suatu tindakan

Bagian ini menjelaskan cara mengonfigurasi laporan kualitas dalam suatu tindakan.

Topik

Penemuan otomatis dan laporan manual

Ketika penemuan otomatis diaktifkan, CodeCatalyst mencari semua input yang diteruskan ke tindakan, dan semua file yang dihasilkan oleh tindakan itu sendiri, mencari pengujian, cakupan kode, analisis komposisi perangkat lunak (SCA), dan laporan analisis statis (SA). Anda dapat melihat dan memanipulasi masing-masing laporan ini di CodeCatalyst.

Anda juga dapat mengonfigurasi laporan mana yang dihasilkan secara manual. Anda dapat menentukan jenis laporan yang ingin Anda hasilkan serta format file. Untuk informasi selengkapnya, lihat Jenis laporan kualitas.

Mengkonfigurasi kriteria keberhasilan untuk laporan

Anda dapat mengatur nilai yang menentukan kriteria keberhasilan untuk pengujian, cakupan kode, analisis komposisi perangkat lunak (SCA), atau laporan analisis statis (SA).

Kriteria keberhasilan adalah ambang batas yang menentukan apakah laporan lolos atau gagal. CodeCatalyst pertama menghasilkan laporan Anda, yang dapat berupa pengujian, cakupan kode, SCA, atau laporan SA, dan kemudian menerapkan kriteria keberhasilan pada laporan yang dihasilkan. Ini kemudian menunjukkan apakah kriteria keberhasilan terpenuhi, dan sejauh mana. Jika ada laporan yang tidak memenuhi kriteria keberhasilan yang ditentukan, CodeCatalyst tindakan yang menentukan kriteria keberhasilan gagal.

Misalnya, ketika Anda menetapkan kriteria keberhasilan untuk laporan SCA Anda, nilai kerentanan yang valid mulai dari yang paling parah hingga yang paling tidak parah adalah:`CRITICAL`,,, `HIGH`MEDIUM,`LOW`. `INFORMATIONAL` Jika Anda menetapkan kriteria untuk memindai satu kerentanan pada `HIGH` tingkat keparahan, laporan akan gagal jika ada setidaknya satu kerentanan pada `HIGH` tingkat keparahan atau tidak ada kerentanan pada tingkat `HIGH` keparahan, tetapi setidaknya

satu kerentanan pada tingkat keparahan yang lebih tinggi, seperti satu kerentanan pada tingkat keparahan. `CRITICAL`

Jika Anda tidak menentukan kriteria keberhasilan, maka:

- CodeCatalyst Laporan yang dihasilkan berdasarkan laporan mentah Anda tidak akan menampilkan kriteria keberhasilan.
- Kriteria keberhasilan tidak akan digunakan untuk menentukan apakah tindakan alur kerja terkait lolos atau gagal.

Visual

Untuk mengkonfigurasi kriteria keberhasilan

1. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih alur kerja yang berisi tindakan yang menghasilkan laporan. Ini adalah laporan yang ingin Anda terapkan kriteria keberhasilannya. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
3. Pilih Edit.
4. Pilih Visual.
5. Dalam diagram alur kerja, pilih tindakan yang telah Anda konfigurasikan untuk menghasilkan CodeCatalyst laporan.
6. Pilih tab Output.
7. Di bawah Auto-temukan laporan atau di bawah Konfigurasi laporan secara manual, pilih Kriteria sukses.

   Kriteria keberhasilan muncul. Bergantung pada pilihan Anda sebelumnya, Anda mungkin melihat salah satu atau semua opsi ini:

   Tingkat kelulusan

   Tentukan persentase pengujian dalam laporan pengujian yang harus lulus agar CodeCatalyst laporan terkait ditandai sebagai lulus. Nilai yang valid termasuk angka desimal. Misalnya:`50`,`60.5`. Kriteria tingkat kelulusan hanya diterapkan pada laporan pengujian. Untuk informasi selengkapnya tentang laporan pengujian, lihatLaporan pengujian.

   Cakupan garis

Tentukan persentase baris dalam laporan cakupan kode yang harus dicakup agar CodeCatalyst laporan terkait ditandai sebagai lulus. Nilai yang valid termasuk angka desimal. Misalnya:`50`,`60.5`. Kriteria cakupan baris hanya diterapkan pada laporan cakupan kode. Untuk informasi selengkapnya tentang laporan cakupan kode, lihatLaporan cakupan kode.

Cakupan cabang

Tentukan persentase cabang dalam laporan cakupan kode yang harus dicakup agar CodeCatalyst laporan terkait ditandai sebagai lulus. Nilai yang valid termasuk angka desimal. Misalnya:`50`,`60.5`. Kriteria cakupan cabang hanya diterapkan pada laporan cakupan kode. Untuk informasi selengkapnya tentang laporan cakupan kode, lihatLaporan cakupan kode.

Kerentanan (SCA)

Tentukan jumlah maksimum dan tingkat keparahan kerentanan yang diizinkan dalam laporan SCA agar CodeCatalyst laporan terkait ditandai sebagai lulus. Untuk menentukan kerentanan, Anda harus menentukan:

- Tingkat keparahan minimum kerentanan yang ingin Anda sertakan dalam hitungan. Nilai yang valid, dari yang paling parah hingga yang paling parah, adalah: `CRITICAL`,`HIGH`,`MEDIUM`,,`LOW`,`INFORMATIONAL`.

  Misalnya, jika Anda memilih`HIGH`, maka `HIGH` dan `CRITICAL` kerentanan akan dihitung.
- Jumlah maksimum kerentanan dari tingkat keparahan yang ditentukan yang Anda inginkan memungkinkan. Melebihi angka ini menyebabkan CodeCatalyst laporan ditandai sebagai gagal. Nilai yang valid adalah bilangan bulat.

Kriteria kerentanan hanya diterapkan pada laporan SCA. Untuk informasi lebih lanjut tentang laporan SCA, lihatLaporan analisis komposisi perangkat lunak.

Bug

Tentukan jumlah maksimum dan tingkat keparahan bug yang diizinkan dalam laporan SA agar CodeCatalyst laporan terkait ditandai sebagai lulus. Untuk menentukan bug, Anda harus menentukan:

- Tingkat keparahan minimum bug yang ingin Anda sertakan dalam hitungan. Nilai yang valid, dari yang paling parah hingga yang paling parah, adalah: `CRITICALHIGH,MEDIUM,,LOW,INFORMATIONAL`.

  Misalnya, jika Anda memilih`HIGH`, maka `HIGH` dan `CRITICAL` bug akan dihitung.
- Jumlah maksimum bug dari tingkat keparahan yang ditentukan yang Anda inginkan izin. Melebihi angka ini menyebabkan CodeCatalyst laporan ditandai sebagai gagal. Nilai yang valid adalah bilangan bulat.

Kriteria bug hanya diterapkan pada PyLint laporan ESLint SA. Untuk informasi lebih lanjut tentang laporan SA, lihatLaporan analisis statis.

Kerentanan keamanan

Tentukan jumlah maksimum dan tingkat keparahan kerentanan keamanan yang diizinkan dalam laporan SA agar CodeCatalyst laporan terkait ditandai sebagai lulus. Untuk menentukan kerentanan keamanan, Anda harus menentukan:

- Tingkat keparahan minimum kerentanan keamanan yang ingin Anda sertakan dalam hitungan. Nilai yang valid, dari yang paling parah hingga yang paling parah, adalah: `CRITICALHIGH,MEDIUM,,LOW,INFORMATIONAL`.

  Misalnya, jika Anda memilih`HIGH`, maka `HIGH` dan kerentanan `CRITICAL` keamanan akan dihitung.
- Jumlah maksimum kerentanan keamanan dari tingkat keparahan tertentu yang Anda inginkan memungkinkan. Melebihi angka ini menyebabkan CodeCatalyst laporan ditandai sebagai gagal. Nilai yang valid adalah bilangan bulat.

Kriteria kerentanan keamanan hanya diterapkan PyLint pada laporan ESLint SA. Untuk informasi lebih lanjut tentang laporan SA, lihatLaporan analisis statis.

Masalah kualitas

Tentukan jumlah maksimum dan tingkat keparahan masalah kualitas yang diizinkan dalam laporan SA agar CodeCatalyst laporan terkait ditandai sebagai lulus. Untuk menentukan masalah kualitas, Anda harus menentukan:

- Tingkat keparahan minimum dari masalah kualitas yang ingin Anda sertakan dalam hitungan. Nilai yang valid, dari yang paling parah hingga yang paling parah, adalah: `CRITICAL`,`HIGH`,`MEDIUM`,,`LOW`,`INFORMATIONAL`.

  Misalnya, jika Anda memilih`HIGH`, maka `HIGH` dan masalah `CRITICAL` kualitas akan dihitung.

- Jumlah maksimum masalah kualitas dari tingkat keparahan yang ditentukan yang Anda inginkan memungkinkan. Melebihi angka ini menyebabkan CodeCatalyst laporan ditandai sebagai gagal. Nilai yang valid adalah bilangan bulat.

Kriteria masalah kualitas hanya diterapkan pada PyLint laporan ESLint SA. Untuk informasi lebih lanjut tentang laporan SA, lihatLaporan analisis statis.

8. Pilih Terapkan.
9. Jalankan alur kerja Anda agar CodeCatalyst menerapkan kriteria keberhasilan pada laporan mentah Anda, dan buat ulang CodeCatalyst laporan terkait dengan informasi kriteria keberhasilan yang disertakan. Untuk informasi selengkapnya, lihat Memulai alur kerja berjalan secara manual.

YAML

Untuk mengkonfigurasi kriteria keberhasilan

1. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih alur kerja yang berisi tindakan yang menghasilkan laporan. Ini adalah laporan yang ingin Anda terapkan kriteria keberhasilannya. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
3. Pilih Edit.
4. Pilih YAMAL.
5. Dalam diagram alur kerja, pilih tindakan yang telah Anda konfigurasikan untuk menghasilkan CodeCatalyst laporan.
6. Di panel detail, pilih tab Output.
7. Dalam tindakan, di `AutoDiscoverReports` bagian, atau di `Reports` bagian, tambahkan SuccessCriteriaproperti, bersama dengan`PassRate`,,`LineCoverage`,`BranchCoverage`,

`VulnerabilitiesStaticAnalysisBug`,`StaticAnalysisSecurity`, dan `StaticAnalysisQuality` properti.

Untuk penjelasan masing-masing properti ini, konsultasikan dengan[Membangun dan menguji definisi YAMAL tindakan](#).

8. Pilih Terapkan.
9. Jalankan alur kerja Anda agar CodeCatalyst menerapkan kriteria keberhasilan pada laporan mentah Anda, dan buat ulang CodeCatalyst laporan terkait dengan informasi kriteria keberhasilan yang disertakan. Untuk informasi selengkapnya tentang memulai alur kerja, lihat[Memulai alur kerja berjalan secara manual](#).

Contoh laporan kualitas YAMAL

Contoh berikut menunjukkan cara mengonfigurasi empat laporan secara manual: laporan pengujian, laporan cakupan kode, laporan analisis komposisi perangkat lunak, dan laporan analisis statis.

```
Reports:
  MyTestReport:
    Format: JUNITXML
    IncludePaths:
      - "*.xml"
    ExcludePaths:
      - report1.xml
      SuccessCriteria:
        PassRate: 90
  MyCoverageReport:
    Format: CLOVERXML
    IncludePaths:
      - output/coverage/jest/clover.xml
      SuccessCriteria:
        LineCoverage: 75
        BranchCoverage: 75
  MySCAReport:
    Format: SARIFSCA
    IncludePaths:
      - output/sca/reports.xml
      SuccessCriteria:
        Vulnerabilities:
          Number: 5
          Severity: HIGH
  MySAReport:
```

```
    Format: ESLINTJSON
    IncludePaths:
      - output/static/eslint.xml
      SuccessCriteria:
        StaticAnalysisBug:
          Number: 10
          Severity: MEDIUM
        StaticAnalysisSecurity:
          Number: 5
          Severity: CRITICAL
        StaticAnalysisQuality:
          Number: 0
          Severity: INFORMATIONAL
```

Mencoba kembali kasus uji laporan

Jika laporan Anda gagal karena beberapa kasus uji, Anda hanya dapat mencoba lagi pengujian individual tersebut. Ini memungkinkan Anda untuk dengan cepat memeriksa kualitas kasus pengujian Anda dan menentukan langkah selanjutnya untuk menyelesaikan masalah Anda, seperti melibatkan ketergantungan yang rusak, atau memulai alur kerja dijalankan kembali. Tindakan pengujian Anda menggabungkan `universal-test-runner` untuk mencoba lagi hanya kasus uji yang dipilih, bukan seluruh tindakan. Anda hanya dapat mencoba lagi satu set kasus uji yang dipilih per tindakan pada satu waktu, dan hanya mencoba lagi lima kali per laporan pengujian. Untuk informasi selengkapnya, lihat Mengintegrasikan universal-test-runner ke dalam tindakan uji.

> **Note**
>
> Jika Anda mencoba lagi kasus pengujian pada laporan, itu tidak akan berpengaruh pada status alur kerja yang menghasilkan laporan asli.

Gunakan petunjuk berikut untuk mencoba kembali kasus uji dalam laporan Anda.

Untuk mencoba kembali kasus uji laporan

1. Di panel navigasi, pilih Laporan.
2. Pilih nama laporan Anda. Anda dapat memfilter berdasarkan nama, status, repositori, cabang, atau jenis laporan.
3. Di bawah nama laporan, pilih Hasil.

4. Pilih kasus uji yang ingin Anda coba lagi, pilih Jalankan ulang, lalu pilih Kasus uji yang dipilih.
5. Setelah percobaan ulang Anda selesai, pilih Segarkan pada spanduk dan lihat hasil yang diperbarui.

Praktik terbaik untuk pengujian di CodeCatalyst

Saat menggunakan fitur pengujian yang disediakan oleh CodeCatalyst, kami sarankan Anda mengikuti praktik terbaik ini.

Topik

Penemuan otomatis

Saat mengonfigurasi tindakan CodeCatalyst, penemuan otomatis memungkinkan Anda secara otomatis menemukan output dari berbagai alat, seperti laporan pengujian JUnit, dan menghasilkan laporan yang relevan CodeCatalyst darinya. Penemuan otomatis membantu memastikan bahwa laporan terus dibuat meskipun nama atau jalur ke output yang ditemukan berubah. Ketika file baru ditambahkan, CodeCatalyst secara otomatis menemukan mereka dan menghasilkan laporan yang relevan. Namun, jika Anda menggunakan penemuan otomatis, penting untuk memperhitungkan beberapa aspek berikut dari fitur ini:

- Saat Anda mengaktifkan penemuan otomatis dalam tindakan Anda, semua laporan yang ditemukan secara otomatis dari jenis yang sama akan berbagi kriteria keberhasilan yang sama. Misalnya, kriteria bersama seperti tingkat kelulusan minimum akan berlaku untuk semua laporan pengujian yang ditemukan secara otomatis. Jika Anda memerlukan kriteria yang berbeda untuk laporan dari jenis yang sama, Anda harus secara eksplisit mengonfigurasi setiap laporan ini.
- Penemuan otomatis juga dapat menemukan laporan yang dihasilkan oleh dependensi Anda dan, jika kriteria keberhasilan dikonfigurasi, mungkin gagal dalam tindakan pada laporan ini. Masalah ini dapat diatasi dengan memperbarui konfigurasi jalur pengecualian.
- Penemuan otomatis tidak dijamin menghasilkan daftar laporan yang sama setiap saat, karena memindai tindakan saat runtime. Jika Anda ingin laporan tertentu selalu diproduksi, Anda harus mengonfigurasi laporan secara eksplisit. Misalnya, jika pengujian berhenti berjalan sebagai

bagian dari build Anda, kerangka pengujian tidak akan menghasilkan output apa pun dan, sebagai hasilnya, tidak ada laporan pengujian yang akan dihasilkan dan tindakan mungkin berhasil. Jika Anda ingin keberhasilan tindakan Anda bergantung pada pengujian tertentu, maka Anda harus mengonfigurasi laporan itu secara eksplisit.

> **Tip**
>
> Saat memulai proyek baru atau yang sudah ada, gunakan penemuan otomatis untuk seluruh direktori proyek (termasuk`**/*`). Ini memanggil pembuatan laporan di semua file dalam proyek Anda, termasuk yang ada di dalam subdirektori.

Untuk informasi selengkapnya, lihat Mengkonfigurasi laporan kualitas dalam suatu tindakan.

Kriteria keberhasilan

Anda dapat menerapkan ambang kualitas pada laporan Anda dengan mengonfigurasi kriteria keberhasilan. Misalnya, jika dua laporan cakupan kode ditemukan secara otomatis, satu dengan cakupan baris 80% dan yang lainnya dengan cakupan garis 60%, Anda memiliki opsi berikut:

- Tetapkan kriteria keberhasilan penemuan otomatis untuk cakupan lini sebesar 80%. Ini akan menyebabkan laporan pertama lulus dan laporan kedua gagal, yang akan mengakibatkan tindakan keseluruhan gagal. Untuk membuka blokir alur kerja, tambahkan pengujian baru ke proyek Anda hingga cakupan baris untuk laporan kedua melebihi 80%.
- Tetapkan kriteria keberhasilan penemuan otomatis untuk cakupan lini pada 60%. Ini akan menyebabkan kedua laporan lolos, yang akan menghasilkan tindakan berhasil. Anda kemudian dapat bekerja untuk meningkatkan cakupan kode di laporan kedua. Namun, dengan pendekatan ini, Anda tidak dapat menjamin bahwa cakupan dalam laporan pertama tidak turun di bawah 80%.
- Konfigurasikan satu atau kedua laporan secara eksplisit dengan menggunakan editor visual atau menambahkan bagian dan jalur YAMB eksplisit untuk setiap laporan. Ini akan memungkinkan Anda untuk mengonfigurasi kriteria keberhasilan terpisah dan nama khusus untuk setiap laporan. Namun, dengan pendekatan ini, tindakan bisa gagal jika jalur laporan berubah.

Untuk informasi selengkapnya, lihat Mengkonfigurasi kriteria keberhasilan untuk laporan.

Sertakan/kecualikan jalur

Saat meninjau hasil tindakan, Anda dapat menyesuaikan daftar laporan yang dihasilkan CodeCatalyst dengan mengonfigurasi dan. `IncludePaths` `ExcludePaths`

- Gunakan `IncludePaths` untuk menentukan file dan jalur file yang CodeCatalyst ingin Anda sertakan saat mencari laporan. Misalnya, jika Anda menentukan`"/test/report/*"`, akan CodeCatalyst mencari seluruh image build yang digunakan oleh tindakan mencari `/test/report/` direktori. Ketika menemukan direktori itu, CodeCatalyst maka cari laporan di direktori itu.

  > **Note**
  >
  > Untuk laporan yang dikonfigurasi secara manual, `IncludePaths` harus berupa pola glob yang cocok dengan satu file.

- Gunakan `ExcludePaths` untuk menentukan file dan jalur file yang CodeCatalyst ingin Anda kecualikan saat mencari laporan. Misalnya, jika Anda menentukan`"/test/reports/**/*"`, tidak CodeCatalyst akan mencari file di `/test/reports/` direktori. Untuk mengabaikan semua file dalam direktori, gunakan pola `**/*` glob.

Berikut ini adalah contoh pola glob yang mungkin.

Pola	Deskripsi
`*.*`	Cocokkan semua nama objek di direktori saat ini yang berisi titik
`*.xml`	Cocokkan semua nama objek di direktori saat ini yang diakhiri dengan `.xml`
`*.{xml,txt}`	Cocokkan semua nama objek di direktori saat ini yang diakhiri dengan `.xml` atau `.txt`
`**/*.xml`	Mencocokkan nama objek di semua direktori yang diakhiri dengan `.xml`
`testFolder`	Cocokkan objek yang disebut`testFolder` , memperlakukannya sebagai file

Pola	Deskripsi
`testFolder/*`	Mencocokkan objek dalam satu tingkat subfolder dari`testFolder` , seperti `testFolder/file.xml`
`testFolder/*/*`	Mencocokkan objek dalam dua tingkat subfolder dari`testFolder` , seperti `testFolder/reportsFolder/file.xml`
`testFolder/**`	Cocok dengan subfolder `testFolder` serta file di bawah `testFolder` , seperti `testFolder/file.xml` dan `testFolder/otherFolder/file.xml`

CodeCatalyst menafsirkan pola glob sebagai berikut:

- Karakter slash (/) memisahkan direktori di jalur file.
- Karakter tanda bintang (*) cocok dengan karakter nol atau karakter lain dari komponen nama tanpa melintasi batas folder.
- Tanda bintang ganda (**) cocok dengan nol atau lebih karakter dari komponen nama di semua direktori.

> **Note**
>
> `ExcludePaths`lebih diutamakan. `IncludePaths` Jika keduanya `IncludePaths` dan `ExcludePaths` menyertakan folder yang sama, folder itu tidak dipindai untuk laporan.

Properti SARIF didukung dalam analisis komposisi perangkat lunak dan laporan analisis statis

SARIF (Static Analysis Results Interchange Format) adalah format file output yang tersedia dalam analisis komposisi perangkat lunak dan laporan analisis statis di. CodeCatalyst Contoh berikut menunjukkan cara mengkonfigurasi SARIF secara manual dalam laporan analisis statis:

```
Reports:
MySAReport:
Format: SARIFSA
IncludePaths:
    - output/sa_report.json
SuccessCriteria:
    StaticAnalysisFinding:
    Number: 25
    Severity: HIGH
```

CodeCatalyst mendukung properti SARIF berikut yang dapat digunakan untuk mengoptimalkan bagaimana hasil analisis akan muncul dalam laporan Anda.

Topik

Objek **`sarifLog`**

Nama	Wajib	Deskripsi
`$schema`	Ya	URI dari skema SARIF JSON untuk versi 2.1.0.
`version`	Ya	CodeCatalyst hanya mendukung SARIF versi 2.1.0.
`runs[]`	Ya	File SARIF berisi array dari satu atau lebih run, yang

Nama	Wajib	Deskripsi
		masing-masing mewakili satu run alat analisis.

Objek **`run`**

Nama	Wajib	Deskripsi
`tool.driver`	Ya	`toolComponent` Objek yang menggambarkan alat analisis.
`tool.name`	Tidak	Properti yang menunjukkan nama alat yang digunakan untuk melakukan analisis.
`results[]`	Ya	Hasil alat analisis yang ditampilkan pada CodeCatalyst.

Objek **`toolComponent`**

Nama	Wajib	Deskripsi
`name`	Ya	Nama alat analisis.
`properties.artifactScanned`	Tidak	Sejumlah artefak dianalisis oleh alat.
`rules[]`	Ya	Sebuah array `reportingDescriptor` objek yang mewakili aturan. Berdasarkan aturan ini, alat analisis menemukan masalah dalam kode yang dianalisis.

Objek **`reportingDescriptor`**

Nama	Wajib	Deskripsi
`id`	Ya	Pengidentifikasi unik untuk aturan yang digunakan untuk referensi temuan. Panjang maksimal: 1.024 karakter
`name`	Tidak	Nama tampilan aturan. Panjang maksimal: 1.024 karakter
`shortDescription.text`	Tidak	Deskripsi singkat tentang aturan tersebut. Panjang maksimum: 3.000 karakter
`fullDescription.text`	Tidak	Deskripsi lengkap aturan. Panjang maksimum: 3.000 karakter
`helpUri`	Tidak	String yang dapat dilokalkan untuk berisi URI absolut dari dokumentasi utama untuk aturan. Panjang maksimum: 3.000 karakter
`properties.unscore`	Tidak	Bendera yang menunjukkan apakah temuan pemindaian telah dinilai.

Nama	Wajib	Deskripsi
`properties.score.severity`	Tidak	Satu set string tetap yang menentukan tingkat keparahan temuan. Panjang maksimal: 1.024 karakter
`properties.cvssv3_baseSeverity`	Tidak	Peringkat keparahan kualitatif dari Common Vulnerability Scoring System v3.1.
`properties.cvssv3_baseScore`	Tidak	Skor Dasar CVSS v3 mulai dari 0,0 - 10,0.
`properties.cvssv2_severity`	Tidak	Jika nilai CVSS v3 tidak tersedia, CodeCatalyst cari nilai CVSS v2.
`properties.cvssv2_score`	Tidak	Skor Dasar CVSS v2 mulai dari 0,0 - 10,0.
`properties.severity`	Tidak	Satu set string tetap yang menentukan tingkat keparahan temuan. Panjang maksimal: 1.024 karakter
`defaultConfiguration.level`	Tidak	Tingkat keparahan default aturan.

Objek **`result`**

Nama	Wajib	Deskripsi
`ruleId`	Ya	Pengidentifikasi unik untuk aturan yang digunakan untuk referensi temuan. Panjang maksimal: 1.024 karakter
`ruleIndex`	Ya	Indeks aturan terkait dalam komponen alat`rules[]`.
`message.text`	Ya	Pesan yang menjelaskan hasil dan menampilkan pesan untuk setiap temuan. Panjang maksimum: 3.000 karakter
`rank`	Tidak	Nilai antara 0,0 hingga 100,0 inklusif yang mewakili prioritas atau pentingnya hasil. Nilai skala ini 0,0 menjadi prioritas terendah dan 100,0 menjadi prioritas tertinggi.
`level`	Tidak	Tingkat keparahan hasilnya. Panjang maksimal: 1.024 karakter
`properties.unscore`	Tidak	Bendera yang menunjukkan apakah temuan pemindaian telah dinilai.

Nama	Wajib	Deskripsi
`properties.score.severity`	Tidak	Satu set string tetap yang menentukan tingkat keparahan temuan. Panjang maksimal: 1.024 karakter
`properties.cvssv3_baseSeverity`	Tidak	Peringkat keparahan kualitatif dari Common Vulnerability Scoring System v3.1.
`properties.cvssv3_baseScore`	Tidak	Skor Dasar CVSS v3 mulai dari 0,0 - 10,0.
`properties.cvssv2_severity`	Tidak	Jika nilai CVSS v3 tidak tersedia, CodeCatalyst cari nilai CVSS v2.
`properties.cvssv2_score`	Tidak	Skor Dasar CVSS v2 mulai dari 0,0 - 10,0.
`properties.severity`	Tidak	Satu set string tetap yang menentukan tingkat keparahan temuan. Panjang maksimal: 1.024 karakter

Nama	Wajib	Deskripsi
`locations[]`	Ya	Kumpulan lokasi di mana hasilnya terdeteksi. Hanya satu lokasi yang harus disertakan kecuali masalahnya hanya dapat diperbaiki dengan membuat perubahan di setiap lokasi yang ditentukan. CodeCatalyst menggunakan nilai pertama dalam array lokasi untuk membubuhi keterangan hasilnya. Jumlah maksimum `location` objek: 10
`relatedLocations[]`	Tidak	Daftar referensi lokasi tambahan dalam temuan. Jumlah maksimum `location` objek: 50
`fixes[]`	Tidak	Array `fix` objek yang mewakili rekomendasi yang disediakan oleh alat pemindaian. CodeCatalyst menggunakan rekomendasi pertama dalam `fixes` array.

Objek **`location`**

Nama	Wajib	Deskripsi
`physicalLocation`	Ya	Mengidentifikasi artefak dan wilayah.

Nama	Wajib	Deskripsi
`logicalLocations[]`	Tidak	Kumpulan lokasi dijelaskan dengan nama tanpa mengacu pada artefak.

Objek **`physicalLocation`**

Nama	Wajib	Deskripsi
`artifactLocation.uri`	Ya	URI yang menunjukkan lokasi artefak, biasanya file baik di repositori atau dihasilkan selama pembuatan.
`fileLocation.uri`	Tidak	URI jatuh kembali yang menunjukkan lokasi file. Ini digunakan jika `artifactLocation.uri` kembali kosong.
`region.startLine`	Ya	Nomor baris karakter pertama di wilayah tersebut.
`region.startColumn`	Ya	Nomor kolom karakter pertama di wilayah tersebut.
`region.endLine`	Ya	Nomor baris karakter terakhir di wilayah tersebut.
`region.endColumn`	Ya	Nomor kolom karakter terakhir di wilayah tersebut.

Objek **`logicalLocation`**

Nama	Wajib	Deskripsi
`fullyQualifiedName`	Tidak	Informasi tambahan yang menggambarkan lokasi hasil. Panjang maksimal: 1.024 karakter

Objek **`fix`**

Nama	Wajib	Deskripsi
`description.text`	Tidak	Pesan yang menampilkan rekomendasi untuk setiap temuan. Panjang maksimum: 3.000 karakter
`artifactChanges.[0].artifactLocation.uri`	Tidak	URI menunjukkan lokasi artefak yang perlu diperbarui.

Menerapkan dengan alur kerja

Dengan menggunakan CodeCatalyst alur kerja, Anda dapat menerapkan aplikasi dan sumber daya lainnya ke berbagai target seperti Amazon ECS AWS Lambda, dan lainnya.

Bagaimana cara menyebarkan aplikasi?

Untuk menerapkan aplikasi atau sumber daya CodeCatalyst, pertama-tama Anda membuat alur kerja, lalu tentukan tindakan penerapan di dalamnya. Tindakan penerapan adalah blok pembangun alur kerja yang mendefinisikan apa yang ingin Anda terapkan, di mana Anda ingin menerapkannya,

dan bagaimana Anda ingin menerapkannya (misalnya, menggunakan skema biru/hijau). Anda menambahkan tindakan penerapan ke alur kerja menggunakan editor visual CodeCatalyst konsol, atau editor YAMAL.

Langkah-langkah tingkat tinggi untuk menyebarkan aplikasi atau sumber daya adalah sebagai berikut.

Untuk menyebarkan aplikasi (tugas tingkat tinggi)

1. Dalam CodeCatalyst proyek Anda, Anda menambahkan kode sumber untuk aplikasi yang ingin Anda terapkan. Untuk informasi selengkapnya, lihat Menyimpan kode sumber di repositori untuk proyek di CodeCatalyst.
2. Dalam CodeCatalyst proyek Anda, Anda membuat alur kerja. Alur kerja adalah tempat Anda menentukan cara membuat, menguji, dan menerapkan aplikasi Anda. Untuk informasi selengkapnya, lihat Memulai dengan alur kerja.
3. Dalam alur kerja, Anda menambahkan pemicu, tindakan build, dan secara opsional, tindakan pengujian. Lihat informasi selengkapnya di Memulai alur kerja berjalan secara otomatis dengan pemicu, Menambahkan aksi build, dan Menambahkan tindakan pengujian.
4. Dalam alur kerja, Anda menambahkan tindakan penerapan. Anda dapat memilih dari beberapa tindakan penerapan CodeCatalyst yang disediakan ke aplikasi Anda ke target yang berbeda, seperti Amazon ECS. (Anda juga dapat menggunakan tindakan build atau GitHub Action untuk menerapkan aplikasi Anda. Untuk informasi selengkapnya tentang tindakan dan GitHub Tindakan build, lihatAlternatif untuk menyebarkan tindakan.)
5. Anda memulai alur kerja baik secara manual atau otomatis melalui pemicu. Alur kerja menjalankan tindakan build, test, dan deploy secara berurutan untuk menerapkan aplikasi dan resource Anda ke target. Untuk informasi selengkapnya, lihat Memulai alur kerja berjalan secara manual.

Daftar tindakan penerapan

Tindakan penerapan berikut tersedia:

- Menyebarkan AWS CloudFormation tumpukan — Tindakan ini membuat CloudFormation tumpukan AWS berdasarkan AWS CloudFormation template atau AWS Serverless Application Model template yang Anda berikan. Untuk informasi selengkapnya, lihat Menerapkan AWS CloudFormation tumpukan dengan alur kerja.

- Terapkan ke Amazon ECS — Tindakan ini mendaftarkan file definisi tugas yang Anda berikan. Untuk informasi selengkapnya, lihat Menyebarkan aplikasi ke Amazon Elastic Container Service (ECS) dengan alur kerja.
- Terapkan ke klaster Kubernetes — Tindakan ini menyebarkan aplikasi ke kluster Amazon Elastic Kubernetes Service. Untuk informasi selengkapnya, lihat Menerapkan aplikasi ke Amazon Elastic Kubernetes Service dengan alur kerja.
- AWS CDK deploy — Tindakan ini menyebarkan AWS CDK aplikasi ke dalam. AWS Untuk informasi selengkapnya, lihat Menerapkan AWS Cloud Development Kit (AWS CDK) aplikasi dengan alur kerja.

> **Note**
>
> Ada CodeCatalyst tindakan lain yang dapat menyebarkan sumber daya; namun, tindakan tersebut tidak dianggap sebagai tindakan penerapan karena informasi penerapannya tidak muncul di halaman Lingkungan. Untuk mempelajari lebih lanjut tentang halaman Lingkungan dan melihat penerapan, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst dan. Melihat status penerapan, komit, dan permintaan tarik

Manfaat tindakan penyebaran

Menggunakan tindakan penerapan dalam alur kerja memiliki manfaat sebagai berikut:

- Riwayat penyebaran — Lihat riwayat penerapan Anda untuk membantu mengelola dan mengkomunikasikan perubahan dalam perangkat lunak yang Anda gunakan.
- Keterlacakan — Lacak status penerapan Anda melalui CodeCatalyst konsol, dan lihat kapan dan di mana setiap revisi aplikasi diterapkan.
- Rollback — Gulung kembali penerapan secara otomatis jika ada kesalahan. Anda juga dapat mengonfigurasi alarm untuk mengaktifkan rollback penerapan.
- Monitoring - Perhatikan penyebaran Anda saat berlangsung melalui berbagai tahapan alur kerja Anda.
- Integrasi dengan CodeCatalyst fitur lain — Simpan kode sumber dan kemudian buat, uji, dan terapkan, semuanya dari satu aplikasi.

Alternatif untuk menyebarkan tindakan

Anda tidak harus menggunakan tindakan penerapan, meskipun mereka direkomendasikan karena mereka menawarkan manfaat yang diuraikan di bagian sebelumnya. Sebagai gantinya, Anda dapat menggunakan CodeCatalyst tindakan berikut:

- Sebuah tindakan membangun.

  Biasanya, Anda menggunakan tindakan build jika ingin menerapkan ke target yang tidak memiliki tindakan penerapan terkait, atau jika Anda ingin lebih mengontrol prosedur penerapan. Untuk informasi selengkapnya tentang penggunaan tindakan build untuk menyebarkan sumber daya, lihatMembangun dengan alur kerja.
- Sebuah GitHub tindakan.

  Anda dapat menggunakan GitHub Action di dalam CodeCatalyst alur kerja untuk menyebarkan aplikasi dan sumber daya (bukan CodeCatalyst tindakan). Untuk informasi tentang cara menggunakan GitHub Tindakan di dalam CodeCatalyst alur kerja, lihat Mengintegrasikan GitHub Tindakan ke dalam alur kerja

Anda juga dapat menggunakan AWS layanan berikut untuk menyebarkan aplikasi Anda, jika Anda tidak ingin menggunakan CodeCatalyst alur kerja untuk melakukannya:

- AWS CodeDeploy — lihat Apa itu CodeDeploy?
- AWS CodeBuild dan AWS CodePipeline — lihat Apa itu AWS CodeBuild? dan Apa itu AWS CodePipeline?
- AWS CloudFormation — lihat Apa itu AWS CloudFormation?

Gunakan CodeDeploy, CodeBuild, CodePipeline, dan CloudFormation layanan untuk penyebaran perusahaan yang kompleks.

Topik

- Menyebarkan aplikasi ke Amazon Elastic Container Service (ECS) dengan alur kerja
- Menerapkan aplikasi ke Amazon Elastic Kubernetes Service dengan alur kerja
- Menerapkan AWS CloudFormation tumpukan dengan alur kerja
- Menerapkan AWS Cloud Development Kit (AWS CDK) aplikasi dengan alur kerja
- Bootstrapping AWS CDK aplikasi dengan alur kerja

- Menerbitkan file ke Amazon S3 dengan alur kerja
- Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst
- Menampilkan URL aplikasi yang digunakan dalam diagram alur kerja
- Menghapus target penerapan
- Melacak status penerapan dengan komit
- Melihat log penerapan
- Melihat status penerapan, komit, dan permintaan tarik

Menyebarkan aplikasi ke Amazon Elastic Container Service (ECS) dengan alur kerja

Bagian ini menjelaskan cara menerapkan aplikasi kontainer ke dalam klaster Amazon ECS menggunakan alur kerja. CodeCatalyst Untuk mencapai hal ini, Anda harus menambahkan tindakan Deploy ke Amazon ECS ke alur kerja Anda. Tindakan ini mendaftarkan file definisi tugas yang Anda berikan. Setelah pendaftaran, definisi tugas akan dipakai oleh layanan Amazon ECS Anda yang berjalan di cluster Amazon ECS Anda. “Membuat instantiasi definisi tugas” setara dengan menerapkan aplikasi ke Amazon ECS.

Untuk menggunakan tindakan ini, Anda harus memiliki kluster Amazon ECS, layanan, dan file definisi tugas yang siap.

Untuk informasi selengkapnya tentang Amazon ECS, lihat Panduan Pengembang Layanan Kontainer Elastis Amazon.

> **Tip**
>
> Untuk tutorial yang menunjukkan cara menggunakan tindakan Deploy to Amazon ECS, lihat. Tutorial: Menyebarkan aplikasi ke Amazon ECS

> **Tip**
>
> Untuk contoh kerja tindakan Deploy to Amazon ECS, buat project dengan API Node.js dengan AWS Fargate atau Java API dengan AWS Fargate cetak biru. Untuk informasi selengkapnya, lihat Membuat proyek dengan cetak biru.

Topik

Tutorial: Menyebarkan aplikasi ke Amazon ECS

Dalam tutorial ini, Anda mempelajari cara menerapkan aplikasi tanpa server ke Amazon Elastic Container Service (Amazon ECS) menggunakan alur kerja, Amazon ECS, dan beberapa layanan lainnya. AWS Aplikasi yang digunakan adalah situs web Hello World sederhana yang dibangun di atas image Docker server web Apache. Tutorial memandu Anda melalui pekerjaan persiapan yang diperlukan seperti menyiapkan cluster, dan kemudian menjelaskan cara membuat alur kerja untuk membangun dan menyebarkan aplikasi.

> **Tip**
>
> Alih-alih mengerjakan tutorial ini, Anda dapat menggunakan cetak biru yang melakukan pengaturan Amazon ECS lengkap untuk Anda. Anda harus menggunakan API Node.js dengan AWS Fargate atau Java API dengan AWS Fargate cetak biru. Untuk informasi selengkapnya, lihat Membuat proyek dengan cetak biru.

Topik

- Bersihkan

Prasyarat

Sebelum Anda memulai:

- Anda membutuhkan CodeCatalyst ruang dengan AWS akun yang terhubung. Untuk informasi selengkapnya, lihat Menciptakan ruang.
- Di ruang Anda, Anda memerlukan CodeCatalyst proyek kosong, Mulai dari awal yang disebut:

  ```
  codecatalyst-ecs-project
  ```

  Untuk informasi selengkapnya, lihat Membuat proyek kosong di Amazon CodeCatalyst.
- Dalam proyek Anda, Anda memerlukan CodeCatalyst lingkungan yang disebut:

  ```
  codecatalyst-ecs-environment
  ```

  Konfigurasikan lingkungan ini sebagai berikut:
  - Pilih jenis apa saja, seperti Non-produksi.
  - Hubungkan AWS akun Anda ke sana.

  Untuk informasi selengkapnya, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst.

Langkah 1: Siapkan AWS pengguna dan AWS CloudShell

Langkah pertama dalam tutorial ini adalah membuat pengguna AWS IAM Identity Center, dan meluncurkan AWS CloudShell instance sebagai pengguna ini. Selama tutorial ini, CloudShell adalah komputer pengembangan Anda dan di mana Anda mengkonfigurasi AWS sumber daya dan layanan. Hapus pengguna ini setelah menyelesaikan tutorial.

> **Note**
>
> Jangan gunakan pengguna root Anda untuk tutorial ini. Anda harus membuat pengguna terpisah atau Anda mungkin mengalami masalah saat melakukan tindakan di AWS Command Line Interface (CLI) nanti.

Untuk informasi selengkapnya tentang pengguna Pusat Identitas IAM dan CloudShell, lihat Panduan AWS IAM Identity Center Pengguna dan Panduan AWS CloudShell Pengguna.

Untuk membuat pengguna IAM Identity Center

1. Masuk ke AWS Management Console dan buka AWS IAM Identity Center konsol di https://console.aws.amazon.com/singlesignon/.

   > **Note**
   >
   > Pastikan Anda masuk menggunakan Akun AWS yang terhubung ke CodeCatalyst ruang Anda. Anda dapat memverifikasi akun mana yang terhubung dengan menavigasi ke ruang Anda dan memilih tab akun AWS. Untuk informasi selengkapnya, lihat Menciptakan ruang.

2. Pada panel navigasi, silakan pilih Pengguna, lalu pilih Tambahkan pengguna.
3. Di Nama Pengguna, masukkan:

   ```
   CodeCatalystECSUser
   ```

4. Di bawah Kata Sandi, pilih Buat kata sandi satu kali yang dapat Anda bagikan dengan pengguna ini.
5. Di Alamat Email dan Konfirmasi alamat email, masukkan alamat email yang belum ada di Pusat Identitas IAM.
6. Di Nama depan dan nama belakang, masukkan:

   ```
   CodeCatalystECSUser
   ```

7. Di Nama tampilan, simpan nama yang dihasilkan secara otomatis:

   ```
   CodeCatalystECSUser CodeCatalystECSUser
   ```

8. Pilih Selanjutnya.
9. Pada halaman Tambahkan pengguna ke grup, pilih Berikutnya.
10. Pada halaman Tinjau dan tambahkan pengguna, tinjau informasi dan pilih Tambah pengguna.

    Kotak dialog kata sandi satu kali muncul.

11. Pilih Salin lalu tempel informasi masuk, termasuk URL portal AWS akses dan kata sandi satu kali.
12. Pilih Tutup.

Untuk membuat set izin

Anda akan menetapkan izin ini disetel `CodeCatalystECSUser` nanti.

1. Di panel navigasi, pilih Set izin, lalu pilih Buat set izin.
2. Pilih Set izin yang telah ditentukan sebelumnya dan kemudian pilih AdministratorAccess. Kebijakan ini memberikan izin penuh untuk semua Layanan AWS.
3. Pilih Selanjutnya.
4. Dalam nama set Izin, masukkan:

   ```
   CodeCatalystECSPermissionSet
   ```

5. Pilih Selanjutnya.
6. Pada halaman Tinjau dan buat, tinjau informasi dan pilih Buat.

Untuk menetapkan izin yang disetel ke CodeCatalyst ECSUSER

1. Di panel navigasi, pilih Akun AWS, lalu pilih kotak centang di Akun AWS samping tempat Anda masuk saat ini.
2. Pilih Tetapkan pengguna atau grup.
3. Pilih tab Pengguna.
4. Pilih kotak centang di sebelah`CodeCatalystECSUser`.
5. Pilih Selanjutnya.
6. Pilih kotak centang di sebelah`CodeCatalystECSPermissionSet`.
7. Pilih Selanjutnya.
8. Tinjau informasi dan pilih Kirim.

   Anda sekarang telah menetapkan `CodeCatalystECSUser` dan `CodeCatalystECSPermissionSet` untuk Anda Akun AWS, mengikat mereka bersama-sama.

Untuk keluar dan masuk kembali sebagai CodeCatalyst ECSUSER

1. Sebelum Anda keluar, pastikan Anda memiliki URL portal AWS akses dan nama pengguna dan kata sandi satu kali untuk`CodeCatalystECSUser`. Anda seharusnya menyalin informasi ini ke editor teks sebelumnya.

   > **Note**
   >
   > Jika Anda tidak memiliki informasi ini, buka halaman `CodeCatalystECSUser` detail di Pusat Identitas IAM, pilih Atur ulang kata sandi, Hasilkan kata sandi satu kali [...] , dan Reset kata sandi lagi untuk menampilkan informasi di layar.

2. Keluar dari AWS.
3. Rekatkan URL portal AWS akses ke bilah alamat browser Anda.
4. Masuk dengan nama pengguna dan kata sandi satu kali untuk`CodeCatalystECSUser`.
5. Di Kata sandi baru, masukkan kata sandi, dan pilih Atur kata sandi baru.

   Sebuah Akun AWSkotak muncul di layar.
6. Pilih Akun AWS, lalu pilih nama yang Akun AWS Anda tetapkan `CodeCatalystECSUser` pengguna dan set izin.
7. Di sebelah`CodeCatalystECSPermissionSet`, pilih Konsol manajemen.

   AWS Management Console Muncul. Anda sekarang masuk `CodeCatalystECSUser` dengan izin yang sesuai.

Untuk meluncurkan sebuah AWS CloudShell instance

1. Seperti`CodeCatalystECSUser`, di bilah navigasi atas, pilih AWS ikon ().

   Halaman utama AWS Management Console muncul.
2. Di bilah navigasi atas, pilih AWS CloudShell ikon ().

   CloudShell terbuka. Tunggu sementara CloudShell lingkungan dibuat.

> **Note**
>
> Jika Anda tidak melihat CloudShell ikon, pastikan Anda berada di Wilayah yang didukung oleh CloudShell. Tutorial ini mengasumsikan Anda berada di Wilayah AS Barat (Oregon).

Untuk memverifikasi bahwa AWS CLI sudah diinstal

1. Di CloudShell terminal, masukkan:

   ```
   aws --version
   ```

2. Periksa apakah ada versi yang muncul.

   Sudah AWS CLI dikonfigurasi untuk pengguna saat ini`CodeCatalystECSUser`, jadi tidak perlu mengkonfigurasi AWS CLI kunci dan kredensyal, seperti biasanya.

Langkah 2: Menyebarkan aplikasi placeholder ke Amazon ECS

Di bagian ini, Anda secara manual menyebarkan aplikasi placeholder ke Amazon ECS. Aplikasi placeholder ini akan digantikan oleh aplikasi Hello World yang digunakan oleh alur kerja Anda. Aplikasi placeholder adalah Apache Web Server.

Untuk informasi selengkapnya tentang Amazon ECS, lihat Panduan Pengembang Layanan Kontainer Elastis Amazon.

Lengkapi serangkaian prosedur berikut untuk menyebarkan aplikasi placeholder.

Untuk membuat peran eksekusi tugas

Peran ini memberikan Amazon ECS dan AWS Fargate (Fargate) izin untuk melakukan panggilan API atas nama Anda.

1. Buat kebijakan kepercayaan:

   a. Di AWS CloudShell, masukkan perintah berikut:

      ```
      cat > codecatalyst-ecs-trust-policy.json
      ```

      Prompt berkedip muncul di CloudShell terminal.

b. Masukkan kode berikut pada prompt:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

c. Tempatkan kursor Anda setelah braket keriting terakhir (). }

d. Tekan **`Enter`** dan kemudian **`Ctrl+d`** untuk menyimpan file dan keluar dari kucing.

2. Buat peran eksekusi tugas:

```
aws iam create-role \
      --role-name codecatalyst-ecs-task-execution-role \
      --assume-role-policy-document file://codecatalyst-ecs-trust-policy.json
```

3. Lampirkan `AmazonECSTaskExecutionRolePolicy` kebijakan AWS terkelola ke peran:

```
aws iam attach-role-policy \
      --role-name codecatalyst-ecs-task-execution-role \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy
```

4. Menampilkan detail peran:

```
aws iam get-role \
      --role-name codecatalyst-ecs-task-execution-role
```

5. Perhatikan `"Arn":` nilai peran, misalnya,`arn:aws:iam::111122223333:role/codecatalyst-ecs-task-execution-role`. Anda akan memerlukan Nama Sumber Daya Amazon (ARN) ini nanti.

Untuk membuat cluster Amazon ECS

Cluster ini akan berisi aplikasi placeholder Apache, dan kemudian, aplikasi Hello World.

1. Seperti`CodeCatalystECSUser`, di AWS CloudShell, buat cluster kosong:

```
aws ecs create-cluster --cluster-name codecatalyst-ecs-cluster
```

2. (Opsional) Verifikasi bahwa cluster berhasil dibuat:

```
aws ecs list-clusters
```

   ARN `codecatalyst-ecs-cluster` cluster akan muncul dalam daftar, menunjukkan penciptaan yang berhasil.

Untuk membuat file definisi tugas

File definisi tugas menunjukkan untuk menjalankan Apache 2.4 Web server Docker image (`httpd:2.4`) yang ditarik dari. DockerHub

1. Seperti`CodeCatalystECSUser`, di AWS CloudShell, buat file definisi tugas:

```
cat > taskdef.json
```

2. Tempel kode berikut pada prompt:

```
{
    "executionRoleArn": "arn:aws:iam::111122223333:role/codecatalyst-ecs-task-
execution-role",
    "containerDefinitions": [
        {
            "name": "codecatalyst-ecs-container",
            "image": "httpd:2.4",
            "essential": true,
            "portMappings": [
                {
                    "hostPort": 80,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ]
```

```
        }
    ],
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "cpu": "256",
    "family": "codecatalyst-ecs-task-def",
    "memory": "512",
    "networkMode": "awsvpc"
}
```

Pada kode sebelumnya, ganti *arn:aws:iam: :111122223333:role/* -role codecatalyst-ecs-task-execution

dengan ARN dari peran eksekusi tugas yang Anda catat. Untuk membuat peran eksekusi tugas

3. Tempatkan kursor Anda setelah braket keriting terakhir (). }
4. Tekan **Enter** dan kemudian **Ctrl+d** untuk menyimpan file dan keluar dari kucing.

Untuk mendaftarkan file definisi tugas dengan Amazon ECS

1. Seperti`CodeCatalystECSUser`, dalam AWS CloudShell, daftarkan definisi tugas:

```
aws ecs register-task-definition \
    --cli-input-json file://taskdef.json
```

2. (Opsional) Verifikasi bahwa definisi tugas telah terdaftar:

```
aws ecs list-task-definitions
```

Definisi `codecatalyst-ecs-task-def` tugas akan muncul dalam daftar.

Untuk membuat layanan Amazon ECS

Layanan Amazon ECS menjalankan tugas (dan wadah Docker terkait) dari aplikasi placeholder Apache, dan kemudian, aplikasi Hello World.

1. Sebagai`CodeCatalystECSUser`, beralihlah ke konsol Amazon Elastic Container Service jika Anda belum melakukannya.
2. Pilih cluster yang Anda buat sebelumnya,`codecatalyst-ecs-cluster`.

3. Di tab Layanan, pilih Buat.
4. Di halaman Create, lakukan hal berikut:

   a. Simpan semua pengaturan default kecuali yang tercantum berikutnya.

   b. Untuk jenis Peluncuran, pilih FARGATE.

   c. Di bawah Definisi tugas, dalam daftar drop-down Keluarga, pilih:

      ```
      codecatalyst-ecs-task-def
      ```

   d. Untuk nama Layanan, masukkan:

      ```
      codecatalyst-ecs-service
      ```

   e. Untuk tugas yang diinginkan, masukkan:

      ```
      3
      ```

      Dalam tutorial ini, setiap tugas meluncurkan satu wadah Docker.

   f. Perluas bagian Jaringan.

   g. Untuk VPC, pilih VPC apa saja.

   h. Untuk Subnet, pilih subnet apa saja.

      > **Note**
      >
      > Hanya tentukan satu subnet. Itu saja yang diperlukan untuk tutorial ini.

      > **Note**
      >
      > Jika Anda tidak memiliki VPC dan subnet, buatlah. Lihat Membuat VPC, dan Membuat subnet di VPC Anda di Panduan Pengguna Amazon VPC.

   i. Untuk grup Keamanan, pilih Buat grup keamanan baru, lalu lakukan hal berikut:

      i. Untuk nama grup Keamanan, masukkan:

         ```
         codecatalyst-ecs-security-group
         ```

      ii. Untuk deskripsi grup Keamanan, masukkan:

```
CodeCatalyst ECS security group
```

iii. Pilih Tambahkan aturan. Untuk Type, pilih HTTP, dan untuk Source, pilih Anywhere.

j. Di bagian bawah, pilih Buat.

k. Tunggu sementara layanan dibuat. Ini mungkin memakan waktu beberapa menit.

5. Pilih tab Tugas, lalu pilih tombol segarkan. Verifikasi bahwa ketiga tugas tersebut memiliki kolom Status Terakhir yang disetel ke Running.

(Opsional) Untuk memverifikasi bahwa aplikasi placeholder Apache Anda sedang berjalan

1. Di tab Tugas, pilih salah satu dari tiga tugas.
2. Di bidang IP Publik, pilih alamat terbuka.

   Sebuah `It Works!` halaman muncul. Ini menunjukkan bahwa layanan Amazon ECS berhasil memulai tugas yang meluncurkan wadah Docker dengan gambar Apache.

   Pada titik ini dalam tutorial, Anda telah secara manual menerapkan klaster Amazon ECS, layanan, dan definisi tugas, serta aplikasi placeholder Apache. Dengan semua item ini di tempat, Anda sekarang siap untuk membuat alur kerja yang akan menggantikan aplikasi placeholder Apache dengan aplikasi Hello World tutorial.

Langkah 3: Buat repositori gambar Amazon ECR

Di bagian ini, Anda membuat repositori gambar pribadi di Amazon Elastic Container Registry (Amazon ECR) Registry ECR). Repositori ini menyimpan gambar Docker tutorial yang akan menggantikan gambar placeholder Apache yang Anda gunakan sebelumnya.

Untuk informasi selengkapnya tentang Amazon ECR, lihat Panduan Pengguna Amazon Elastic Container Registry.

Untuk membuat repositori gambar di Amazon ECR

1. Seperti`CodeCatalystECSUser`, di AWS CloudShell, buat repositori kosong di Amazon ECR:

```
aws ecr create-repository --repository-name codecatalyst-ecs-image-repo
```

2. Tampilkan detail repositori Amazon ECR:

```
aws ecr describe-repositories \
      --repository-names codecatalyst-ecs-image-repo
```

3. Perhatikan "`repositoryUri`": nilainya, misalnya,`111122223333.dkr.ecr.us-west-2.amazonaws.com/codecatalyst-ecs-image-repo`.

   Anda membutuhkannya nanti saat menambahkan repositori ke alur kerja Anda.

Langkah 4: Buat AWS peran

Di bagian ini, Anda membuat peran AWS IAM yang dibutuhkan CodeCatalyst alur kerja Anda agar berfungsi. Peran ini adalah:

- Peran build - Memberikan izin tindakan CodeCatalyst build (dalam alur kerja) untuk mengakses AWS akun Anda dan menulis ke Amazon ECR dan Amazon EC2.
- Menyebarkan peran - Memberikan izin tindakan CodeCatalyst Deploy to ECS (dalam alur kerja) untuk mengakses akun Anda AWS , Amazon ECS, dan beberapa layanan lainnya. AWS

Untuk informasi selengkapnya tentang peran IAM, lihat peran IAM di AWS Identity and Access Management Panduan Pengguna.

> **Note**
>
> Untuk menghemat waktu, Anda dapat membuat satu peran, yang disebut `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran, alih-alih dua peran yang tercantum sebelumnya. Untuk informasi selengkapnya, lihat Membuat CodeCatalystWorkflowDevelopmentRole-***`spaceName`***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami menyarankan Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian. Tutorial ini mengasumsikan Anda membuat dua peran yang tercantum sebelumnya.

Untuk membuat peran build dan deploy, Anda dapat menggunakan peran AWS Management Console atau. AWS CLI

AWS Management Console

Untuk membuat peran build dan deploy, selesaikan rangkaian prosedur berikut.

Untuk membuat peran build

1. Buat kebijakan untuk peran tersebut, sebagai berikut:

   a. Masuk ke AWS.
   b. Buka konsol IAM di https://console.aws.amazon.com/iam/.
   c. Di panel navigasi, pilih Kebijakan.
   d. Pilih Buat kebijakan.
   e. Pilih tab JSON.
   f. Hapus kode yang ada.
   g. Tempel kode berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:*",
                "ec2:*"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

   h. Pilih Berikutnya: Tanda.

i. Pilih Berikutnya: Tinjau.

j. Dalam Nama, masukkan:

```
codecatalyst-ecs-build-policy
```

k. Pilih Buat kebijakan.

Anda sekarang telah membuat kebijakan izin.

2. Buat peran build, sebagai berikut:

a. Di panel navigasi, pilih Peran, lalu pilih Buat peran.

b. Pilih Kebijakan kepercayaan khusus.

c. Hapus kebijakan kepercayaan kustom yang ada.

d. Tambahkan kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                 ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

e. Pilih Selanjutnya.

f. Di Kebijakan izin, cari`codecatalyst-ecs-build-policy`, pilih kotak centang.

g. Pilih Selanjutnya.

h. Untuk nama Peran, masukkan:

```
codecatalyst-ecs-build-role
```

i. Untuk deskripsi Peran, masukkan:

```
CodeCatalyst ECS build role
```

j. Pilih Buat peran.

Anda sekarang telah membuat peran build dengan kebijakan izin dan kebijakan kepercayaan.

3. Dapatkan peran membangun ARN, sebagai berikut:

a. Di panel navigasi, pilih Peran.

b. Di kotak pencarian, masukkan nama peran yang baru saja Anda buat (`codecatalyst-ecs-build-role`).

c. Pilih peran dari daftar.

Halaman Ringkasan peran muncul.

d. Di bagian atas, salin nilai ARN. Anda membutuhkannya nanti.

Untuk membuat peran penerapan

1. Buat kebijakan untuk peran tersebut, sebagai berikut:

a. Masuk ke AWS.

b. Buka konsol IAM di https://console.aws.amazon.com/iam/.

c. Di panel navigasi, pilih Kebijakan.

d. Pilih Buat Kebijakan.

e. Pilih tab JSON.

f. Hapus kode yang ada.

g. Tempel kode berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [{
    "Action":[
      "ecs:DescribeServices",
      "ecs:CreateTaskSet",
      "ecs:DeleteTaskSet",
      "ecs:ListClusters",
```

```
      "ecs:RegisterTaskDefinition",
      "ecs:UpdateServicePrimaryTaskSet",
      "ecs:UpdateService",
      "elasticloadbalancing:DescribeTargetGroups",
      "elasticloadbalancing:DescribeListeners",
      "elasticloadbalancing:ModifyListener",
      "elasticloadbalancing:DescribeRules",
      "elasticloadbalancing:ModifyRule",
      "lambda:InvokeFunction",
      "lambda:ListFunctions",
      "cloudwatch:DescribeAlarms",
      "sns:Publish",
      "sns:ListTopics",
      "s3:GetObject",
      "s3:GetObjectVersion",
      "codedeploy:CreateApplication",
      "codedeploy:CreateDeployment",
      "codedeploy:CreateDeploymentGroup",
      "codedeploy:GetApplication",
      "codedeploy:GetDeployment",
      "codedeploy:GetDeploymentGroup",
      "codedeploy:ListApplications",
      "codedeploy:ListDeploymentGroups",
      "codedeploy:ListDeployments",
      "codedeploy:StopDeployment",
      "codedeploy:GetDeploymentTarget",
      "codedeploy:ListDeploymentTargets",
      "codedeploy:GetDeploymentConfig",
      "codedeploy:GetApplicationRevision",
      "codedeploy:RegisterApplicationRevision",
      "codedeploy:BatchGetApplicationRevisions",
      "codedeploy:BatchGetDeploymentGroups",
      "codedeploy:BatchGetDeployments",
      "codedeploy:BatchGetApplications",
      "codedeploy:ListApplicationRevisions",
      "codedeploy:ListDeploymentConfigs",
      "codedeploy:ContinueDeployment"
   ],
   "Resource":"*",
   "Effect":"Allow"
},{"Action":[
      "iam:PassRole"
   ],
   "Effect":"Allow",
```

```
   "Resource":"*",
   "Condition":{"StringLike":{"iam:PassedToService":[
            "ecs-tasks.amazonaws.com",
            "codedeploy.amazonaws.com"
         ]
      }
   }
}]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya. Anda kemudian dapat mencatat kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

h. Pilih Berikutnya: Tanda.

i. Pilih Berikutnya: Tinjau.

j. Dalam Nama, masukkan:

```
codecatalyst-ecs-deploy-policy
```

k. Pilih Buat kebijakan.

Anda sekarang telah membuat kebijakan izin.

2. Buat peran deploy, sebagai berikut:

a. Di panel navigasi, pilih Peran, lalu pilih Buat peran.

b. Pilih Kebijakan kepercayaan khusus.

c. Hapus kebijakan kepercayaan kustom yang ada.

d. Tambahkan kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                 ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

e. Pilih Selanjutnya.

f. Di Kebijakan izin, cari`codecatalyst-ecs-deploy-policy`, pilih kotak centang.

g. Pilih Selanjutnya.

h. Untuk nama Peran, masukkan:

```
codecatalyst-ecs-deploy-role
```

i. Untuk deskripsi Peran, masukkan:

```
CodeCatalyst ECS deploy role
```

j. Pilih Buat peran.

Anda sekarang telah membuat peran penerapan dengan kebijakan kepercayaan.

3. Dapatkan peran penyebaran ARN, sebagai berikut:

a. Di panel navigasi, pilih Peran.

b. Di kotak pencarian, masukkan nama peran yang baru saja Anda buat (`codecatalyst-ecs-deploy-role`).

c. Pilih peran dari daftar.

Halaman Ringkasan peran muncul.

d. Di bagian atas, salin nilai ARN. Anda membutuhkannya nanti.

AWS CLI

Untuk membuat peran build dan deploy, selesaikan rangkaian prosedur berikut.

Untuk membuat kebijakan kepercayaan untuk kedua peran

Seperti`CodeCatalystECSUser`, di AWS CloudShell, buat file kebijakan kepercayaan:

1. Buat file:

```
cat > codecatalyst-ecs-trust-policy.json
```

2. Pada prompt terminal, tempel kode berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                 ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

3. Tempatkan kursor Anda setelah braket keriting terakhir (). }
4. Tekan **`Enter`** dan kemudian **`Ctrl+d`** untuk menyimpan file dan keluar dari kucing.

Untuk membuat kebijakan membangun dan membangun peran

1. Buat kebijakan build:

   a. Seperti`CodeCatalystECSUser`, dalam AWS CloudShell, membuat file kebijakan build:

```
cat > codecatalyst-ecs-build-policy.json
```

b. Pada prompt, masukkan kode berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:*",
                "ec2:*"
            ],
            "Resource": "*"
        }
    ]
}
```

c. Tempatkan kursor Anda setelah braket keriting terakhir (). }

d. Tekan **`Enter`** dan kemudian **`Ctrl+d`** untuk menyimpan file dan keluar dari kucing.

2. Tambahkan kebijakan build ke AWS:

```
aws iam create-policy \
    --policy-name codecatalyst-ecs-build-policy \
    --policy-document file://codecatalyst-ecs-build-policy.json
```

3. Dalam output perintah, perhatikan `"arn":` nilainya, misalnya,`arn:aws:iam::111122223333:policy/codecatalyst-ecs-build-policy`. Anda membutuhkan ARN ini nanti.

4. Buat peran build dan lampirkan kebijakan kepercayaan padanya:

```
aws iam create-role \
      --role-name codecatalyst-ecs-build-role \
      --assume-role-policy-document file://codecatalyst-ecs-trust-policy.json
```

5. Lampirkan kebijakan build ke peran build:

```
aws iam attach-role-policy \
      --role-name codecatalyst-ecs-build-role \
      --policy-arn arn:aws:iam::111122223333:policy/codecatalyst-ecs-build-policy
```

Di mana *arn:aws:iam: codecatalyst-ecs-build-policy :111122223333:policy/* diganti dengan ARN dari kebijakan build yang Anda sebutkan sebelumnya.

6. Menampilkan detail peran build:

```
aws iam get-role \
      --role-name codecatalyst-ecs-build-role
```

7. Perhatikan `"Arn":` nilai peran, misalnya,`arn:aws:iam::111122223333:role/codecatalyst-ecs-build-role`. Anda membutuhkan ARN ini nanti.

Untuk membuat kebijakan penerapan dan peran penerapan

1. Buat kebijakan penerapan:

   a. Di AWS CloudShell, buat file kebijakan penerapan:

   ```
   cat > codecatalyst-ecs-deploy-policy.json
   ```

   b. Pada prompt, masukkan kode berikut:

   ```
   {
       "Version": "2012-10-17",
       "Statement": [{
       "Action":[
         "ecs:DescribeServices",
         "ecs:CreateTaskSet",
         "ecs:DeleteTaskSet",
         "ecs:ListClusters",
         "ecs:RegisterTaskDefinition",
         "ecs:UpdateServicePrimaryTaskSet",
         "ecs:UpdateService",
         "elasticloadbalancing:DescribeTargetGroups",
         "elasticloadbalancing:DescribeListeners",
         "elasticloadbalancing:ModifyListener",
         "elasticloadbalancing:DescribeRules",
         "elasticloadbalancing:ModifyRule",
         "lambda:InvokeFunction",
         "lambda:ListFunctions",
         "cloudwatch:DescribeAlarms",
   ```

```
      "sns:Publish",
      "sns:ListTopics",
      "s3:GetObject",
      "s3:GetObjectVersion",
      "codedeploy:CreateApplication",
      "codedeploy:CreateDeployment",
      "codedeploy:CreateDeploymentGroup",
      "codedeploy:GetApplication",
      "codedeploy:GetDeployment",
      "codedeploy:GetDeploymentGroup",
      "codedeploy:ListApplications",
      "codedeploy:ListDeploymentGroups",
      "codedeploy:ListDeployments",
      "codedeploy:StopDeployment",
      "codedeploy:GetDeploymentTarget",
      "codedeploy:ListDeploymentTargets",
      "codedeploy:GetDeploymentConfig",
      "codedeploy:GetApplicationRevision",
      "codedeploy:RegisterApplicationRevision",
      "codedeploy:BatchGetApplicationRevisions",
      "codedeploy:BatchGetDeploymentGroups",
      "codedeploy:BatchGetDeployments",
      "codedeploy:BatchGetApplications",
      "codedeploy:ListApplicationRevisions",
      "codedeploy:ListDeploymentConfigs",
      "codedeploy:ContinueDeployment"
   ],
   "Resource":"*",
   "Effect":"Allow"
},{"Action":[
      "iam:PassRole"
   ],
   "Effect":"Allow",
   "Resource":"*",
   "Condition":{"StringLike":{"iam:PassedToService":[
            "ecs-tasks.amazonaws.com",
            "codedeploy.amazonaws.com"
         ]
      }
   }
}]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

c. Tempatkan kursor Anda setelah braket keriting terakhir (). }

d. Tekan **Enter** dan kemudian **Ctrl+d** untuk menyimpan file dan keluar dari kucing.

2. Tambahkan kebijakan penerapan ke AWS:

```
aws iam create-policy \
    --policy-name codecatalyst-ecs-deploy-policy \
    --policy-document file://codecatalyst-ecs-deploy-policy.json
```

3. Dalam output perintah, perhatikan `"arn":` nilai kebijakan penerapan, misalnya,`arn:aws:iam::111122223333:policy/codecatalyst-ecs-deploy-policy`. Anda membutuhkan ARN ini nanti.

4. Buat peran penerapan dan lampirkan kebijakan kepercayaan padanya:

```
aws iam create-role \
      --role-name codecatalyst-ecs-deploy-role \
      --assume-role-policy-document file://codecatalyst-ecs-trust-policy.json
```

5. Lampirkan kebijakan penerapan ke peran penerapan, di mana *arn:aws:iam: :111122223333:policy/* diganti dengan ARN dari kebijakan penerapan yang Anda sebutkan sebelumnya. codecatalyst-ecs-deploy-policy

```
aws iam attach-role-policy \
      --role-name codecatalyst-ecs-deploy-role \
      --policy-arn arn:aws:iam::111122223333:policy/codecatalyst-ecs-deploy-policy
```

6. Menampilkan detail peran penerapan:

```
aws iam get-role \
```

```
    --role-name codecatalyst-ecs-deploy-role
```

7. Perhatikan `"Arn":` nilai peran, misalnya,`arn:aws:iam::111122223333:role/codecatalyst-ecs-deploy-role`. Anda membutuhkan ARN ini nanti.

Langkah 5: Tambahkan AWS peran ke CodeCatalyst

Pada langkah ini, Anda menambahkan build role (`codecatalyst-ecs-build-role`) dan deploy role (`codecatalyst-ecs-deploy-role`) ke koneksi CodeCatalyst akun di ruang Anda.

Untuk menambahkan peran build dan deploy ke koneksi akun Anda

1. Masuk CodeCatalyst, navigasikan ke ruang Anda.
2. Pilih AWS akun. Daftar koneksi akun muncul.
3. Pilih koneksi akun yang mewakili AWS akun tempat Anda membuat peran build dan deploy.
4. Pilih Kelola peran dari konsol AWS manajemen.

   Peran Tambahkan IAM ke halaman CodeCatalyst ruang Amazon muncul. Anda mungkin perlu masuk untuk mengakses halaman.
5. Pilih Tambahkan peran yang sudah Anda buat di IAM.

   Daftar drop-down muncul. Daftar ini menampilkan semua peran IAM dengan kebijakan kepercayaan yang mencakup prinsip `codecatalyst-runner.amazonaws.com` dan `codecatalyst.amazonaws.com` layanan.
6. Dalam daftar drop-down, pilih`codecatalyst-ecs-build-role`, dan pilih Tambah peran.

   > **Note**
   >
   > Jika Anda melihat`The security token included in the request is invalid`, itu mungkin karena Anda tidak memiliki izin yang tepat. Untuk memperbaiki masalah ini, keluar dari AWS sebagai masuk kembali dengan AWS akun yang Anda gunakan saat membuat CodeCatalyst ruang.
7. Pilih Tambahkan peran IAM, pilih Tambahkan peran yang ada yang telah Anda buat di IAM, dan dalam daftar drop-down, pilih. `codecatalyst-ecs-deploy-role` Pilih Tambahkan peran.

   Anda sekarang telah menambahkan peran build dan deploy ke ruang Anda.

8. Salin nilai nama CodeCatalyst tampilan Amazon. Anda akan membutuhkan nilai ini nanti, saat membuat alur kerja Anda.

Langkah 6: Buat repositori sumber

Pada langkah ini, Anda membuat repositori sumber di. CodeCatalyst Repositori ini menyimpan file sumber tutorial, seperti file definisi tugas.

Untuk informasi selengkapnya tentang repositori sumber, lihat. Membuat repositori sumber

Untuk membuat repositori sumber

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda,`codecatalyst-ecs-project`.
3. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
4. Pilih Tambahkan repositori, lalu pilih Buat repositori.
5. Dalam nama Repositori, masukkan:

   ```
   codecatalyst-ecs-source-repository
   ```

6. Pilih Buat.

Langkah 7: Tambahkan file sumber

Di bagian ini, Anda menambahkan file sumber Hello World ke CodeCatalyst repositori Anda,. `codecatalyst-ecs-source-repository` Mereka terdiri dari:

- `index.html`File - Menampilkan pesan Hello World di browser.
- A Dockerfile - Menjelaskan gambar dasar yang akan digunakan untuk image Docker Anda dan perintah Docker untuk menerapkannya.
- `taskdef.json`File - Mendefinisikan image Docker yang akan digunakan saat meluncurkan tugas ke cluster Anda.

Struktur folder adalah sebagai berikut:

```
.
|— public-html
|   |— index.html
```

```
|— Dockerfile
|— taskdef.json
```

> **Note**
>
> Petunjuk berikut menunjukkan cara menambahkan file menggunakan CodeCatalyst konsol tetapi Anda dapat menggunakan Git jika Anda mau. Lihat perinciannya di Mengkloning repositori sumber.

Topik

- index.html
- Dockerfile
- taskdef.json

index.html

`index.html`File menampilkan pesan Hello World di browser.

Untuk menambahkan file index.html

1. Di CodeCatalyst konsol, buka repositori sumber Anda,. `codecatalyst-ecs-source-repository`
2. Di File, pilih Buat file.
3. Untuk nama File, masukkan:

   ```
   public-html/index.html
   ```

   > **Important**
   >
   > Pastikan untuk menyertakan `public-html/` awalan untuk membuat folder dengan nama yang sama. `index.html`Diharapkan ada di folder ini.

4. Di kotak teks, masukkan kode berikut:

   ```
   <html>
     <head>
   ```

```
    <title>Hello World</title>
    <style>
      body {
      background-color: black;
      text-align: center;
      color: white;
      font-family: Arial, Helvetica, sans-serif;
      }
    </style>
  </head>
  <body>
    <h1>Hello World</h1>
  </body>
</html>
```

5. Pilih Komit, lalu pilih Komit lagi.

   Ditambahkan ke repositori Anda dalam folder. `index.html` `public-html`

Dockerfile

Dockerfile menjelaskan image Docker dasar yang akan digunakan dan perintah Docker untuk diterapkan padanya. Untuk informasi selengkapnya tentang Dockerfile, lihat Referensi Dockerfile.

Dockerfile yang ditentukan di sini menunjukkan untuk menggunakan image dasar Apache 2.4 (). `httpd` Ini juga mencakup instruksi untuk menyalin file sumber yang dipanggil `index.html` ke folder di server Apache yang melayani halaman web. `EXPOSE`Instruksi di Dockerfile memberi tahu Docker bahwa wadah mendengarkan pada port 80.

Untuk menambahkan Dockerfile

1. Di repositori sumber Anda, pilih Buat file.
2. Untuk nama File, masukkan:

   ```
   Dockerfile
   ```

   Jangan sertakan ekstensi file.

> **Important**
>
> Dockerfile harus berada di folder root repositori Anda. `Docker build`Perintah alur kerja mengharapkannya ada di sana.

3. Di kotak teks, masukkan kode berikut:

```
FROM httpd:2.4
COPY ./public-html/index.html /usr/local/apache2/htdocs/index.html
EXPOSE 80
```

4. Pilih Komit, lalu pilih Komit lagi.

   Dockerfile ditambahkan ke repositori Anda.

taskdef.json

`taskdef.json`File yang Anda tambahkan pada langkah ini sama dengan yang sudah Anda tentukan Langkah 2: Menyebarkan aplikasi placeholder ke Amazon ECS dengan perbedaan berikut:

Alih-alih menentukan nama gambar Docker yang di-hardcode di `image:` bidang (`httpd:2.4`), definisi tugas di sini menggunakan beberapa variabel untuk menunjukkan gambar: dan. `$REPOSITORY_URI` `$IMAGE_TAG` Variabel ini akan diganti dengan nilai nyata yang dihasilkan oleh aksi build alur kerja saat Anda menjalankan alur kerja di langkah selanjutnya.

Untuk detail tentang parameter definisi tugas, lihat Parameter definisi tugas di Panduan Pengembang Layanan Kontainer Elastis Amazon.

Untuk menambahkan file taskdef.json

1. Di repositori sumber Anda, pilih Buat file.
2. Untuk nama File, masukkan:

```
taskdef.json
```

3. Di kotak teks, masukkan kode berikut:

```
{
    "executionRoleArn": "arn:aws:iam::account_ID:role/codecatalyst-ecs-task-
execution-role",
```

```
    "containerDefinitions": [
        {
            "name": "codecatalyst-ecs-container",
            # The $REPOSITORY_URI and $IMAGE_TAG variables will be replaced
            # by the workflow at build time (see the build action in the
            # workflow)
            "image": $REPOSITORY_URI:$IMAGE_TAG,
            "essential": true,
            "portMappings": [
                {
                    "hostPort": 80,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ]
        }
    ],
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "networkMode": "awsvpc",
    "cpu": "256",
    "memory": "512",
    "family": "codecatalyst-ecs-task-def"
}
```

Pada kode sebelumnya, ganti

arn:aws:iam: :account_id:peran/ -peran codecatalyst-ecs-task-execution

dengan ARN dari peran eksekusi tugas yang Anda catat. Untuk membuat peran eksekusi tugas

4. Pilih Komit, lalu pilih Komit lagi.

   `taskdef.json`File ditambahkan ke repositori Anda.

Langkah 8: Buat dan jalankan alur kerja

Pada langkah ini, Anda membuat alur kerja yang mengambil file sumber Anda, membuatnya menjadi image Docker, dan kemudian menyebarkan gambar ke cluster Amazon ECS Anda. Deployment ini menggantikan aplikasi placeholder Apache yang ada.

Alur kerja terdiri dari blok bangunan berikut yang berjalan secara berurutan:

- Pemicu - Pemicu ini memulai alur kerja yang dijalankan secara otomatis saat Anda mendorong perubahan ke repositori sumber Anda. Untuk informasi lebih lanjut tentang menggunakan pemicu, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.
- Tindakan build (`BuildBackend`) — Pada pemicu, aksi membangun image Docker menggunakan Dockerfile dan mendorong gambar ke Amazon ECR. Tindakan build juga memperbarui `taskdef.json` dengan nilai `image` bidang yang benar, dan kemudian membuat artefak keluaran file ini. Artefak ini digunakan sebagai masukan untuk tindakan penyebaran, yang berikutnya.

  Untuk informasi selengkapnya tentang tindakan build, lihatMembangun dengan alur kerja.
- Tindakan penerapan (`DeployToECS`) — Setelah menyelesaikan aksi build, tindakan deploy mencari artefak keluaran yang dihasilkan oleh build action (`TaskDefArtifact`), menemukan `taskdef.json` bagian dalamnya, dan mendaftarkannya dengan layanan Amazon ECS Anda. Layanan kemudian mengikuti instruksi dalam `taskdef.json` file untuk menjalankan tiga tugas Amazon ECS — dan container Hello World Docker terkait — di dalam cluster Amazon ECS Anda.

Untuk membuat alur kerja

1. Di CodeCatalyst konsol, di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih Buat alur kerja.
3. Untuk repositori Sumber, pilih. `codecatalyst-ecs-source-repository`
4. Untuk Cabang, pilih`main`.
5. Pilih Buat.
6. Hapus kode sampel YAMAL.
7. Tambahkan kode YAML berikut:

```
Name: codecatalyst-ecs-workflow
SchemaVersion: 1.0

Triggers:
  - Type: PUSH
    Branches:
      - main
Actions:
  BuildBackend:
    Identifier: aws/build@v1
    Environment:
      Name: codecatalyst-ecs-environment
```

```
      Connections:
        - Name: codecatalyst-account-connection
          Role: codecatalyst-ecs-build-role
    Inputs:
      Sources:
        - WorkflowSource
      Variables:
        - Name: REPOSITORY_URI
          Value: 111122223333.dkr.ecr.us-west-2.amazonaws.com/codecatalyst-ecs-
image-repo
        - Name: IMAGE_TAG
          Value: ${WorkflowSource.CommitId}
    Configuration:
      Steps:
        #pre_build:
        - Run: echo Logging in to Amazon ECR...
        - Run: aws --version
        - Run: aws ecr get-login-password --region us-west-2 | docker login --
username AWS --password-stdin 111122223333.dkr.ecr.us-west-2.amazonaws.com
        #build:
        - Run: echo Build started on `date`
        - Run: echo Building the Docker image...
        - Run: docker build -t $REPOSITORY_URI:latest .
        - Run: docker tag $REPOSITORY_URI:latest $REPOSITORY_URI:$IMAGE_TAG
        #post_build:
        - Run: echo Build completed on `date`
        - Run: echo Pushing the Docker images...
        - Run: docker push $REPOSITORY_URI:latest
        - Run: docker push $REPOSITORY_URI:$IMAGE_TAG
        # Replace the variables in taskdef.json
        - Run: find taskdef.json -type f | xargs sed -i "s|\$REPOSITORY_URI|
$REPOSITORY_URI|g"
        - Run: find taskdef.json -type f | xargs sed -i "s|\$IMAGE_TAG|$IMAGE_TAG|
g"
        - Run: cat taskdef.json
        # The output artifact will be a zip file that contains a task definition
 file.
    Outputs:
      Artifacts:
        - Name: TaskDefArtifact
          Files:
            - taskdef.json
  DeployToECS:
    DependsOn:
```

```
      - BuildBackend
    Identifier: aws/ecs-deploy@v1
    Environment:
      Name: codecatalyst-ecs-environment
      Connections:
        - Name: codecatalyst-account-connection
          Role: codecatalyst-ecs-deploy-role
    Inputs:
      Sources: []
      Artifacts:
        - TaskDefArtifact
    Configuration:
      region: us-west-2
      cluster: codecatalyst-ecs-cluster
      service: codecatalyst-ecs-service
      task-definition: taskdef.json
```

Pada kode sebelumnya, ganti:

- Kedua contoh *codecatalyst-ecs-environment*dengan nama lingkungan yang Anda buat. Prasyarat
- Kedua contoh *codecatalyst-account-connection*dengan nama tampilan koneksi akun Anda. Nama tampilan mungkin nomor. Untuk informasi selengkapnya, lihat Langkah 5: Tambahkan AWS peran ke CodeCatalyst.
- *codecatalyst-ecs-build-role*dengan nama peran build yang Anda buatLangkah 4: Buat AWS peran.
- *111122223333.dkr. ecr.us-west-2.amazonaws.com/codecatalyst-ecs-image-repo*(di propertiValue:) dengan URI repositori Amazon ECR yang Anda buat. Langkah 3: Buat repositori gambar Amazon ECR
- *111122223333.dkr. ecr.us-west-2.amazonaws.com* (dalam perintahRun: aws ecr) dengan URI repositori Amazon ECR tanpa akhiran gambar (). /codecatalyst-ecs-image-repo
- *codecatalyst-ecs-deploy-role*dengan nama peran penerapan yang Anda buat. Langkah 4: Buat AWS peran
- Kedua instance *us-west-2* dengan kode Wilayah Anda. AWS Untuk daftar kode Region, lihat Titik akhir Regional di. Referensi Umum AWS

> **Note**
>
> Jika Anda memutuskan untuk tidak membuat peran build dan deploy, ganti *codecatalyst-ecs-build-role*dan *codecatalyst-ecs-deploy-role*dengan nama `CodeCatalystWorkflowDevelopmentRole-`*spaceName* peran tersebut. Untuk informasi selengkapnya tentang peran ini, silakan lihat Langkah 4: Buat AWS peran.

> **Tip**
>
> Alih-alih menggunakan `sed` perintah `find` dan yang ditampilkan dalam kode alur kerja sebelumnya untuk memperbarui repositori dan nama gambar, Anda dapat menggunakan tindakan definisi tugas Render Amazon ECS untuk tujuan ini. Untuk informasi selengkapnya, lihat Memodifikasi file definisi tugas Amazon ECS menggunakan alur kerja.

8. (Opsional) Pilih Validasi untuk memastikan bahwa kode YAMAL valid sebelum melakukan.
9. Pilih Terapkan.
10. Dalam kotak dialog Commit workflow, masukkan yang berikut ini:

    a. Untuk pesan Commit, hapus teks dan masukkan:

    ```
    Add first workflow
    ```

    b. Untuk Repositori, pilih. `codecatalyst-ecs-source-repository`

    c. Untuk nama Branch, pilih main.

    d. Pilih Terapkan.

Anda sekarang telah membuat alur kerja. Jalankan alur kerja dimulai secara otomatis karena pemicu yang ditentukan di bagian atas alur kerja. Khususnya, ketika Anda melakukan (dan mendorong) `workflow.yaml` file ke repositori sumber Anda, pemicu memulai alur kerja dijalankan.

Untuk melihat alur kerja, jalankan kemajuan

1. Di panel navigasi CodeCatalyst konsol, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih alur kerja yang baru saja Anda buat,`codecatalyst-ecs-workflow`.
3. Pilih BuildBackenduntuk melihat kemajuan pembangunan.
4. Pilih DeployToECS untuk melihat kemajuan penerapan.

   Untuk informasi selengkapnya tentang melihat detail run, lihatMelihat alur kerja menjalankan status dan detail.

Untuk memverifikasi penyebaran

1. Buka konsol klasik Amazon ECS di https://console.aws.amazon.com/ecs/.
2. Pilih cluster Anda,`codecatalyst-ecs-cluster`.
3. Pilih tab Tugas.
4. Pilih salah satu dari tiga tugas.
5. Di bidang IP Publik, pilih alamat terbuka.

   Halaman “Hello World” muncul di browser, menunjukkan bahwa layanan Amazon ECS berhasil menyebarkan aplikasi Anda.

Langkah 9: Buat perubahan pada file sumber Anda

Di bagian ini, Anda membuat perubahan pada `index.html` file di repositori sumber Anda. Perubahan ini menyebabkan alur kerja membuat image Docker baru, menandainya dengan ID komit, mendorongnya ke Amazon ECR, dan menerapkannya ke Amazon ECS.

Untuk mengubah index.html

1. Di CodeCatalyst konsol, di panel navigasi, pilih Kode, lalu pilih repositori Sumber, lalu pilih repositori Anda,. `codecatalyst-ecs-source-repository`
2. Pilih`public-html`, lalu pilih`index.html`.

   Isi `index.html` muncul.
3. Pilih Edit.
4. Pada baris 14, ubah `Hello World` teks menjadi`Tutorial complete!`.

5. Pilih Komit, lalu pilih Komit lagi.

   Komit menyebabkan alur kerja baru dijalankan.
6. (Opsional) Buka halaman utama repositori sumber Anda, pilih Lihat komit, lalu catat ID komit untuk perubahan tersebut. `index.html`
7. Perhatikan kemajuan penerapan:

   a. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.

   b. Pilih `codecatalyst-ecs-workflow` untuk melihat proses terbaru.

   c. Pilih BuildBackend, dan DeployToECS untuk melihat alur kerja berjalan kemajuan.
8. Verifikasi bahwa aplikasi Anda telah diperbarui, sebagai berikut:

   a. Buka konsol klasik Amazon ECS di https://console.aws.amazon.com/ecs/.

   b. Pilih cluster Anda,`codecatalyst-ecs-cluster`.

   c. Pilih tab Tugas.

   d. Pilih salah satu dari tiga tugas.

   e. Di bidang IP Publik, pilih alamat terbuka.

      Sebuah `Tutorial complete!` halaman muncul.
9. (Opsional) Di AWS, alihkan ke konsol Amazon ECR dan verifikasi bahwa image Docker baru ditandai dengan ID komit dari langkah 6.

Bersihkan

Bersihkan file dan layanan yang digunakan dalam tutorial ini untuk menghindari biaya untuk mereka.

Dalam AWS Management Console, bersihkan dalam urutan ini:

1. Di Amazon ECS, lakukan hal berikut:
   a. Hapus`codecatalyst-ecs-service`.
   b. Hapus`codecatalyst-ecs-cluster`.
   c. Deregister`codecatalyst-ecs-task-definition`.
2. Di Amazon ECR, hapus`codecatalyst-ecs-image-repo`.
3. Di Amazon EC2, hapus. `codecatalyst-ecs-security-group`
4. Di Pusat Identitas IAM, hapus:

a. `CodeCatalystECSUser`

b. `CodeCatalystECSPermissionSet`

Di CodeCatalyst konsol, bersihkan sebagai berikut:

1. Hapus`codecatalyst-ecs-workflow`.
2. Hapus`codecatalyst-ecs-environment`.
3. Hapus`codecatalyst-ecs-source-repository`.
4. Hapus`codecatalyst-ecs-project`.

Dalam tutorial ini, Anda mempelajari cara menerapkan aplikasi ke layanan Amazon ECS menggunakan CodeCatalyst alur kerja dan tindakan Deploy ke Amazon ECS.

Menambahkan tindakan “Terapkan ke Amazon ECS”

Gunakan petunjuk berikut untuk menambahkan tindakan Deploy ke Amazon ECS ke alur kerja Anda.

Visual

Untuk menambahkan tindakan “Terapkan ke Amazon ECS” menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan Deploy to Amazon ECS, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

- Pilih Terapkan ke Amazon ECS. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
  - (Opsional) Pilih Unduh untuk melihat kode sumber tindakan.
  - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

10. Pada tab Input dan Konfigurasi, lengkapi bidang sesuai dengan kebutuhan Anda. Untuk deskripsi setiap bidang, lihatTindakan “Terapkan ke Amazon ECS” definisi YAMAL. Referensi ini memberikan informasi rinci tentang setiap bidang (dan nilai properti YAMAL yang sesuai) seperti yang muncul di YAMAL dan editor visual.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menambahkan tindakan “Deploy to Amazon ECS” menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan Deploy to Amazon ECS, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih Terapkan ke Amazon ECS. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Unduh untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

10. Ubah properti dalam kode YAMAL sesuai dengan kebutuhan Anda. Penjelasan tentang setiap properti yang tersedia disediakan diTindakan “Terapkan ke Amazon ECS” definisi YAMAL.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Variabel yang dihasilkan oleh tindakan “Terapkan ke Amazon ECS”

Tindakan Deploy to Amazon ECS menghasilkan dan menetapkan variabel berikut pada waktu berjalan. Ini dikenal sebagai variabel yang telah ditentukan.

Untuk informasi tentang mereferensikan variabel-variabel ini dalam alur kerja, lihat. Menggunakan variabel yang telah ditentukan

Kunci	Nilai
cluster	Nama cluster Amazon ECS yang digunakan selama alur kerja dijalankan. Contoh: `codecatalyst-ecs-cluster`
platform penyebaran	Nama platform penyebaran. Hardcode ke. `AWS:ECS`
layanan	Nama layanan Amazon ECS yang digunakan selama alur kerja dijalankan. Contoh: `codecatalyst-ecs-service`
task-definition-arn	Nama Sumber Daya Amazon (ARN) dari definisi tugas yang didaftarkan selama alur kerja dijalankan. Contoh: `arn:aws:ecs:us-west-2:111122223333:task-definition/codecatalyst-task-def:8`

Kunci	Nilai
	Contoh sebelumnya menunjukkan revisi yang terdaftar. `:8`
penyebaran-url	Tautan ke tab Acara konsol Amazon ECS, tempat Anda dapat melihat detail penerapan Amazon ECS yang terkait dengan alur kerja yang dijalankan. Contoh: `https://console.aws.amazon.com/ecs/home?region=us-west-2#/clusters/codecatalyst-ecs-cluster/services/codecatalyst-ecs-service/events`
region	Kode wilayah Wilayah AWS yang digunakan selama alur kerja dijalankan. Contoh: `us-west-2`

Tindakan “Terapkan ke Amazon ECS” definisi YAMAL

Berikut ini adalah definisi YAMAL dari tindakan Deploy to Amazon ECS. Untuk mempelajari cara menggunakan tindakan ini, lihatMenyebarkan aplikasi ke Amazon Elastic Container Service (ECS) dengan alur kerja.

Definisi tindakan ini ada sebagai bagian dalam file definisi alur kerja yang lebih luas. Untuk informasi selengkapnya tentang file ini, lihatAlur kerja definisi YAMAL.

> **Note**
>
> Sebagian besar properti YAMAL yang mengikuti memiliki elemen UI yang sesuai di editor visual. Untuk mencari elemen UI, gunakan Ctrl+F. Elemen akan terdaftar dengan properti YAMLnya yang terkait.

```
# The workflow definition starts here.
```

```
# See Properti tingkat atas for details.

Name: MyWorkflow
SchemaVersion: 1.0
Actions:

# The action definition starts here.
  ECSDeployAction_nn:
    Identifier: aws/ecs-deploy@v1
    DependsOn:
      - build-action
    Compute:
      Type: EC2 | Lambda
      Fleet: fleet-name
    Timeout: timeout-minutes
    Environment:
      Name: environment-name
      Connections:
        - Name: account-connection-name
          Role: DeployToECS
    Inputs:
      # Specify a source or an artifact, but not both.
      Sources:
        - source-name-1
      Artifacts:
        - task-definition-artifact
    Configuration:
      region: us-east-1
      cluster: ecs-cluster
      service: ecs-service
      task-definition: task-definition-path
      force-new-deployment: false|true
      codedeploy-appspec: app-spec-file-path
      codedeploy-application: application-name
      codedeploy-deployment-group: deployment-group-name
      codedeploy-deployment-description: deployment-description
```

ECSDeployAction

(Diperlukan)

Tentukan nama tindakan. Semua nama tindakan harus unik dalam alur kerja. Nama aksi terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak

diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama tindakan.

Default: `ECSDeployAction_nn`.

UI yang sesuai: Tab konfigurasi/Nama tampilan tindakan

Identifier

(*`ECSDeployAction`*/Identifier)

(Diperlukan)

Mengidentifikasi tindakan. Jangan mengubah properti ini kecuali Anda ingin mengubah versi. Untuk informasi selengkapnya, lihat Menentukan versi mayor, minor, atau patch dari suatu tindakan.

Default: `aws/ecs-deploy@v1`.

UI yang sesuai: Diagram alur kerja/ECS _nn/ DeployAction aws/ecs-deploy @v1 label

DependsOn

(*`ECSDeployAction`*/DependsOn)

(Opsional)

Tentukan tindakan, grup tindakan, atau gerbang yang harus berjalan dengan sukses agar tindakan ini berjalan.

Untuk informasi selengkapnya tentang fungsionalitas 'tergantung pada', lihat. Mengkonfigurasi tindakan untuk bergantung pada tindakan lain

UI yang sesuai: Tab masukan/Tergantung pada - opsional

Compute

(*`ECSDeployAction`*/Compute)

(Opsional)

Mesin komputasi yang digunakan untuk menjalankan tindakan alur kerja Anda. Anda dapat menentukan komputasi baik di tingkat alur kerja atau di tingkat tindakan, tetapi tidak keduanya. Ketika ditentukan pada tingkat alur kerja, konfigurasi komputasi berlaku untuk semua tindakan yang ditentukan dalam alur kerja. Pada tingkat alur kerja, Anda juga dapat menjalankan beberapa

tindakan pada instance yang sama. Untuk informasi selengkapnya, lihat Berbagi komputasi di seluruh tindakan.

UI yang sesuai: tidak ada

Type

(*ECSDeployAction*/Compute/Type)

(Diperlukan Compute jika disertakan)

Jenis mesin komputasi. Anda dapat menggunakan salah satu nilai berikut:

- EC2 (editor visual) atau `EC2` (editor YAMG)

  Dioptimalkan untuk fleksibilitas selama aksi berjalan.
- Lambda (editor visual) atau `Lambda` (editor YAMG)

  Kecepatan start-up aksi yang dioptimalkan.

Untuk informasi selengkapnya tentang jenis komputasi, lihatJenis komputasi.

UI yang sesuai: Tab Konfigurasi/Tingkat Lanjut - opsional/Jenis komputasi

Fleet

(*ECSDeployAction*/Compute/Fleet)

(Opsional)

Tentukan mesin atau armada yang akan menjalankan alur kerja atau tindakan alur kerja Anda. Dengan armada sesuai permintaan, ketika suatu tindakan dimulai, alur kerja menyediakan sumber daya yang dibutuhkan, dan mesin dihancurkan ketika tindakan selesai. Contoh armada sesuai permintaan:`Linux.x86-64.Large`,. `Linux.x86-64.XLarge` Untuk informasi lebih lanjut tentang armada sesuai permintaan, lihat. Properti armada sesuai permintaan

Dengan armada yang disediakan, Anda mengonfigurasi satu set mesin khusus untuk menjalankan tindakan alur kerja Anda. Mesin-mesin ini tetap menganggur, siap untuk memproses tindakan segera. Untuk informasi selengkapnya tentang armada yang disediakan, lihat. Properti armada yang disediakan

Jika `Fleet` dihilangkan, defaultnya adalah. `Linux.x86-64.Large`

UI yang sesuai: Tab Konfigurasi/Advanced - armada opsional/Komputasi

Timeout

(*ECSDeployAction*/Timeout)

(Opsional)

Tentukan jumlah waktu dalam menit (editor YAMAL), atau jam dan menit (editor visual), bahwa tindakan dapat berjalan sebelum CodeCatalyst mengakhiri tindakan. Minimal adalah 5 menit dan maksimum dijelaskan dalamKuota untuk alur kerja. Batas waktu default sama dengan batas waktu maksimum.

UI yang sesuai: Tab konfigurasi/Timeout - opsional

Environment

(*ECSDeployAction*/Environment)

(Diperlukan)

Tentukan CodeCatalyst lingkungan yang akan digunakan dengan tindakan.

Untuk informasi lebih lanjut tentang lingkungan, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst danPembuatan lingkungan.

UI yang sesuai: Tab Konfigurasi/'lingkungan/akun/peran'/Lingkungan

Name

(*ECSDeployAction*/Environment/Name)

(Diperlukan Environment jika disertakan)

Tentukan nama lingkungan yang ada yang ingin Anda kaitkan dengan tindakan.

UI yang sesuai: Tab Konfigurasi/'lingkungan/akun/peran'/Lingkungan

Connections

(*ECSDeployAction*/Environment/Connections)

(Diperlukan Environment jika disertakan)

Tentukan koneksi akun untuk dikaitkan dengan tindakan. Anda dapat menentukan maksimum satu koneksi akun di bawah`Environment`.

Untuk informasi selengkapnya tentang koneksi akun, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS. Untuk informasi tentang cara mengaitkan koneksi akun dengan lingkungan Anda, lihatPembuatan lingkungan.

UI yang sesuai: Tab konfigurasi/'lingkungan/akun/peran'/koneksi akun AWS

Name

(*ECSDeployAction*/Environment/Connections/Name)

(Diperlukan)

Tentukan nama koneksi akun.

UI yang sesuai: Tab konfigurasi/'lingkungan/akun/peran'/koneksi akun AWS

Role

(*ECSDeployAction*/Environment/Connections/Role)

(Diperlukan)

Tentukan nama peran IAM yang digunakan tindakan Deploy to Amazon ECS untuk mengakses. AWS Pastikan bahwa peran ini mencakup kebijakan berikut:

- Kebijakan izin berikut:

> **Warning**
>
> Batasi izin untuk yang ditampilkan dalam kebijakan berikut. Menggunakan peran dengan izin yang lebih luas dapat menimbulkan risiko keamanan.

```
{
    "Version": "2012-10-17",
    "Statement": [{
    "Action":[
      "ecs:DescribeServices",
      "ecs:CreateTaskSet",
      "ecs:DeleteTaskSet",
      "ecs:ListClusters",
      "ecs:RegisterTaskDefinition",
      "ecs:UpdateServicePrimaryTaskSet",
```

```
      "ecs:UpdateService",
      "elasticloadbalancing:DescribeTargetGroups",
      "elasticloadbalancing:DescribeListeners",
      "elasticloadbalancing:ModifyListener",
      "elasticloadbalancing:DescribeRules",
      "elasticloadbalancing:ModifyRule",
      "lambda:InvokeFunction",
      "lambda:ListFunctions",
      "cloudwatch:DescribeAlarms",
      "sns:Publish",
      "sns:ListTopics",
      "s3:GetObject",
      "s3:GetObjectVersion",
      "codedeploy:CreateApplication",
      "codedeploy:CreateDeployment",
      "codedeploy:CreateDeploymentGroup",
      "codedeploy:GetApplication",
      "codedeploy:GetDeployment",
      "codedeploy:GetDeploymentGroup",
      "codedeploy:ListApplications",
      "codedeploy:ListDeploymentGroups",
      "codedeploy:ListDeployments",
      "codedeploy:StopDeployment",
      "codedeploy:GetDeploymentTarget",
      "codedeploy:ListDeploymentTargets",
      "codedeploy:GetDeploymentConfig",
      "codedeploy:GetApplicationRevision",
      "codedeploy:RegisterApplicationRevision",
      "codedeploy:BatchGetApplicationRevisions",
      "codedeploy:BatchGetDeploymentGroups",
      "codedeploy:BatchGetDeployments",
      "codedeploy:BatchGetApplications",
      "codedeploy:ListApplicationRevisions",
      "codedeploy:ListDeploymentConfigs",
      "codedeploy:ContinueDeployment"
   ],
   "Resource":"*",
   "Effect":"Allow"
},{"Action":[
      "iam:PassRole"
   ],
   "Effect":"Allow",
   "Resource":"*",
   "Condition":{"StringLike":{"iam:PassedToService":[
```

```
            "ecs-tasks.amazonaws.com",
            "codedeploy.amazonaws.com"
         ]
      }
   }
}]
}
```

> **Note**
>
> Pertama kali peran digunakan, gunakan wildcard berikut dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

- Kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                 ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Pastikan peran ini ditambahkan ke koneksi akun Anda. Untuk mempelajari selengkapnya tentang menambahkan peran IAM ke koneksi akun, lihatMenambahkan peran IAM ke koneksi akun.

> **Note**
>
> Anda dapat menentukan nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran di sini, jika Anda mau. Untuk informasi selengkapnya tentang peran ini, silakan lihat Membuat CodeCatalystWorkflowDevelopmentRole-***spaceName***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami menyarankan Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian.

UI yang sesuai: Tab Konfigurasi/'lingkungan/akun/peran'/ Peran

Inputs

(*`ECSDeployAction`*/Inputs)

(Opsional)

`Inputs`Bagian ini mendefinisikan data yang `ECSDeployAction` dibutuhkan selama menjalankan alur kerja.

> **Note**
>
> Hanya satu input (baik sumber atau artefak) yang diizinkan per tindakan Deploy ke Amazon ECS.

UI yang sesuai: Tab input

Sources

(*`ECSDeployAction`*/Inputs/Sources)

(Diperlukan jika file definisi tugas Anda disimpan dalam repositori sumber)

Jika file definisi tugas Anda disimpan dalam repositori sumber, tentukan label repositori sumber tersebut. Saat ini, satu-satunya label yang didukung adalah`WorkflowSource`.

Jika file definisi tugas Anda tidak terkandung dalam repositori sumber, itu harus berada dalam artefak yang dihasilkan oleh tindakan lain.

Untuk informasi selengkapnya tentang sumber, lihat Menghubungkan alur kerja ke repositori sumber.

UI yang sesuai: Tab/Sumber Input - opsional

Artifacts - input

(*ECSDeployAction*/Inputs/Artifacts)

(Diperlukan jika file definisi tugas Anda disimpan dalam artefak keluaran dari tindakan sebelumnya)

Jika file definisi tugas yang ingin Anda terapkan terkandung dalam artefak yang dihasilkan oleh tindakan sebelumnya, tentukan artefak tersebut di sini. Jika file definisi tugas Anda tidak terkandung dalam artefak, itu harus berada di repositori sumber Anda.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab konfigurasi/Artefak - opsional

Configuration

(*ECSDeployAction*/Configuration)

(Diperlukan)

Bagian di mana Anda dapat menentukan properti konfigurasi tindakan.

UI yang sesuai: Tab konfigurasi

region

(Configuration/region)

(Diperlukan)

Tentukan AWS Wilayah tempat klaster dan layanan Amazon ECS Anda berada. Untuk daftar kode Wilayah, lihat Titik akhir Regional di. Referensi Umum AWS

UI yang sesuai: Tab konfigurasi/Wilayah

cluster

(*ECSDeployAction*/Configuration/cluster)

(Diperlukan)

Tentukan nama cluster Amazon ECS yang ada. Tindakan Deploy to Amazon ECS akan menerapkan aplikasi kontainer Anda sebagai tugas ke dalam klaster ini. Untuk informasi selengkapnya tentang klaster Amazon ECS, lihat Cluster di Panduan Pengembang Layanan Kontainer Elastis Amazon.

UI yang sesuai: Tab konfigurasi/Cluster

service

(*ECSDeployAction*/Configuration/service)

(Diperlukan)

Tentukan nama layanan Amazon ECS yang ada yang akan membuat instance file definisi tugas. Layanan ini harus berada di bawah cluster yang ditentukan di `cluster` bidang. Untuk informasi selengkapnya tentang layanan Amazon ECS, lihat Layanan Amazon ECS di Panduan Pengembang Layanan Kontainer Elastis Amazon.

UI yang sesuai: Tab konfigurasi/Layanan

task-definition

(*ECSDeployAction*/Configuration/task-definition)

(Diperlukan)

Tentukan jalur ke file definisi tugas yang ada. Jika file berada di repositori sumber Anda, jalurnya relatif terhadap folder root repositori sumber. Jika file Anda berada dalam artefak dari tindakan alur kerja sebelumnya, jalurnya relatif terhadap folder root artefak. Untuk informasi selengkapnya tentang file definisi tugas, lihat Definisi tugas di Panduan Pengembang Layanan Kontainer Elastis Amazon.

UI yang sesuai: Tab konfigurasi/Definisi tugas

force-new-deployment

(*ECSDeployAction*/Configuration/force-new-deployment)

(Diperlukan)

Jika diaktifkan, layanan Amazon ECS dapat memulai penerapan baru tanpa perubahan definisi layanan. Memaksa penerapan menyebabkan layanan menghentikan semua tugas yang sedang berjalan dan meluncurkan tugas baru. Untuk informasi selengkapnya tentang pemaksaan penerapan baru, lihat Memperbarui layanan di Panduan Pengembang Layanan Amazon Elastic Container.

Default: `false`

UI yang sesuai: Tab konfigurasi/Paksa penerapan layanan baru

codedeploy-appspec

(*`ECSDeployAction`*/Configuration/codedeploy-appspec)

(Diperlukan jika Anda telah mengonfigurasi layanan Amazon ECS Anda untuk menggunakan penerapan biru/hijau, jika tidak, hilangkan)

Tentukan nama dan jalur ke file spesifikasi CodeDeploy aplikasi (AppSpec) yang ada. File ini harus berada di root repositori CodeCatalyst sumber Anda. Untuk informasi selengkapnya tentang AppSpec file, lihat file spesifikasi CodeDeploy aplikasi (AppSpec) di Panduan AWS CodeDeploy Pengguna.

> **Note**
>
> Hanya berikan CodeDeploy informasi jika Anda telah mengonfigurasi layanan Amazon ECS Anda untuk melakukan penerapan biru/hijau. Untuk penerapan pembaruan bergulir (default), hilangkan CodeDeploy informasi. Untuk informasi selengkapnya tentang penerapan Amazon ECS, lihat jenis penerapan Amazon ECS di Panduan Pengembang Layanan Kontainer Elastis Amazon.

> **Note**
>
> CodeDeployBidang mungkin disembunyikan di editor visual. Untuk membuat mereka muncul, lihatMengapa CodeDeploy bidang hilang dari editor visual?.

UI yang sesuai: Tab konfigurasi/CodeDeploy AppSpec

codedeploy-application

(*`ECSDeployAction`*/Configuration/codedeploy-application)

(Diperlukan `codedeploy-appspec` jika disertakan)

Tentukan nama CodeDeploy aplikasi yang ada. Untuk informasi selengkapnya tentang CodeDeploy aplikasi, lihat Bekerja dengan aplikasi CodeDeploy di Panduan AWS CodeDeploy Pengguna.

UI yang sesuai: Tab/aplikasi CodeDeploy konfigurasi

codedeploy-deployment-group

(*ECSDeployAction*/Configuration/codedeploy-deployment-group)

(Diperlukan `codedeploy-appspec` jika disertakan)

Tentukan nama grup CodeDeploy penyebaran yang ada. Untuk informasi selengkapnya tentang grup CodeDeploy penerapan, lihat Bekerja dengan grup penerapan CodeDeploy di AWS CodeDeploy Panduan Pengguna.

UI yang sesuai: Tab konfigurasi/grup CodeDeploy penerapan

codedeploy-deployment-description

(*ECSDeployAction*/Configuration/codedeploy-deployment-description)

(Opsional)

Tentukan deskripsi penerapan yang akan dibuat oleh tindakan ini. Untuk informasi selengkapnya, lihat Bekerja dengan penerapan CodeDeploy di AWS CodeDeploy Panduan Pengguna.

UI yang sesuai: Tab konfigurasi/deskripsi CodeDeploy penerapan

Menerapkan aplikasi ke Amazon Elastic Kubernetes Service dengan alur kerja

> **Tip**
>
> Untuk tutorial yang menunjukkan cara menggunakan aksi cluster Deploy to Kubernetes, lihat. Tutorial: Menyebarkan aplikasi ke Amazon EKS

Bagian ini menjelaskan cara menerapkan aplikasi kontainer ke dalam klaster Kubernetes menggunakan alur kerja. CodeCatalyst Untuk mencapai hal ini, Anda harus menambahkan tindakan cluster Deploy ke Kubernetes ke alur kerja Anda. Tindakan ini menyebarkan aplikasi Anda ke klaster Kubernetes yang telah Anda siapkan di Amazon Elastic Kubernetes Service (EKS) menggunakan satu atau beberapa file manifes Kubernetes. Untuk contoh manifes, lihat deployment.yaml diTutorial: Menyebarkan aplikasi ke Amazon EKS.

Untuk informasi selengkapnya tentang Kubernetes, lihat Dokumentasi Kubernetes.

Untuk informasi selengkapnya tentang Amazon EKS, lihat Apa itu Amazon EKS? di Panduan Pengguna Amazon EKS.

Cara kerja aksi “Deploy to Kubernetes cluster”

Kluster Deploy ke Kubernetes berfungsi sebagai berikut:

1. Saat runtime, action akan menginstal `kubectl` utilitas Kubernetes ke mesin CodeCatalyst komputasi tempat aksi berjalan. Tindakan dikonfigurasi `kubectl` untuk menunjuk ke kluster Amazon EKS yang Anda berikan saat mengonfigurasi tindakan. `kubectl`Utilitas diperlukan untuk menjalankan `kubectl apply` perintah, selanjutnya.
2. Tindakan ini menjalankan `kubectl apply -f` *`my-manifest.yaml`* perintah, yang menjalankan instruksi di *`my-manifest.yaml`* untuk menerapkan aplikasi Anda sebagai satu set kontainer dan pod ke dalam cluster yang dikonfigurasi. Untuk informasi selengkapnya tentang perintah ini, lihat topik kubectl apply di Kubernetes Reference Documentation.

Topik

- Tutorial: Menyebarkan aplikasi ke Amazon EKS
- Menambahkan aksi “Deploy to Kubernetes cluster”
- Variabel yang dihasilkan oleh aksi “Deploy to Kubernetes cluster”
- Tindakan “Terapkan ke Kluster Kubernetes” Definisi YAMAL

Tutorial: Menyebarkan aplikasi ke Amazon EKS

Dalam tutorial ini, Anda mempelajari cara menerapkan aplikasi container ke Amazon Elastic Kubernetes Service menggunakan alur kerja Amazon, CodeCatalyst Amazon EKS, dan beberapa layanan lainnya. AWS Aplikasi yang digunakan adalah 'Halo, Dunia! 'sederhana situs web yang dibangun di atas gambar Docker server web Apache. Tutorial memandu Anda melalui pekerjaan persiapan yang diperlukan seperti menyiapkan mesin pengembangan dan kluster Amazon EKS, dan kemudian menjelaskan cara membuat alur kerja untuk membangun aplikasi dan menerapkannya ke dalam cluster.

Setelah penerapan awal selesai, tutorial menginstruksikan Anda untuk membuat perubahan pada sumber aplikasi Anda. Perubahan ini menyebabkan image Docker baru dibangun dan didorong ke repositori gambar Docker Anda dengan informasi revisi baru. Revisi baru gambar Docker kemudian diterapkan ke Amazon EKS.

> **Tip**
>
> Alih-alih mengerjakan tutorial ini, Anda dapat menggunakan cetak biru yang melakukan pengaturan Amazon EKS lengkap untuk Anda. Anda harus menggunakan cetak biru Penerapan Aplikasi EKS. Untuk informasi selengkapnya, lihat Membuat proyek dengan cetak biru.

Topik

Prasyarat

Sebelum Anda memulai tutorial ini:

- Anda memerlukan CodeCatalyst ruang Amazon dengan AWS akun yang terhubung. Untuk informasi selengkapnya, lihat Menciptakan ruang.
- Di ruang Anda, Anda memerlukan CodeCatalyst proyek kosong, Mulai dari awal yang disebut:

  ```
  codecatalyst-eks-project
  ```

  Untuk informasi selengkapnya, lihat Membuat proyek kosong di Amazon CodeCatalyst.
- Dalam proyek Anda, Anda memerlukan repositori CodeCatalyst sumber kosong yang disebut:

```
codecatalyst-eks-source-repository
```

Untuk informasi selengkapnya, lihat Simpan dan berkolaborasi pada kode dengan repositori sumber di CodeCatalyst.

- Dalam proyek Anda, Anda memerlukan lingkungan CodeCatalyst CI/CD (bukan Lingkungan Dev) yang disebut:

```
codecatalyst-eks-environment
```

Konfigurasikan lingkungan ini sebagai berikut:

- Pilih jenis apa saja, seperti Non-produksi.
- Hubungkan AWS akun Anda ke sana.

Untuk informasi selengkapnya, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst.

Langkah 1: Siapkan mesin pengembangan Anda

Langkah pertama dalam tutorial ini adalah mengkonfigurasi mesin pengembangan dengan beberapa alat yang akan Anda gunakan sepanjang tutorial ini. Alat-alat ini adalah:

- `eksctl`utilitas - untuk pembuatan cluster
- `kubectl`utilitas — prasyarat untuk `eksctl`
- yang AWS CLI — juga merupakan prasyarat untuk `eksctl`

Anda dapat menginstal alat ini di mesin pengembangan yang ada jika Anda memilikinya, atau Anda dapat menggunakan Lingkungan CodeCatalyst Dev, yang berbasis Cloud. Manfaat dari CodeCatalyst Dev Environment adalah mudah untuk berputar dan turun, dan terintegrasi dengan CodeCatalyst layanan lain, memungkinkan Anda untuk bekerja melalui tutorial ini dalam langkah-langkah yang lebih sedikit.

Tutorial ini mengasumsikan Anda akan menggunakan CodeCatalyst Dev Environment.

Petunjuk berikut menjelaskan cara cepat untuk meluncurkan Lingkungan CodeCatalyst Dev dan mengonfigurasinya dengan alat yang diperlukan, tetapi jika Anda menginginkan instruksi terperinci, lihat:

- Membuat Lingkungan Dev dalam panduan ini.
- Menginstal kubectl di Panduan Pengguna Amazon EKS.
- Menginstal atau memutakhirkan eksctl di Panduan Pengguna Amazon EKS.
- Menginstal atau memperbarui versi terbaru dari AWS CLI dalam Panduan AWS Command Line Interface Pengguna.

Untuk meluncurkan Lingkungan Dev

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda,`codecatalyst-eks-project`.
3. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
4. Pilih nama repositori sumber Anda,. `codecatalyst-eks-source-repository`
5. Di dekat bagian atas pilih Create Dev Environment, lalu pilih AWS Cloud9 (di browser).
6. Pastikan bahwa Work di cabang dan main yang ada dipilih, lalu pilih Buat.

   Lingkungan Dev Anda diluncurkan di tab browser baru, dan repositori (`codecatalyst-eks-source-repository`) Anda dikloning ke dalamnya.

Untuk menginstal dan mengkonfigurasi kubectl

1. Di terminal Dev Environment, masukkan:

   ```
   curl -o kubectl https://amazon-eks.s3.us-west-2.amazonaws.com/1.18.9/2020-11-02/bin/linux/amd64/kubectl
   ```

2. Masukkan:

   ```
   chmod +x ./kubectl
   ```

3. Masukkan:

   ```
   mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$PATH:$HOME/bin
   ```

4. Masukkan:

   ```
   echo 'export PATH=$PATH:$HOME/bin' >> ~/.bashrc
   ```

5. Masukkan:

```
kubectl version --short --client
```

6. Periksa apakah ada versi yang muncul.

   Anda sekarang telah menginstal`kubectl`.

Untuk menginstal dan mengkonfigurasi eksctl

> **Note**
>
> `eksctl`tidak sepenuhnya diperlukan karena Anda dapat menggunakannya `kubectl` sebagai gantinya. Namun, `eksctl` memiliki manfaat mengotomatisasi banyak konfigurasi cluster, dan oleh karena itu alat yang direkomendasikan untuk tutorial ini.

1. Di terminal Dev Environment, masukkan:

```
curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/
download/eksctl_$(uname -s)_amd64.tar.gz" | tar xz -C /tmp
```

2. Masukkan:

```
sudo cp /tmp/eksctl /usr/bin
```

3. Masukkan:

```
eksctl version
```

4. Periksa apakah ada versi yang muncul.

   Anda sekarang telah menginstale`ksctl`.

Untuk memverifikasi bahwa AWS CLI sudah diinstal

1. Di terminal Dev Environment, masukkan:

```
aws --version
```

2. Periksa apakah versi muncul untuk memverifikasi bahwa AWS CLI sudah diinstal.

   Lengkapi prosedur yang tersisa untuk mengkonfigurasi AWS CLI dengan izin yang diperlukan untuk mengakses AWS.

Untuk mengkonfigurasi AWS CLI

Anda harus mengonfigurasi tombol akses AWS CLI dengan dan token sesi untuk memberikannya akses ke AWS layanan. Petunjuk berikut menyediakan cara cepat untuk mengonfigurasi kunci dan token, tetapi jika Anda menginginkan instruksi terperinci, lihat Mengonfigurasi AWS CLI di Panduan AWS Command Line Interface Pengguna.

1. Buat pengguna IAM Identity Center, sebagai berikut:

   a. Masuk ke AWS Management Console dan buka AWS IAM Identity Center konsol di https://console.aws.amazon.com/singlesignon/.

      (Anda mungkin perlu memilih Aktifkan jika Anda belum pernah masuk ke Pusat Identitas IAM sebelumnya.)

      > **Note**
      >
      > Pastikan Anda masuk menggunakan Akun AWS yang terhubung ke CodeCatalyst ruang Anda. Anda dapat memverifikasi akun mana yang terhubung dengan menavigasi ke ruang Anda dan memilih tab akun AWS. Untuk informasi selengkapnya, lihat Menciptakan ruang.

   b. Pada panel navigasi, silakan pilih Pengguna, lalu pilih Tambahkan pengguna.

   c. Di Nama Pengguna, masukkan:

      ```
      codecatalyst-eks-user
      ```

   d. Di bawah Kata Sandi, pilih Buat kata sandi satu kali yang dapat Anda bagikan dengan pengguna ini.

   e. Di Alamat Email dan Konfirmasi alamat email, masukkan alamat email yang belum ada di Pusat Identitas IAM.

   f. Di Nama depan, masukkan:

```
codecatalyst-eks-user
```

g. Di Nama belakang, masukkan:

```
codecatalyst-eks-user
```

h. Di Nama tampilan, simpan:

```
codecatalyst-eks-user codecatalyst-eks-user
```

i. Pilih Selanjutnya.

j. Pada halaman Tambahkan pengguna ke grup, pilih Berikutnya.

k. Pada halaman Tinjau dan tambahkan pengguna, tinjau informasi dan pilih Tambah pengguna.

Kotak dialog kata sandi satu kali muncul.

l. Pilih Salin lalu tempel informasi masuk ke file teks. Informasi masuk terdiri dari URL portal AWS akses, nama pengguna, dan kata sandi satu kali.

m. Pilih Tutup.

2. Buat set izin, sebagai berikut:

a. Di panel navigasi, pilih Set izin, lalu pilih Buat set izin.

b. Pilih Set izin yang telah ditentukan sebelumnya dan kemudian pilih AdministratorAccess. Kebijakan ini memberikan izin penuh untuk semua Layanan AWS.

c. Pilih Selanjutnya.

d. Dalam nama set Izin, hapus `AdministratorAccess` dan masukkan:

```
codecatalyst-eks-permission-set
```

e. Pilih Selanjutnya.

f. Pada halaman Tinjau dan buat, tinjau informasi dan pilih Buat.

3. Tetapkan izin yang disetel ke`codecatalyst-eks-user`, sebagai berikut:

a. Di panel navigasi, pilih Akun AWS, lalu pilih kotak centang di Akun AWS samping tempat Anda masuk saat ini.

b. Pilih Tetapkan pengguna atau grup.

c. Pilih tab Pengguna.

d. Pilih kotak centang di sebelah`codecatalyst-eks-user`.

e. Pilih Selanjutnya.

f. Pilih kotak centang di sebelah`codecatalyst-eks-permission-set`.

g. Pilih Selanjutnya.

h. Tinjau informasi dan pilih Kirim.

Anda sekarang telah menetapkan `codecatalyst-eks-user` dan `codecatalyst-eks-permission-set` untuk Anda Akun AWS, mengikat mereka bersama-sama.

4. Dapatkan `codecatalyst-eks-user` kunci akses dan token sesi, sebagai berikut:

a. Pastikan Anda memiliki URL portal AWS akses dan nama pengguna dan kata sandi satu kali untuk`codecatalyst-eks-user`. Anda seharusnya menyalin informasi ini ke editor teks sebelumnya.

> **Note**
>
> Jika Anda tidak memiliki informasi ini, buka halaman `codecatalyst-eks-user` detail di Pusat Identitas IAM, pilih Atur ulang kata sandi, Hasilkan kata sandi satu kali [...] , dan Atur ulang kata sandi lagi untuk menampilkan informasi di layar.

b. Keluar dari AWS.

c. Rekatkan URL portal AWS akses ke bilah alamat browser Anda.

d. Masuk dengan:

- Nama Pengguna:

```
codecatalyst-eks-user
```

- Kata sandi:

one-time-password

e. Di Tetapkan kata sandi baru, masukkan kata sandi baru dan pilih Atur kata sandi baru.

Sebuah Akun AWSkotak muncul di layar.

f. Pilih Akun AWS, lalu pilih nama yang Akun AWS Anda tetapkan `codecatalyst-eks-user` pengguna dan set izin.

g. Di sampingcodecatalyst-eks-permission-set, pilih Baris perintah atau akses terprogram.

h. Salin perintah di tengah halaman. Mereka terlihat mirip dengan yang berikut:

```
export AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
export AWS_SECRET_ACCESS_KEY="wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
export AWS_SESSION_TOKEN="session-token"
```

... di mana *sesi-token* adalah string acak panjang.

5. Tambahkan kunci akses dan token sesi ke AWS CLI, sebagai berikut:

a. Kembali ke Lingkungan CodeCatalyst Pengembang Anda.

b. Pada prompt terminal, tempel perintah yang Anda salin. Tekan Enter.

Anda sekarang telah mengonfigurasi tombol akses AWS CLI dengan dan token sesi. Anda sekarang dapat menggunakan AWS CLI untuk menyelesaikan tugas-tugas yang diperlukan oleh tutorial ini.

> **Important**
>
> Jika sewaktu-waktu selama tutorial ini Anda melihat pesan yang mirip dengan:
> `Unable to locate credentials. You can configure credentials by running "aws configure".`
> Atau:
> `ExpiredToken: The security token included in the request is expired`
> ... itu karena AWS CLI sesi Anda telah kedaluwarsa. Dalam hal ini, jangan jalankan `aws configure` perintah. Sebagai gantinya, gunakan instruksi pada langkah 4 dari prosedur ini yang dimulai dengan `Obtain codecatalyst-eks-user's access key and session token` untuk menyegarkan sesi Anda.

Langkah 2: Buat cluster Amazon EKS

Di bagian ini, Anda membuat cluster di Amazon EKS. Petunjuk di bawah ini menjelaskan cara cepat untuk membuat cluster menggunakan`eksctl`, tetapi jika Anda ingin petunjuk rinci, lihat:

- Memulai dengan eksctl di Panduan Pengguna Amazon EKS

atau

- Memulai konsol dan AWS CLI di Panduan Pengguna Amazon EKS (topik ini memberikan `kubectl` instruksi untuk membuat cluster)

> **Note**
>
> Cluster pribadi tidak didukung oleh CodeCatalyst integrasi dengan Amazon EKS.

Sebelum Anda memulai

Pastikan Anda telah menyelesaikan tugas-tugas berikut di mesin pengembangan Anda:

- Menginstal `eksctl` utilitas.
- Menginstal `kubectl` utilitas.
- Menginstal AWS CLI dan mengkonfigurasinya dengan kunci akses dan token sesi.

Untuk informasi tentang cara menyelesaikan tugas-tugas ini, lihatLangkah 1: Siapkan mesin pengembangan Anda.

Untuk membuat klaster DB

> **Important**
>
> Jangan gunakan antarmuka pengguna layanan Amazon EKS untuk membuat klaster karena klaster tidak akan dikonfigurasi dengan benar. Gunakan `eksctl` utilitas, seperti yang dijelaskan dalam langkah-langkah berikut.

1. Pergi ke Lingkungan Pengembang Anda.
2. Buat cluster dan node:

   ```
   eksctl create cluster --name codecatalyst-eks-cluster --region us-west-2
   ```

   Di mana:

- *codecatalyst-eks-cluster*diganti dengan nama yang ingin Anda berikan pada cluster Anda.
- *us-west-2* diganti dengan Wilayah Anda.

Setelah 10-20 menit, pesan yang mirip dengan yang berikut ini muncul:

```
EKS cluster "codecatalyst-eks-cluster" in "us-west-2" region is ready
```

> **Note**
>
> Anda akan melihat beberapa `waiting for CloudFormation stack` pesan saat AWS membuat cluster Anda. Ini yang diharapkan.

3. Verifikasi bahwa klaster Anda berhasil dibuat:

```
kubectl cluster-info
```

Anda akan melihat pesan yang mirip dengan berikut ini, yang menunjukkan pembuatan cluster yang berhasil:

```
Kubernetes master is running at https://long-string.gr7.us-west-2.eks.amazonaws.com
CoreDNS is running at https://long-string.gr7.us-west-2.eks.amazonaws.com/api/v1/
namespaces/kube-system/services/kube-dns:dns/proxy
```

Langkah 3: Buat repositori gambar Amazon ECR

Di bagian ini, Anda membuat repositori gambar pribadi di Amazon Elastic Container Registry (Amazon ECR) Registry ECR). Repositori ini menyimpan gambar Docker untuk tutorial.

Untuk informasi selengkapnya tentang Amazon ECR, lihat Panduan Pengguna Amazon Elastic Container Registry.

Untuk membuat repositori gambar di Amazon ECR

1. Pergi ke Lingkungan Pengembang Anda.
2. Buat repositori kosong di Amazon ECR:

```
aws ecr create-repository --repository-name codecatalyst-eks-image-repo
```

Ganti *codecatalyst-eks-image-repo*dengan nama yang ingin Anda berikan pada repositori Amazon ECR.

Tutorial ini mengasumsikan Anda menamai `codecatalyst-eks-image-repo` repositori Anda.

3. Tampilkan detail repositori Amazon ECR:

```
aws ecr describe-repositories \
      --repository-names codecatalyst-eks-image-repo
```

4. Perhatikan `"repositoryUri":` nilainya, misalnya,`111122223333.dkr.ecr.us-west-2.amazonaws.com/codecatalyst-eks-image-repo`.

   Anda membutuhkannya nanti saat menambahkan repositori ke alur kerja Anda.

Langkah 4: Tambahkan file sumber

Di bagian ini, Anda menambahkan file sumber aplikasi ke repositori sumber Anda ()`codecatalyst-eks-source-repository`. Mereka terdiri dari:

- Sebuah `index.html` file - Menampilkan 'Hello, World! ' pesan di browser.
- A Dockerfile - Menjelaskan gambar dasar yang akan digunakan untuk image Docker Anda dan perintah Docker untuk menerapkannya.
- `deployment.yaml`File — Manifes Kubernetes yang mendefinisikan layanan dan penerapan Kubernetes.

Struktur folder adalah sebagai berikut:

```
|— codecatalyst-eks-source-repository
   |— Kubernetes
      |— deployment.yaml
   |— public-html
   |  |— index.html
   |— Dockerfile
```

Topik

- index.html
- Dockerfile
- deployment.yaml

index.html

`index.html`File tersebut menampilkan 'Hello, World! ' pesan di browser.

Untuk menambahkan file index.html

1. Pergi ke Lingkungan Pengembang Anda.
2. Di`codecatalyst-eks-source-repository`, buat folder bernama`public-html`.
3. Di`/public-html`, buat file yang disebut `index.html` dengan konten berikut:

```
<html>
  <head>
    <title>Hello World</title>
    <style>
      body {
      background-color: black;
      text-align: center;
      color: white;
      font-family: Arial, Helvetica, sans-serif;
      }
    </style>
  </head>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```

4. Pada prompt terminal, masukkan:

```
cd /projects/codecatalyst-eks-source-repository
```

5. Tambahkan, komit, dan dorong:

```
git add .
git commit -m "add public-html/index.html"
git push
```

Ditambahkan ke repositori Anda dalam folder. `index.html` `public-html`

Dockerfile

Dockerfile menjelaskan image Docker dasar yang akan digunakan dan perintah Docker untuk diterapkan padanya. Untuk informasi selengkapnya tentang Dockerfile, lihat Referensi Dockerfile.

Dockerfile yang ditentukan di sini menunjukkan untuk menggunakan image dasar Apache 2.4 (). `httpd` Ini juga mencakup instruksi untuk menyalin file sumber yang dipanggil `index.html` ke folder di server Apache yang melayani halaman web. `EXPOSE`Instruksi di Dockerfile memberi tahu Docker bahwa wadah mendengarkan pada port 80.

Untuk menambahkan Dockerfile

1. Di`codecatalyst-eks-source-repository`, buat file yang disebut `Dockerfile` dengan konten berikut:

   ```
   FROM httpd:2.4
   COPY ./public-html/index.html /usr/local/apache2/htdocs/index.html
   EXPOSE 80
   ```

   Jangan sertakan ekstensi file.

   > **Important**
   >
   > Dockerfile harus berada di folder root repositori Anda. `Docker build`Perintah alur kerja mengharapkannya ada di sana.

2. Tambahkan, komit, dan dorong:

   ```
   git add .
   git commit -m "add Dockerfile"
   git push
   ```

   Dockerfile ditambahkan ke repositori Anda.

deployment.yaml

Di bagian ini, Anda menambahkan `deployment.yaml` file ke repositori Anda. `deployment.yaml`File ini adalah manifes Kubernetes yang mendefinisikan dua tipe atau jenis resource Kubernetes untuk dijalankan: sebuah 'service' dan 'deployment'.

- 'Layanan' menyebarkan penyeimbang beban ke Amazon EC2. Load balancer memberi Anda URL publik yang menghadap ke Internet dan port standar (port 80) yang dapat Anda gunakan untuk menelusuri 'Hello, World! ' aplikasi.
- 'Deployment 'menyebarkan tiga pod, dan setiap pod akan berisi kontainer Docker dengan 'Hello, World!' aplikasi. Ketiga pod tersebut di-deploy ke node yang dibuat saat Anda membuat cluster.

Manifes dalam tutorial ini singkat; namun, manifes dapat mencakup sejumlah tipe sumber daya Kubernetes, seperti pod, job, ingresses, dan kebijakan jaringan. Selanjutnya, Anda dapat menggunakan beberapa file manifes jika penerapan Anda rumit.

Untuk menambahkan berkas deployment.yaml

1. Di`codecatalyst-eks-source-repository`, buat folder bernama`Kubernetes`.
2. Di`/Kubernetes`, buat file yang disebut `deployment.yaml` dengan konten berikut:

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
  labels:
    app: my-app
spec:
  type: LoadBalancer
  selector:
    app: my-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deployment
```

```
  labels:
    app: my-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      containers:
      - name: codecatalyst-eks-container
        # The $REPOSITORY_URI and $IMAGE_TAG placeholders will be replaced by
 actual values supplied by the build action in your workflow
        image: $REPOSITORY_URI:$IMAGE_TAG
        ports:
        - containerPort: 80
```

3. Tambahkan, komit, dan dorong:

```
git add .
git commit -m "add Kubernetes/deployment.yaml"
git push
```

`deployment.yaml`File ditambahkan ke repositori Anda dalam folder bernama. `Kubernetes`

Anda sekarang telah menambahkan semua file sumber Anda.

Luangkan waktu sejenak untuk memeriksa ulang pekerjaan Anda dan pastikan Anda menempatkan semua file di folder yang benar. Struktur folder adalah sebagai berikut:

```
|— codecatalyst-eks-source-repository
   |— Kubernetes
      |— deployment.yaml
   |— public-html
   |  |— index.html
   |— Dockerfile
```

Langkah 5: Buat AWS peran

Di bagian ini, Anda membuat peran AWS IAM yang dibutuhkan CodeCatalyst alur kerja Anda agar berfungsi. Peran ini adalah:

- Peran build - Memberikan izin tindakan CodeCatalyst build (dalam alur kerja) untuk mengakses AWS akun Anda dan menulis ke Amazon ECR dan Amazon EC2.
- Deploy role — Memberikan izin tindakan klaster CodeCatalyst Deploy ke Kubernetes (dalam alur kerja) untuk mengakses akun Anda dan Amazon EKS. AWS

Untuk informasi selengkapnya tentang peran IAM, lihat peran IAM di AWS Identity and Access Management Panduan Pengguna.

> **Note**
>
> Untuk menghemat waktu, Anda dapat membuat satu peran, yang disebut `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran, alih-alih dua peran yang tercantum sebelumnya. Untuk informasi selengkapnya, lihat Membuat CodeCatalystWorkflowDevelopmentRole-***`spaceName`***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami menyarankan Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian. Tutorial ini mengasumsikan Anda membuat dua peran yang tercantum sebelumnya.

Untuk membuat peran build dan deploy, selesaikan rangkaian prosedur berikut.

1. Untuk membuat kebijakan kepercayaan untuk kedua peran

1. Pergi ke Lingkungan Pengembang Anda.
2. Di `Cloud9-`*`long-string`* direktori, buat file yang disebut `codecatalyst-eks-trust-policy.json` dengan konten berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
```

```
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                 ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

2. Untuk membuat kebijakan build untuk peran build

- Di `Cloud9-`*`long-string`* direktori, buat file yang disebut `codecatalyst-eks-build-policy.json` dengan konten berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:*",
                "ec2:*"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

3. Untuk membuat kebijakan penerapan untuk peran penerapan

- Di `Cloud9-long-string` direktori, buat file yang disebut `codecatalyst-eks-deploy-policy.json` dengan konten berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:DescribeCluster",
                "eks:ListClusters"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

Anda sekarang telah menambahkan tiga dokumen kebijakan ke Lingkungan Pengembang Anda. Struktur direktori Anda sekarang terlihat seperti ini:

```
|— Cloud9-long-string
   |— .c9
   |— codecatalyst-eks-source-repository
      |— Kubernetes
      |— public-html
      |— Dockerfile
   codecatalyst-eks-build-policy.json
   codecatalyst-eks-deploy-policy.json
   codecatalyst-eks-trust-policy.json
```

4. Untuk menambahkan kebijakan build ke AWS

1. Di terminal Dev Environment, masukkan:

```
cd /projects
```

2. Masukkan:

```
aws iam create-policy \
    --policy-name codecatalyst-eks-build-policy \
    --policy-document file://codecatalyst-eks-build-policy.json
```

3. Tekan Enter.
4. Dalam output perintah, perhatikan `"arn":` nilainya, misalnya,`arn:aws:iam::111122223333:policy/codecatalyst-eks-build-policy`. Anda membutuhkan ARN ini nanti.

5. Untuk menambahkan kebijakan penerapan ke AWS

1. Masukkan:

```
aws iam create-policy \
    --policy-name codecatalyst-eks-deploy-policy \
    --policy-document file://codecatalyst-eks-deploy-policy.json
```

2. Tekan Enter.
3. Dalam output perintah, perhatikan `"arn":` nilai kebijakan penerapan, misalnya,`arn:aws:iam::111122223333:policy/codecatalyst-eks-deploy-policy`. Anda membutuhkan ARN ini nanti.

6. Untuk membuat peran build

1. Masukkan:

```
aws iam create-role \
      --role-name codecatalyst-eks-build-role \
      --assume-role-policy-document file://codecatalyst-eks-trust-policy.json
```

2. Tekan Enter.

3. Masukkan:

```
aws iam attach-role-policy \
      --role-name codecatalyst-eks-build-role \
      --policy-arn arn:aws:iam::111122223333:policy/codecatalyst-eks-build-policy
```

Di mana *arn:aws:iam: codecatalyst-eks-build-policy :111122223333:policy/* diganti dengan ARN dari kebijakan build yang Anda sebutkan sebelumnya.

4. Tekan Enter.
5. Pada prompt terminal, masukkan:

```
aws iam get-role \
      --role-name codecatalyst-eks-build-role
```

6. Tekan Enter.
7. Perhatikan `"Arn":` nilai peran, misalnya,`arn:aws:iam::111122223333:role/codecatalyst-eks-build-role`. Anda membutuhkan ARN ini nanti.

7. Untuk membuat peran penerapan

1. Masukkan:

```
aws iam create-role \
      --role-name codecatalyst-eks-deploy-role \
      --assume-role-policy-document file://codecatalyst-eks-trust-policy.json
```

2. Tekan Enter.
3. Masukkan:

```
aws iam attach-role-policy \
      --role-name codecatalyst-eks-deploy-role \
      --policy-arn arn:aws:iam::111122223333:policy/codecatalyst-eks-deploy-policy
```

Di mana *arn:aws:iam: codecatalyst-eks-deploy-policy :111122223333:policy/* diganti dengan ARN dari kebijakan penerapan yang Anda sebutkan sebelumnya.

4. Tekan Enter.
5. Masukkan:

```
aws iam get-role \
      --role-name codecatalyst-eks-deploy-role
```

6. Tekan Enter.
7. Perhatikan `"Arn":` nilai peran, misalnya,`arn:aws:iam::111122223333:role/codecatalyst-eks-deploy-role`. Anda membutuhkan ARN ini nanti.

Anda sekarang telah membuat peran build dan deploy dan mencatat ARN mereka.

Langkah 6: Tambahkan AWS peran ke CodeCatalyst

Pada langkah ini, Anda menambahkan build role (`codecatalyst-eks-build-role`) dan deploy role (`codecatalyst-eks-deploy-role`) ke Akun AWS yang Anda sambungkan ke ruang Anda. Ini membuat peran tersedia untuk digunakan dalam alur kerja Anda.

Untuk menambahkan peran build dan deploy ke Akun AWS

1. Di CodeCatalyst konsol, navigasikan ke ruang Anda.
2. Di bagian atas, pilih Pengaturan.
3. Di panel navigasi, pilih AWS akun. Daftar akun muncul.
4. Di kolom nama CodeCatalyst tampilan Amazon, salin nama tampilan Akun AWS tempat Anda membuat peran build dan deploy. (Mungkin nomor.) Anda akan membutuhkan nilai ini nanti, saat membuat alur kerja Anda.
5. Pilih nama tampilan.
6. Pilih Kelola peran dari konsol AWS manajemen.

   Peran Tambahkan IAM ke halaman CodeCatalyst ruang Amazon muncul. Anda mungkin perlu masuk untuk mengakses halaman.
7. Pilih Tambahkan peran yang sudah Anda buat di IAM.

   Daftar drop-down muncul. Daftar ini menampilkan peran build dan deploy, dan peran IAM lainnya dengan kebijakan kepercayaan yang mencakup prinsip `codecatalyst-runner.amazonaws.com` dan `codecatalyst.amazonaws.com` layanan.
8. Dari daftar drop-down, tambahkan:

   - `codecatalyst-eks-build-role`

- `codecatalyst-eks-deploy-role`

> **Note**
>
> Jika Anda melihat`The security token included in the request is invalid`, itu mungkin karena Anda tidak memiliki izin yang tepat. Untuk memperbaiki masalah ini, keluar dari AWS sebagai masuk kembali dengan AWS akun yang Anda gunakan saat membuat CodeCatalyst ruang.

9. Kembali ke CodeCatalyst konsol dan segarkan halaman.

   Peran build dan deploy sekarang akan muncul di bawah peran IAM.

   Peran ini sekarang tersedia untuk digunakan dalam CodeCatalyst alur kerja.

Langkah 7: Perbarui ConfigMap

Anda harus menambahkan peran deploy yang Anda buat ke `ConfigMap` file Kubernetes Langkah 5: Buat AWS peran untuk memberikan tindakan klaster Deploy ke Kubernetes (dalam alur kerja Anda) kemampuan untuk mengakses dan berinteraksi dengan klaster Anda. Anda dapat menggunakan `eksctl` atau `kubectl` melakukan tugas ini.

Untuk mengkonfigurasi file Kubernetes ConfigMap menggunakan eksctl

- Di terminal Dev Environment, masukkan:

```
eksctl create iamidentitymapping --cluster codecatalyst-eks-cluster --
arn arn:aws:iam::111122223333:role/codecatalyst-eks-deploy-role --group
 system:masters --username codecatalyst-eks-deploy-role --region us-west-2
```

  Di mana:

  - *`codecatalyst-eks-cluster`*diganti dengan nama cluster cluster Amazon EKS.
  - *`arn:aws:iam: :111122223333:role/ codecatalyst-eks-deploy-role`* diganti dengan ARN dari peran penerapan yang Anda buat. Langkah 5: Buat AWS peran
  - *`codecatalyst-eks-deploy-role`*(di sebelah`--username`) diganti dengan nama peran penerapan yang Anda buat. Langkah 5: Buat AWS peran

> **Note**
>
> Jika Anda memutuskan untuk tidak membuat peran penerapan, ganti *codecatalyst-eks-deploy-role* dengan nama `CodeCatalystWorkflowDevelopmentRole-`*spaceName* peran. Untuk informasi selengkapnya tentang peran ini, silakan lihat Langkah 5: Buat AWS peran.

- *us-west-2* diganti dengan Wilayah Anda.

Untuk detail tentang perintah ini, lihat Mengelola pengguna dan peran IAM.

Pesan yang mirip dengan berikut ini muncul:

```
2023-06-09 00:58:29 [#]  checking arn arn:aws:iam::111122223333:role/codecatalyst-
eks-deploy-role against entries in the auth ConfigMap
2023-06-09 00:58:29 [#]  adding identity "arn:aws:iam::111122223333:role/
codecatalyst-eks-deploy-role" to auth ConfigMap
```

Untuk mengkonfigurasi file Kubernetes ConfigMap menggunakan kubectl

1. Di terminal Dev Environment, masukkan:

```
kubectl edit configmap -n kube-system aws-auth
```

ConfigMap File muncul di layar.

2. Tambahkan teks dengan huruf miring merah:

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file
 will be
# reopened with the relevant failures.
#
apiVersion: v1
data:
  mapRoles: |
    - groups:
      - system:bootstrappers
      - system:nodes
```

```
      rolearn: arn:aws:iam::111122223333:role/eksctl-codecatalyst-eks-cluster-n-
NodeInstanceRole-16BC456ME6YR5
      username: system:node:{{EC2PrivateDNSName}}
    - groups:
      - system:masters
      rolearn: arn:aws:iam::111122223333:role/codecatalyst-eks-deploy-role
      username: codecatalyst-eks-deploy-role
  mapUsers: |
    []
kind: ConfigMap
metadata:
  creationTimestamp: "2023-06-08T19:04:39Z"
  managedFields:
  ...
```

Di mana:

- *arn:aws:iam: :111122223333:role/ codecatalyst-eks-deploy-role* diganti dengan ARN dari peran penerapan yang Anda buat. Langkah 5: Buat AWS peran
- *codecatalyst-eks-deploy-role*(di sebelah`username:`) diganti dengan nama peran penerapan yang Anda buat. Langkah 5: Buat AWS peran

> **Note**
>
> Jika Anda memutuskan untuk tidak membuat peran penerapan, ganti *codecatalyst-eks-deploy-role*dengan nama `CodeCatalystWorkflowDevelopmentRole-`*spaceName* peran. Untuk informasi selengkapnya tentang peran ini, silakan lihat Langkah 5: Buat AWS peran.

Untuk detailnya, lihat Mengaktifkan akses utama IAM ke klaster Anda di Panduan Pengguna Amazon EKS.

Anda sekarang telah memberikan peran deploy, dan dengan ekstensi tindakan Deploy to Amazon EKS, `system:masters` izin ke klaster Kubernetes Anda.

Langkah 8: Buat dan jalankan alur kerja

Pada langkah ini, Anda membuat alur kerja yang mengambil file sumber Anda, membuatnya menjadi image Docker, dan kemudian menyebarkan gambar ke pod pohon di cluster Amazon EKS Anda.

Alur kerja terdiri dari blok bangunan berikut yang berjalan secara berurutan:

- Pemicu - Pemicu ini memulai alur kerja yang dijalankan secara otomatis saat Anda mendorong perubahan ke repositori sumber Anda. Untuk informasi lebih lanjut tentang menggunakan pemicu, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.
- Tindakan build (`BuildBackend`) — Pada pemicu, aksi membangun image Docker menggunakan Dockerfile dan mendorong gambar ke Amazon ECR. Tindakan build juga memperbarui `$IMAGE_TAG` variabel `$REPOSITORY_URI` dan dalam `deployment.yaml` file dengan nilai yang benar, dan kemudian membuat artefak keluaran file ini dan yang lainnya di `Kubernetes` folder. Dalam tutorial ini, satu-satunya file dalam `Kubernetes` folder adalah `deployment.yaml` tetapi Anda dapat menyertakan lebih banyak file. Artefak digunakan sebagai input untuk tindakan penyebaran, yang berikutnya.

  Untuk informasi selengkapnya tentang tindakan build, lihatMembangun dengan alur kerja.
- Tindakan penerapan (`DeployToEKS`) — Setelah menyelesaikan aksi build, tindakan deploy mencari artefak keluaran yang dihasilkan oleh build action (`Manifests`), dan menemukan `deployment.yaml` file di dalamnya. Tindakan kemudian mengikuti instruksi dalam `deployment.yaml` file untuk menjalankan tiga pod—masing-masing berisi satu 'Hello, World! ' Kontainer Docker—di dalam kluster Amazon EKS Anda.

Untuk membuat alur kerja

1. Pergi ke CodeCatalyst konsol.
2. Arahkan ke proyek Anda (`codecatalyst-eks-project`).
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih Buat alur kerja.
5. Untuk repositori Sumber, pilih. `codecatalyst-eks-source-repository`
6. Untuk Cabang, pilih`main`.
7. Pilih Buat.
8. Hapus kode sampel YAMAL.
9. Tambahkan kode YAMAL berikut untuk membuat file definisi alur kerja baru:

> **Note**
>
> Untuk informasi selengkapnya tentang file definisi alur kerja, lihatAlur kerja definisi YAMAL.

```
Name: codecatalyst-eks-workflow
SchemaVersion: 1.0

Triggers:
  - Type: PUSH
    Branches:
      - main
Actions:
  BuildBackend:
    Identifier: aws/build@v1
    Environment:
      Name: codecatalyst-eks-environment
      Connections:
        - Name: codecatalyst-account-connection
          Role: codecatalyst-eks-build-role
    Inputs:
      Sources:
        - WorkflowSource
      Variables:
        - Name: REPOSITORY_URI
          Value: 111122223333.dkr.ecr.us-west-2.amazonaws.com/codecatalyst-eks-
image-repo
        - Name: IMAGE_TAG
          Value: ${WorkflowSource.CommitId}
    Configuration:
      Steps:
        #pre_build:
        - Run: echo Logging in to Amazon ECR...
        - Run: aws --version
        - Run: aws ecr get-login-password --region us-west-2 | docker login --
username AWS --password-stdin 111122223333.dkr.ecr.us-west-2.amazonaws.com
        #build:
        - Run: echo Build started on `date`
        - Run: echo Building the Docker image...
        - Run: docker build -t $REPOSITORY_URI:latest .
```

```
        - Run: docker tag $REPOSITORY_URI:latest $REPOSITORY_URI:$IMAGE_TAG
        #post_build:
        - Run: echo Build completed on `date`
        - Run: echo Pushing the Docker images...
        - Run: docker push $REPOSITORY_URI:latest
        - Run: docker push $REPOSITORY_URI:$IMAGE_TAG
        # Replace the variables in deployment.yaml
        - Run: find Kubernetes/ -type f | xargs sed -i "s|\$REPOSITORY_URI|
$REPOSITORY_URI|g"
        - Run: find Kubernetes/ -type f | xargs sed -i "s|\$IMAGE_TAG|$IMAGE_TAG|g"
        - Run: cat Kubernetes/*
        # The output artifact will be a zip file that contains Kubernetes manifest
 files.
    Outputs:
      Artifacts:
        - Name: Manifests
          Files:
            - "Kubernetes/*"
  DeployToEKS:
    DependsOn:
      - BuildBackend
    Identifier: aws/kubernetes-deploy@v1
    Environment:
      Name: codecatalyst-eks-environment
      Connections:
        - Name: codecatalyst-account-connection
          Role: codecatalyst-eks-deploy-role
    Inputs:
      Artifacts:
        - Manifests
    Configuration:
      Namespace: default
      Region: us-west-2
      Cluster: codecatalyst-eks-cluster
      Manifests: Kubernetes/
```

Pada kode sebelumnya, ganti:

- Kedua contoh *codecatalyst-eks-environment*dengan nama lingkungan yang Anda buat. Prasyarat

- Kedua contoh *codecatalyst-account-connection*dengan nama tampilan koneksi akun Anda. Nama tampilan mungkin nomor. Untuk informasi selengkapnya, lihat Langkah 6: Tambahkan AWS peran ke CodeCatalyst.
- *codecatalyst-eks-build-role*dengan nama peran build yang Anda buatLangkah 5: Buat AWS peran.
- *111122223333.dkr. ecr.us-west-2.amazonaws.com/codecatalyst-eks-image-repo*(di propertiValue:) dengan URI repositori Amazon ECR yang Anda buat. Langkah 3: Buat repositori gambar Amazon ECR
- *111122223333.dkr. ecr.us-west-2.amazonaws.com* (dalam perintahRun: aws ecr) dengan URI repositori Amazon ECR tanpa akhiran gambar (). /codecatalyst-eks-image-repo
- *codecatalyst-eks-deploy-role*dengan nama peran penerapan yang Anda buat. Langkah 5: Buat AWS peran
- Kedua instance *us-west-2* dengan kode Wilayah Anda. AWS Untuk daftar kode Wilayah, lihat Titik akhir Regional di. Referensi Umum AWS

> **Note**
>
> Jika Anda memutuskan untuk tidak membuat peran build dan deploy, ganti *codecatalyst-eks-build-role*dan *codecatalyst-eks-deploy-role*dengan nama CodeCatalystWorkflowDevelopmentRole-*spaceName* peran tersebut. Untuk informasi selengkapnya tentang peran ini, silakan lihat Langkah 5: Buat AWS peran.

10. (Opsional) Pilih Validasi untuk memastikan bahwa kode YAMG valid sebelum melakukan.
11. Pilih Terapkan.
12. Dalam kotak dialog Commit workflow, masukkan yang berikut ini:

    a. Untuk pesan Commit, hapus teks dan masukkan:

    ```
    Add first workflow
    ```

    b. Untuk Repositori, pilih. codecatalyst-eks-source-repository

    c. Untuk nama cabang, pilih main.

    d. Pilih Terapkan.

Anda sekarang telah membuat alur kerja. Jalankan alur kerja dimulai secara otomatis karena pemicu yang ditentukan di bagian atas alur kerja. Khususnya, ketika Anda melakukan (dan mendorong) `workflow.yaml` file ke repositori sumber Anda, pemicu memulai alur kerja dijalankan.

Untuk melihat alur kerja, jalankan kemajuan

1. Di panel navigasi CodeCatalyst konsol, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih alur kerja yang baru saja Anda buat,`codecatalyst-eks-workflow`.
3. Pilih BuildBackenduntuk melihat kemajuan pembangunan.
4. Pilih DeployToEKS untuk melihat kemajuan penerapan.

   Untuk informasi selengkapnya tentang melihat detail run, lihatMelihat alur kerja menjalankan status dan detail.

Untuk memverifikasi penyebaran

1. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.
2. Di sebelah kiri, dekat bagian bawah, pilih Load Balancers.
3. Pilih load balancer yang dibuat sebagai bagian dari penerapan Kubernetes Anda. Jika Anda tidak yakin penyeimbang beban mana yang harus dipilih, cari tag berikut di bawah tab Tag:

   - `kubernetes.io/service-name`
   - `kubernetes.io/cluster/ekstutorialcluster`
4. Dengan penyeimbang beban yang benar dipilih, pilih tab Deskripsi.
5. Salin dan tempel nilai nama DNS ke bilah alamat browser Anda.

   'Halo, Dunia! ' halaman web muncul di browser Anda, menunjukkan bahwa Anda berhasil menerapkan aplikasi Anda.

Langkah 9: Buat perubahan pada file sumber Anda

Di bagian ini, Anda membuat perubahan pada `index.html` file di repositori sumber Anda. Perubahan ini menyebabkan alur kerja membuat image Docker baru, menandainya dengan ID komit, mendorongnya ke Amazon ECR, dan menerapkannya ke Amazon ECS.

Untuk mengubah index.html

1. Pergi ke Lingkungan Pengembang Anda.
2. Pada prompt terminal, ubah ke repositori sumber Anda:

   ```
   cd /projects/codecatalyst-eks-source-repository
   ```

3. Tarik perubahan alur kerja terbaru:

   ```
   git pull
   ```

4. Buka `codecatalyst-eks-source-repository/public-html/index.html`.
5. Pada baris 14, ubah `Hello, World!` teks menjadi`Tutorial complete!`.
6. Tambahkan, komit, dan dorong:

   ```
   git add .
   git commit -m "update index.html title"
   git push
   ```

   Jalankan alur kerja dimulai secara otomatis.
7. (Opsional) Masukkan:

   ```
   git show HEAD
   ```

   Perhatikan ID komit untuk `index.html` perubahan tersebut. ID komit ini akan ditandai ke image Docker yang akan digunakan oleh alur kerja yang baru saja Anda mulai.
8. Perhatikan kemajuan penerapan:

   a. Di CodeCatalyst konsol, di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.

   b. Pilih `codecatalyst-eks-workflow` untuk melihat proses terbaru.

   c. Pilih BuildBackend, dan DeployToEKS untuk melihat alur kerja berjalan kemajuan.
9. Verifikasi bahwa aplikasi Anda telah diperbarui, sebagai berikut:

   a. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.

   b. Di sebelah kiri, dekat bagian bawah, pilih Load Balancers.

   c. Pilih load balancer yang dibuat sebagai bagian dari penerapan Kubernetes Anda.

   d. Salin dan tempel nilai nama DNS ke bilah alamat browser Anda.

'Tutorial Lengkap! ' halaman web muncul di browser Anda, menunjukkan bahwa Anda berhasil menerapkan revisi baru aplikasi Anda.

10. (Opsional) Di AWS, alihkan ke konsol Amazon ECR dan verifikasi bahwa image Docker baru ditandai dengan ID komit dari langkah 7 prosedur ini.

Bersihkan

Anda harus membersihkan lingkungan Anda sehingga Anda tidak dikenakan biaya yang tidak perlu untuk penyimpanan dan sumber daya komputasi yang digunakan oleh tutorial ini.

Untuk membersihkan

1. Hapus klaster Anda:

   - Di terminal Dev Environment, masukkan:

     ```
     eksctl delete cluster --region=us-west-2 --name=codecatalyst-eks-cluster
     ```

     Di mana:

     - *us-west-2* diganti dengan Wilayah Anda.
     - *codecatalyst-eks-cluster*diganti dengan nama cluster yang Anda buat.

     Setelah 5-10 menit, cluster dan sumber daya terkait dihapus, termasuk namun tidak terbatas pada AWS CloudFormation tumpukan, grup node (di Amazon EC2), dan penyeimbang beban.

> **Important**
>
> Jika `eksctl delete cluster` perintah tidak berfungsi, Anda mungkin perlu me-refresh kredensyal atau AWS kredensyal Andakubectl. Jika Anda tidak yakin kredensyal mana yang akan disegarkan, segarkan kredensialnya terlebih AWS dahulu. Untuk menyegarkan AWS kredensyal Anda, lihat. Bagaimana cara memperbaiki kesalahan “Tidak dapat menemukan kredensil” dan “ExpiredToken”? Untuk menyegarkan `kubectl` kredensyal Anda, lihat. Bagaimana cara memperbaiki kesalahan “Tidak dapat terhubung ke server”?

2. Di AWS konsol, bersihkan sebagai berikut:

    1. Di Amazon ECR, hapus`codecatalyst-eks-image-repo`.
    2. Di Pusat Identitas IAM, hapus:
        a. `codecatalyst-eks-user`
        b. `codecatalyst-eks-permission-set`
    3. Di IAM, hapus:
        - `codecatalyst-eks-build-role`
        - `codecatalyst-eks-deploy-role`
        - `codecatalyst-eks-build-policy`
        - `codecatalyst-eks-deploy-policy`
3. Di CodeCatalyst konsol, bersihkan sebagai berikut:

    1. Hapus`codecatalyst-eks-workflow`.
    2. Hapus`codecatalyst-eks-environment`.
    3. Hapus`codecatalyst-eks-source-repository`.
    4. Hapus Lingkungan Pengembang Anda.
    5. Hapus`codecatalyst-eks-project`.

Dalam tutorial ini, Anda mempelajari cara menerapkan aplikasi ke layanan Amazon EKS menggunakan CodeCatalyst alur kerja dan tindakan cluster Deploy to Kubernetes.

Menambahkan aksi “Deploy to Kubernetes cluster”

Gunakan instruksi berikut untuk menambahkan tindakan cluster Deploy ke Kubernetes ke alur kerja Anda.

Sebelum Anda memulai

Sebelum Anda menambahkan aksi klaster Deploy ke Kubernetes ke alur kerja Anda, Anda harus menyiapkan hal-hal berikut:

> **Tip**
>
> Untuk mengatur prasyarat ini dengan cepat, ikuti instruksi di. Tutorial: Menyebarkan aplikasi ke Amazon EKS

- Cluster Kubernetes di Amazon EKS. Untuk informasi tentang cluster, lihat kluster Amazon EKS di Panduan Pengguna Amazon EKS.
- Setidaknya satu Dockerfile yang menjelaskan cara merakit aplikasi Anda menjadi image Docker. Untuk informasi selengkapnya tentang Dockerfiles, lihat referensi Dockerfile.
- Setidaknya satu file manifes Kubernetes, yang disebut file konfigurasi atau konfigurasi dalam dokumentasi Kubernetes. Untuk informasi selengkapnya, lihat Mengelola sumber daya dalam dokumentasi Kubernetes.
- Peran IAM yang memberikan tindakan cluster Deploy to Kubernetes kemampuan untuk mengakses dan berinteraksi dengan cluster Amazon EKS Anda. Untuk informasi lebih lanjut, lihat Role topik diTindakan “Terapkan ke Kluster Kubernetes” Definisi YAMAL.

  Setelah membuat peran ini, Anda harus menambahkannya ke:

  - File Kubernetes ConfigMap Anda. Untuk mempelajari cara menambahkan peran ke ConfigMap file, lihat Mengaktifkan akses utama IAM ke klaster Anda di Panduan Pengguna Amazon EKS.
  - CodeCatalyst. Untuk mempelajari cara menambahkan peran IAM CodeCatalyst, lihatMenambahkan peran IAM ke koneksi akun.
- CodeCatalyst Ruang, proyek, dan lingkungan. Ruang dan lingkungan keduanya harus terhubung ke AWS akun tempat Anda akan menggunakan aplikasi Anda. Lihat informasi selengkapnya di Menciptakan ruang, Membuat proyek kosong di Amazon CodeCatalyst, dan Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst.
- Repositori sumber yang didukung oleh. CodeCatalyst Repositori menyimpan file sumber aplikasi Anda, Dockerfiles, dan manifes Kubernetes. Untuk informasi selengkapnya, lihat Simpan dan berkolaborasi pada kode dengan repositori sumber di CodeCatalyst.

Visual

Untuk menambahkan aksi “Deploy to Kubernetes cluster” menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.

7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari aksi cluster Deploy to Kubernetes, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih Deploy ke klaster Kubernetes. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Unduh untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Pada tab Input dan Konfigurasi, lengkapi bidang sesuai dengan kebutuhan Anda. Untuk deskripsi setiap bidang, lihatTindakan “Terapkan ke Kluster Kubernetes” Definisi YAMAL. Referensi ini memberikan informasi rinci tentang setiap bidang (dan nilai properti YAMAL yang sesuai) seperti yang muncul di YAMAL dan editor visual.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menambahkan aksi “Deploy to Kubernetes cluster” menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari aksi cluster Deploy to Kubernetes, dan lakukan salah satu hal berikut:

- Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

  Atau

- Pilih Deploy ke klaster Kubernetes. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
  - (Opsional) Pilih Unduh untuk melihat kode sumber tindakan.
  - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

10. Ubah properti dalam kode YAMAL sesuai dengan kebutuhan Anda. Penjelasan tentang setiap properti yang tersedia disediakan diTindakan “Terapkan ke Kluster Kubernetes” Definisi YAMAL.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Variabel yang dihasilkan oleh aksi “Deploy to Kubernetes cluster”

Tindakan klaster Deploy to Kubernetes menghasilkan dan menetapkan variabel-variabel berikut pada waktu proses. Ini dikenal sebagai variabel yang telah ditentukan.

Untuk informasi tentang mereferensikan variabel-variabel ini dalam alur kerja, lihat. Menggunakan variabel yang telah ditentukan

Kunci	Nilai
cluster	Nama Sumber Daya Amazon.com (ARN) dari kluster Amazon EKS yang digunakan selama alur kerja dijalankan. Contoh: `arn:aws:eks:us-west-2:111122223333:cluster/codecatalyst-eks-cluster`
platform penyebaran	Nama platform penyebaran. Hardcode ke. `AWS:EKS`

Kunci	Nilai
Metadata	Dicadangkan. Metadata berformat JSON yang terkait dengan cluster yang digunakan selama alur kerja dijalankan.
namespace	Namespace Kubernetes tempat cluster digunakan. Contoh: `default`
sumber daya	Dicadangkan. Metadata berformat JSON terkait dengan sumber daya yang digunakan selama alur kerja dijalankan.
server	Nama endpoint server API yang dapat Anda gunakan untuk berkomunikasi dengan cluster Anda menggunakan alat manajemen seperti`kubectl`. Untuk informasi selengkapnya tentang titik akhir layanan API, lihat kontrol akses titik akhir klaster Amazon EKS di Panduan Pengguna Amazon EKS. Contoh: `https://random-string .gr7.us-west-2.eks.amazonaws.com`

Tindakan “Terapkan ke Kluster Kubernetes” Definisi YAMAL

Berikut ini adalah definisi YAMAL dari aksi cluster Deploy to Kubernetes. Untuk mempelajari cara menggunakan tindakan ini, lihatMenerapkan aplikasi ke Amazon Elastic Kubernetes Service dengan alur kerja.

Definisi tindakan ini ada sebagai bagian dalam file definisi alur kerja yang lebih luas. Untuk informasi selengkapnya tentang file ini, lihatAlur kerja definisi YAMAL.

> **Note**
>
> Sebagian besar properti YAMM yang mengikuti memiliki elemen UI yang sesuai di editor visual. Untuk mencari elemen UI, gunakan Ctrl+F. Elemen akan terdaftar dengan properti YAMLnya yang terkait.

```
# The workflow definition starts here.
# See Properti tingkat atas for details.

Name: MyWorkflow
SchemaVersion: 1.0
Actions:

# The action definition starts here.
  DeployToKubernetesCluster_nn:
    Identifier: aws/kubernetes-deploy@v1
    DependsOn:
      - build-action
    Compute:
        - Type: EC2 | Lambda
        - Fleet: fleet-name
    Timeout: timeout-minutes
    Environment:
      Name: environment-name
      Connections:
        - Name: account-connection-name
          Role: DeployToEKS
    Inputs:
      # Specify a source or an artifact, but not both.
      Sources:
        - source-name-1
      Artifacts:
        - manifest-artifact
    Configuration:
      Namespace: namespace
      Region: us-east-1
      Cluster: eks-cluster
      Manifests: manifest-path
```

DeployToKubernetesCluster

(Diperlukan)

Tentukan nama tindakan. Semua nama tindakan harus unik dalam alur kerja. Nama aksi terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama tindakan.

Default: `DeployToKubernetesCluster_nn`.

UI yang sesuai: Tab konfigurasi/Nama tampilan tindakan

Identifier

(*`DeployToKubernetesCluster`*/Identifier)

(Diperlukan)

Mengidentifikasi tindakan. Jangan mengubah properti ini kecuali Anda ingin mengubah versi. Untuk informasi selengkapnya, lihat Menentukan versi mayor, minor, atau patch dari suatu tindakan.

Default: `aws/kubernetes-deploy@v1`.

UI yang sesuai: Diagram alur kerja/ DeployToKubernetesCluster _nn/ aws/kubernetes-deploy @v1 label

DependsOn

(*`DeployToKubernetesCluster`*/DependsOn)

(Opsional)

Tentukan tindakan, grup tindakan, atau gerbang yang harus berjalan dengan sukses agar tindakan ini berjalan.

Untuk informasi selengkapnya tentang fungsionalitas 'tergantung pada', lihat. Mengkonfigurasi tindakan untuk bergantung pada tindakan lain

UI yang sesuai: Tab masukan/Tergantung pada - opsional

Compute

(*`DeployToKubernetesCluster`*/Compute)

(Opsional)

Mesin komputasi yang digunakan untuk menjalankan tindakan alur kerja Anda. Anda dapat menentukan komputasi baik di tingkat alur kerja atau di tingkat tindakan, tetapi tidak keduanya. Ketika ditentukan pada tingkat alur kerja, konfigurasi komputasi berlaku untuk semua tindakan yang ditentukan dalam alur kerja. Pada tingkat alur kerja, Anda juga dapat menjalankan beberapa tindakan pada instance yang sama. Untuk informasi selengkapnya, lihat Berbagi komputasi di seluruh tindakan.

UI yang sesuai: tidak ada

Type

(*DeployToKubernetesCluster*/Compute/Type)

(Diperlukan Compute jika disertakan)

Jenis mesin komputasi. Anda dapat menggunakan salah satu nilai berikut:

- EC2 (editor visual) atau `EC2` (editor YAMG)

  Dioptimalkan untuk fleksibilitas selama aksi berjalan.
- Lambda (editor visual) atau `Lambda` (editor YAMG)

  Kecepatan start-up aksi yang dioptimalkan.

Untuk informasi selengkapnya tentang jenis komputasi, lihatJenis komputasi.

UI yang sesuai: Tab Konfigurasi/Tingkat Lanjut - opsional/Jenis komputasi

Fleet

(*DeployToKubernetesCluster*/Compute/Fleet)

(Opsional)

Tentukan mesin atau armada yang akan menjalankan alur kerja atau tindakan alur kerja Anda. Dengan armada sesuai permintaan, ketika suatu tindakan dimulai, alur kerja menyediakan sumber daya yang dibutuhkan, dan mesin dihancurkan ketika tindakan selesai. Contoh armada sesuai permintaan:`Linux.x86-64.Large`,. `Linux.x86-64.XLarge` Untuk informasi lebih lanjut tentang armada sesuai permintaan, lihat. Properti armada sesuai permintaan

Dengan armada yang disediakan, Anda mengonfigurasi satu set mesin khusus untuk menjalankan tindakan alur kerja Anda. Mesin-mesin ini tetap menganggur, siap untuk memproses tindakan segera. Untuk informasi lebih lanjut tentang armada yang disediakan, lihat. Properti armada yang disediakan

Jika `Fleet` dihilangkan, defaultnya adalah. `Linux.x86-64.Large`

UI yang sesuai: Tab Konfigurasi/Advanced - armada opsional/Komputasi

Timeout

(*`DeployToKubernetesCluster`*/Timeout)

(Opsional)

Tentukan jumlah waktu dalam menit (editor YAMM), atau jam dan menit (editor visual), bahwa tindakan dapat berjalan sebelum CodeCatalyst mengakhiri tindakan. Minimal adalah 5 menit dan maksimum dijelaskan dalamKuota untuk alur kerja. Batas waktu default sama dengan batas waktu maksimum.

UI yang sesuai: Tab konfigurasi/Timeout - opsional

Environment

(*`DeployToKubernetesCluster`*/Environment)

(Diperlukan)

Tentukan CodeCatalyst lingkungan yang akan digunakan dengan tindakan.

Untuk informasi lebih lanjut tentang lingkungan, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst danPembuatan lingkungan.

UI yang sesuai: Tab Konfigurasi/'Lingkungan/Akun/Peran'/Lingkungan

Name

(*`DeployToKubernetesCluster`*/Environment/Name)

(Diperlukan Environment jika disertakan)

Tentukan nama lingkungan yang ada yang ingin Anda kaitkan dengan tindakan.

UI yang sesuai: Tab Konfigurasi/'Lingkungan/Akun/Peran'/Lingkungan

Connections

(*DeployToKubernetesCluster*/Environment/Connections)

(Diperlukan Environment jika disertakan)

Tentukan koneksi akun untuk dikaitkan dengan tindakan. Anda dapat menentukan maksimum satu koneksi akun di bawah`Environment`.

Untuk informasi selengkapnya tentang koneksi akun, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS. Untuk informasi tentang cara mengaitkan koneksi akun dengan lingkungan Anda, lihatPembuatan lingkungan.

UI yang sesuai: Tab konfigurasi/'lingkungan/akun/peran'/koneksi akun AWS

Name

(*DeployToKubernetesCluster*/Environment/Connections/Name)

(Diperlukan)

Tentukan nama koneksi akun.

UI yang sesuai: Tab konfigurasi/'lingkungan/akun/peran'/koneksi akun AWS

Role

(*DeployToKubernetesCluster*/Environment/Connections/Role)

(Diperlukan)

Tentukan nama peran IAM yang digunakan tindakan klaster Deploy to Kubernetes untuk mengakses. AWS Pastikan bahwa peran ini mencakup kebijakan berikut:

- Kebijakan izin berikut:

> ⚠ Warning
>
> Batasi izin untuk yang ditampilkan dalam kebijakan berikut. Menggunakan peran dengan izin yang lebih luas dapat menimbulkan risiko keamanan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:DescribeCluster",
                "eks:ListClusters"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan, gunakan wildcard berikut dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

- Kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                 ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Pastikan bahwa peran ini ditambahkan ke:

- Koneksi akun Anda. Untuk mempelajari selengkapnya tentang menambahkan peran IAM ke koneksi akun, lihatMenambahkan peran IAM ke koneksi akun.
- Kubernetes ConfigMap Anda. Untuk mempelajari selengkapnya tentang menambahkan peran IAM ke sebuah ConfigMap, lihat Mengelola pengguna dan peran IAM dalam dokumentasi. `eksctl`

> **Tip**
>
> Lihat juga Tutorial: Menyebarkan aplikasi ke Amazon EKS untuk petunjuk tentang menambahkan peran AM IAM ke koneksi akun dan ConfigMap.

> **Note**
>
> Anda dapat menentukan nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran di sini, jika Anda mau. Untuk informasi selengkapnya tentang peran ini, silakan lihat Membuat CodeCatalystWorkflowDevelopmentRole-***`spaceName`***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami menyarankan Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian. Jika Anda memutuskan untuk menggunakan `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran, pastikan untuk menambahkannya ke ConfigMap file Anda, mengikuti petunjuk di Kelola pengguna IAM dan peran dalam `eksctl` dokumentasi.

UI yang sesuai: Tab Konfigurasi/'lingkungan/akun/peran'/ Peran

Inputs

(*`DeployToKubernetesCluster`*/Inputs)

(Opsional)

`Inputs`Bagian ini mendefinisikan data yang `DeployToKubernetesCluster` dibutuhkan selama menjalankan alur kerja.

> **Note**
>
> Hanya satu input (baik sumber atau artefak) yang diizinkan per tindakan Deploy ke Amazon EKS.

UI yang sesuai: Tab input

Sources

(*DeployToKubernetesCluster*/Inputs/Sources)

(Diperlukan jika file manifes Anda disimpan dalam repositori sumber)

Jika file manifes Kubernetes atau file disimpan dalam repositori sumber, tentukan label repositori sumber tersebut. Saat ini, satu-satunya label yang didukung adalah`WorkflowSource`.

Jika file manifes Anda tidak terkandung dalam repositori sumber, mereka harus berada dalam artefak yang dihasilkan oleh tindakan lain.

Untuk informasi selengkapnya tentang sumber, lihat Menghubungkan alur kerja ke repositori sumber.

UI yang sesuai: Tab/Sumber Input - opsional

Artifacts - input

(*DeployToKubernetesCluster*/Inputs/Artifacts)

(Diperlukan jika file manifes Anda disimpan dalam artefak keluaran dari tindakan sebelumnya)

Jika file atau file manifes Kubernetes terkandung dalam artefak yang dihasilkan oleh tindakan sebelumnya, tentukan artefak tersebut di sini. Jika file manifes Anda tidak terkandung dalam artefak, mereka harus berada di repositori sumber Anda.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab konfigurasi/Artefak - opsional

Configuration

(*DeployToKubernetesCluster*/Configuration)

(Diperlukan)

Bagian di mana Anda dapat menentukan properti konfigurasi tindakan.

UI yang sesuai: Tab konfigurasi

Namespace

(*DeployToKubernetesCluster*/Configuration/Namespace)

(Opsional)

Tentukan namespace Kubernetes tempat aplikasi Kubernetes Anda akan di-deploy. Gunakan `default` jika Anda tidak menggunakan ruang nama dengan cluster Anda. Untuk informasi selengkapnya tentang namespace, lihat Membagi klaster Anda menggunakan ruang nama Kubernetes dalam dokumentasi Kubernetes.

Jika Anda menghilangkan namespace, nilai digunakan. `default`

UI yang sesuai: Tab konfigurasi/Namespace

Region

(*DeployToKubernetesCluster*/Configuration/Region)

(Diperlukan)

Tentukan AWS Wilayah tempat klaster dan layanan Amazon EKS Anda berada. Untuk daftar kode Region, lihat Titik akhir Regional di. Referensi Umum AWS

UI yang sesuai: Tab konfigurasi/Wilayah

Cluster

(*DeployToKubernetesCluster*/Configuration/Cluster)

(Diperlukan)

Tentukan nama kluster Amazon EKS yang ada. Tindakan klaster Deploy to Kubernetes akan menyebarkan aplikasi kontainer Anda ke dalam klaster ini. Untuk informasi selengkapnya tentang kluster Amazon EKS, lihat Cluster di Panduan Pengguna Amazon EKS.

UI yang sesuai: Tab konfigurasi/Cluster

Manifests

(*DeployToKubernetesCluster*/Configuration/Manifests)

(Diperlukan)

Tentukan path ke file manifes Kubernetes Anda yang diformat YAML, yang disebut file konfigurasi, file konfigurasi, atau hanya, konfigurasi dalam dokumentasi Kubernetes.

Jika Anda menggunakan beberapa file manifes, letakkan di satu folder dan rujuk folder itu. File manifes diproses secara alfanumerik oleh Kubernetes, jadi pastikan untuk mengawali nama file dengan angka atau huruf yang bertambah untuk mengontrol urutan pemrosesan. Sebagai contoh:

```
00-namespace.yaml
```

```
01-deployment.yaml
```

Jika file manifes Anda berada di repositori sumber Anda, jalurnya relatif terhadap folder root repositori sumber. Jika file berada dalam artefak dari tindakan alur kerja sebelumnya, jalurnya relatif terhadap folder root artefak.

Contoh:

```
Manifests/
```

```
deployment.yaml
```

```
my-deployment.yml
```

Jangan gunakan wildcard (`*`).

> **Note**
>
> Bagan helm dan file kustomisasi tidak didukung.

Untuk informasi selengkapnya tentang file manifes, lihat Mengatur konfigurasi sumber daya dalam dokumentasi Kubernetes.

UI yang sesuai: Tab konfigurasi/Manifes

Menerapkan AWS CloudFormation tumpukan dengan alur kerja

Bagian ini menjelaskan cara menerapkan AWS CloudFormation tumpukan menggunakan CodeCatalyst alur kerja. Untuk mencapai ini, Anda harus menambahkan tindakan AWS CloudFormation tumpukan Deploy ke alur kerja Anda. Tindakan ini menyebarkan CloudFormation tumpukan sumber daya AWS berdasarkan templat yang Anda berikan. Template dapat berupa:

- AWS CloudFormation template — Untuk informasi selengkapnya, lihat Bekerja dengan AWS CloudFormation template.
- AWS SAM template — Untuk informasi lebih lanjut, lihat AWS Serverless Application Model (AWS SAM) spesifikasi.

> **Note**
>
> Untuk menggunakan AWS SAM template, Anda harus terlebih dahulu mengemas AWS SAM aplikasi Anda menggunakan `sam package` operasi. Untuk tutorial yang menunjukkan cara melakukan pengemasan ini secara otomatis sebagai bagian dari CodeCatalyst alur kerja Amazon, lihatTutorial: Menyebarkan aplikasi tanpa server menggunakan AWS CloudFormation.

Jika tumpukan sudah ada, tindakan menjalankan CloudFormation `CreateChangeSet` operasi, dan kemudian `ExecuteChangeSet` operasi. Tindakan kemudian menunggu perubahan diterapkan dan menandai dirinya sebagai berhasil karena gagal, tergantung pada hasilnya.

Gunakan tindakan AWS CloudFormation tumpukan Deploy jika Anda sudah memiliki AWS CloudFormation atau AWS SAM template yang berisi sumber daya yang ingin Anda terapkan, atau Anda berencana untuk membuatnya secara otomatis sebagai bagian dari tindakan pembuatan alur kerja menggunakan alat seperti dan. AWS SAM AWS Cloud Development Kit (AWS CDK)

Tidak ada batasan pada template yang dapat Anda gunakan—apa pun yang dapat Anda buat CloudFormation atau AWS SAM Anda dapat gunakan dengan tindakan tumpukan Deploy AWS CloudFormation .

> **Tip**
>
> Untuk tutorial yang menunjukkan cara menerapkan aplikasi tanpa server menggunakan tindakan AWS CloudFormation tumpukan Deploy, lihat. Tutorial: Menyebarkan aplikasi tanpa server menggunakan AWS CloudFormation

Topik

- Tutorial: Menyebarkan aplikasi tanpa server menggunakan AWS CloudFormation
- Menambahkan tindakan “Deploy AWS CloudFormation stack”
- Mengkonfigurasi rollback

- Variabel yang dihasilkan oleh tindakan “Deploy AWS CloudFormation stack”
- Tindakan “Terapkan AWS CloudFormation tumpukan” definisi YAMAL

Tutorial: Menyebarkan aplikasi tanpa server menggunakan AWS CloudFormation

Dalam tutorial ini, Anda mempelajari cara membangun, menguji, dan menyebarkan aplikasi tanpa server sebagai CloudFormation tumpukan menggunakan alur kerja.

Aplikasi dalam tutorial ini adalah aplikasi web sederhana yang mengeluarkan pesan “Hello World”. Ini terdiri dari AWS Lambda fungsi dan Amazon API Gateway, dan Anda membangunnya menggunakan AWS Serverless Application Model (AWS SAM), yang merupakan ekstensi dari AWS CloudFormation.

Topik

- Prasyarat
- Langkah 1: Buat repositori sumber
- Langkah 2: Buat AWS peran
- Langkah 3: Tambahkan AWS peran ke CodeCatalyst
- Langkah 4: Buat ember Amazon S3
- Langkah 5: Tambahkan file sumber
- Langkah 6: Buat dan jalankan alur kerja
- Langkah 7: Buat perubahan
- Bersihkan

Prasyarat

Sebelum Anda memulai:

- Anda membutuhkan CodeCatalyst ruang dengan AWS akun yang terhubung. Untuk informasi selengkapnya, lihat Menciptakan ruang.
- Di ruang Anda, Anda memerlukan CodeCatalyst proyek kosong, Mulai dari awal yang disebut:

  ```
  codecatalyst-cfn-project
  ```

  Untuk informasi selengkapnya, lihat Membuat proyek kosong di Amazon CodeCatalyst.

- Dalam proyek Anda, Anda memerlukan CodeCatalyst lingkungan yang disebut:

  ```
  codecatalyst-cfn-environment
  ```

  Konfigurasikan lingkungan ini sebagai berikut:

  - Pilih jenis apa pun, seperti Non-produksi.
  - Hubungkan AWS akun Anda ke sana.

  Untuk informasi selengkapnya, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst.

Langkah 1: Buat repositori sumber

Pada langkah ini, Anda membuat repositori sumber di. CodeCatalyst Repositori ini digunakan untuk menyimpan file sumber tutorial, seperti file fungsi Lambda.

Untuk informasi selengkapnya tentang repositori sumber, lihat. Membuat repositori sumber

Untuk membuat repositori sumber

1. Di CodeCatalyst, di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
2. Pilih Tambahkan repositori, lalu pilih Buat repositori.
3. Dalam nama Repositori, masukkan:

   ```
   codecatalyst-cfn-source-repository
   ```

4. Pilih Buat.

Anda sekarang telah membuat repositori yang disebut. `codecatalyst-cfn-source-repository`

Langkah 2: Buat AWS peran

Pada langkah ini, Anda membuat peran AWS IAM berikut:

- Menyebarkan peran — Memberikan izin tindakan AWS CloudFormation tumpukan CodeCatalyst Deploy untuk mengakses AWS akun dan CloudFormation layanan tempat Anda akan menerapkan aplikasi tanpa server. Tindakan AWS CloudFormation tumpukan Deploy adalah bagian dari alur kerja Anda.

- Peran build - Memberikan izin tindakan CodeCatalyst build untuk mengakses AWS akun Anda dan menulis ke Amazon S3 tempat paket aplikasi tanpa server Anda akan disimpan. Tindakan build adalah bagian dari alur kerja Anda.
- Peran tumpukan - Memberikan CloudFormation izin untuk membaca dan memodifikasi sumber daya yang ditentukan dalam AWS SAM template yang akan Anda berikan nanti. Juga memberikan izin untuk CloudWatch.

Untuk informasi selengkapnya tentang peran IAM, lihat peran IAM di AWS Identity and Access Management Panduan Pengguna.

> **Note**
>
> Untuk menghemat waktu, Anda dapat membuat satu peran, yang disebut `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran, alih-alih tiga peran yang tercantum sebelumnya. Untuk informasi selengkapnya, lihat Membuat CodeCatalystWorkflowDevelopmentRole-***`spaceName`***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami menyarankan Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian. Tutorial ini mengasumsikan Anda membuat tiga peran yang tercantum sebelumnya.

> **Note**
>
> Peran eksekusi Lambda juga diperlukan, tetapi Anda tidak perlu membuatnya sekarang karena `sam-template.yml` file membuatnya untuk Anda ketika Anda menjalankan alur kerja di langkah 5.

Untuk membuat peran penerapan

1. Buat kebijakan untuk peran tersebut, sebagai berikut:
   a. Masuk ke AWS.
   b. Buka konsol IAM di https://console.aws.amazon.com/iam/.

c. Di panel navigasi, pilih Kebijakan.

d. Pilih Buat kebijakan.

e. Pilih tab JSON.

f. Hapus kode yang ada.

g. Tempel kode berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [{
    "Action": [
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack",
        "cloudformation:Describe*",
        "cloudformation:UpdateStack",
        "cloudformation:CreateChangeSet",
        "cloudformation:DeleteChangeSet",
        "cloudformation:ExecuteChangeSet",
        "cloudformation:SetStackPolicy",
        "cloudformation:ValidateTemplate",
        "cloudformation:List*",
        "iam:PassRole"
    ],
    "Resource": "*",
    "Effect": "Allow"
}]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

h. Pilih Berikutnya: Tanda.

i. Pilih Berikutnya: Tinjau.

j. Dalam Nama, masukkan:

```
codecatalyst-deploy-policy
```

k. Pilih Buat kebijakan.

Anda sekarang telah membuat kebijakan izin.

2. Buat peran deploy, sebagai berikut:

a. Di panel navigasi, pilih Peran, lalu pilih Buat peran.

b. Pilih Kebijakan kepercayaan khusus.

c. Hapus kebijakan kepercayaan kustom yang ada.

d. Tambahkan kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                 ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

e. Pilih Selanjutnya.

f. Di Kebijakan izin, cari `codecatalyst-deploy-policy` dan pilih kotak centang.

g. Pilih Selanjutnya.

h. Untuk nama Peran, masukkan:

```
codecatalyst-deploy-role
```

i. Untuk deskripsi Peran, masukkan:

```
CodeCatalyst deploy role
```

j. Pilih Buat peran.

Anda sekarang telah membuat peran penerapan dengan kebijakan kepercayaan dan kebijakan izin.

3. Dapatkan peran penyebaran ARN, sebagai berikut:

a. Di panel navigasi, pilih Peran.

b. Di kotak pencarian, masukkan nama peran yang baru saja Anda buat (`codecatalyst-deploy-role`).

c. Pilih peran dari daftar.

Halaman Ringkasan peran muncul.

d. Di bagian atas, salin nilai ARN.

Anda sekarang telah membuat peran penerapan dengan izin yang sesuai, dan memperoleh ARN-nya.

Untuk membuat peran build

1. Buat kebijakan untuk peran tersebut, sebagai berikut:

a. Masuk ke AWS.

b. Buka konsol IAM di https://console.aws.amazon.com/iam/.

c. Di panel navigasi, pilih Kebijakan.

d. Pilih Buat kebijakan.

e. Pilih tab JSON.

f. Hapus kode yang ada.

g. Tempel kode berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [{
    "Action": [
```

```
            "s3:PutObject",
            "iam:PassRole"
        ],
        "Resource": "*",
        "Effect": "Allow"
}]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

h. Pilih Berikutnya: Tanda.

i. Pilih Berikutnya: Tinjau.

j. Dalam Nama, masukkan:

```
codecatalyst-build-policy
```

k. Pilih Buat kebijakan.

Anda sekarang telah membuat kebijakan izin.

2. Buat peran build, sebagai berikut:

a. Di panel navigasi, pilih Peran, lalu pilih Buat peran.

b. Pilih Kebijakan kepercayaan khusus.

c. Hapus kebijakan kepercayaan kustom yang ada.

d. Tambahkan kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
```

```
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

e. Pilih Selanjutnya.

f. Di Kebijakan izin, cari `codecatalyst-build-policy` dan pilih kotak centang.

g. Pilih Selanjutnya.

h. Untuk nama Peran, masukkan:

```
codecatalyst-build-role
```

i. Untuk deskripsi Peran, masukkan:

```
CodeCatalyst build role
```

j. Pilih Buat peran.

Anda sekarang telah membuat peran build dengan kebijakan kepercayaan dan kebijakan izin.

3. Dapatkan peran membangun ARN, sebagai berikut:

a. Di panel navigasi, pilih Peran.

b. Di kotak pencarian, masukkan nama peran yang baru saja Anda buat (`codecatalyst-build-role`).

c. Pilih peran dari daftar.

Halaman Ringkasan peran muncul.

d. Di bagian atas, salin nilai ARN.

Anda sekarang telah membuat peran build dengan izin yang sesuai, dan memperoleh ARN-nya.

Untuk membuat peran tumpukan

1. Masuk untuk AWS menggunakan akun tempat Anda ingin menyebarkan tumpukan Anda.
2. Buka konsol IAM di https://console.aws.amazon.com/iam/.
3. Buat peran tumpukan sebagai berikut:

   a. Di panel navigasi, pilih Peran.

   b. Pilih Buat peran.

   c. Pilih Layanan AWS .

   d. Di bagian Use case, pilih CloudFormationdari daftar drop-down.

   e. Pilih tombol CloudFormationradio.

   f. Di bagian bawah, pilih Berikutnya.

   g. Menggunakan kotak pencarian, temukan kebijakan izin berikut, lalu pilih kotak centang masing-masing.

   > **Note**
   >
   > Jika Anda mencari kebijakan dan tidak muncul, pastikan untuk memilih Hapus filter dan coba lagi.

   - CloudWatchFullAccess
   - AWS CloudFormationFullAccess
   - IAM FullAccess
   - AWS Lambda_ FullAccess
   - AmazonAPI GatewayAdministrator
   - AmazonS3 FullAccess
   - AmazonEC2 ContainerRegistryFullAccess

   Kebijakan pertama memungkinkan akses CloudWatch untuk mengaktifkan rollback tumpukan saat alarm terjadi.

   Kebijakan yang tersisa memungkinkan AWS SAM untuk mengakses layanan dan sumber daya di tumpukan yang akan digunakan dalam tutorial ini. Untuk informasi selengkapnya, lihat Izin di Panduan AWS Serverless Application Model Pengembang.

h. Pilih Selanjutnya.

i. Untuk nama Peran, masukkan:

```
codecatalyst-stack-role
```

j. Pilih Buat peran.

4. Dapatkan ARN peran tumpukan, sebagai berikut:

a. Di panel navigasi, pilih Peran.

b. Di kotak pencarian, masukkan nama peran yang baru saja Anda buat (`codecatalyst-stack-role`).

c. Pilih peran dari daftar.

d. Di bagian Ringkasan, salin nilai ARN. Anda membutuhkannya nanti.

Anda sekarang telah membuat peran tumpukan dengan izin yang sesuai, dan Anda telah memperoleh ARN-nya.

Langkah 3: Tambahkan AWS peran ke CodeCatalyst

Pada langkah ini, Anda menambahkan build role (`codecatalyst-build-role`) dan deploy role (`codecatalyst-deploy-role`) ke koneksi CodeCatalyst akun di ruang Anda.

> **Note**
>
> Anda tidak perlu menambahkan stack role (`codecatalyst-stack-role`) ke koneksi. Ini karena peran tumpukan digunakan oleh CloudFormation(not CodeCatalyst), setelah koneksi sudah dibuat antara CodeCatalyst dan AWS menggunakan peran penerapan. Karena peran tumpukan tidak digunakan oleh CodeCatalyst untuk mendapatkan akses ke AWS, itu tidak perlu dikaitkan dengan koneksi akun.

Untuk menambahkan peran build dan deploy ke koneksi akun Anda

1. Masuk CodeCatalyst, navigasikan ke ruang Anda.
2. Pilih AWS akun. Daftar koneksi akun muncul.
3. Pilih koneksi akun yang mewakili AWS akun tempat Anda membuat peran build dan deploy.

4. Pilih Kelola peran dari konsol AWS manajemen.

   Peran Tambahkan IAM ke halaman CodeCatalyst ruang Amazon muncul. Anda mungkin perlu masuk untuk mengakses halaman.

5. Pilih Tambahkan peran yang sudah Anda buat di IAM.

   Daftar drop-down muncul. Daftar ini menampilkan semua peran IAM dengan kebijakan kepercayaan yang mencakup prinsip `codecatalyst-runner.amazonaws.com` dan `codecatalyst.amazonaws.com` layanan.

6. Dalam daftar drop-down, pilih`codecatalyst-build-role`, dan pilih Tambah peran.

7. Pilih Tambahkan peran IAM, pilih Tambahkan peran yang ada yang telah Anda buat di IAM, dan dalam daftar drop-down, pilih. `codecatalyst-deploy-role` Pilih Tambahkan peran.

   Anda sekarang telah menambahkan peran build dan deploy ke ruang Anda.

8. Salin nilai nama CodeCatalyst tampilan Amazon. Anda akan membutuhkan nilai ini nanti, saat membuat alur kerja Anda.

Langkah 4: Buat ember Amazon S3

Pada langkah ini, Anda membuat bucket Amazon S3 tempat Anda menyimpan file paket deployment aplikasi tanpa server .zip Anda.

Untuk membuat bucket Amazon S3

1. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.
2. Di panel utama, pilih Buat ember.
3. Untuk nama Bucket, masukkan:

   ```
   codecatalyst-cfn-s3-bucket
   ```

4. Untuk Wilayah AWS , pilih Wilayah. Tutorial ini mengasumsikan Anda memilih US West (Oregon) us-west-2. Untuk informasi tentang Wilayah yang didukung oleh Amazon S3, lihat titik akhir dan kuota Amazon Simple Storage Service di. Referensi Umum AWS
5. Di bagian bawah halaman, pilih Buat ember.

Anda sekarang telah membuat ember yang disebut **`codecatalyst-cfn-s3-bucket`** di Wilayah AS Barat (Oregon) us-west-2.

Langkah 5: Tambahkan file sumber

Pada langkah ini, Anda menambahkan beberapa file sumber aplikasi ke repositori CodeCatalyst sumber Anda. `hello-world`Folder berisi file aplikasi yang akan Anda gunakan. `tests`Folder berisi tes unit. Struktur folder adalah sebagai berikut:

```
.
|— hello-world
|  |— tests
|     |— unit
|        |— test-handler.js
|  |— app.js
|— .npmignore
|— package.json
|— sam-template.yml
|— setup-sam.sh
```

.npmignore

`.npmignore`File menunjukkan file dan folder npm mana yang harus dikecualikan dari paket aplikasi. Dalam tutorial ini, npm mengecualikan `tests` folder karena bukan bagian dari aplikasi.

Untuk menambahkan file.npmignore

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda, `codecatalyst-cfn-project`
3. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
4. Dari daftar repositori sumber, pilih repositori Anda,. `codecatalyst-cfn-source-repository`
5. Di File, pilih Buat file.
6. Untuk nama File, masukkan:

   ```
   .npmignore
   ```

7. Di kotak teks, masukkan kode berikut:

   ```
   tests/*
   ```

8. Pilih Komit, lalu pilih Komit lagi.

Anda sekarang telah membuat file yang disebut `.npmignore` di root repositori Anda.

berkas package.json

`package.json`File berisi metadata penting tentang proyek Node Anda seperti nama proyek, nomor versi, deskripsi, dependensi, dan detail lainnya yang menjelaskan cara berinteraksi dan menjalankan aplikasi Anda.

`package.json`Dalam tutorial ini mencakup daftar dependensi dan skrip. `test` Skrip pengujian melakukan hal berikut:

- Menggunakan moka, skrip pengujian menjalankan pengujian unit yang ditentukan `hello-world/tests/unit/` dan menulis hasilnya ke `junit.xml` file menggunakan reporter xunit.
- Menggunakan Istanbul (nyc), skrip pengujian menghasilkan laporan cakupan kode (`clover.xml`) menggunakan reporter semanggi. Untuk informasi lebih lanjut, lihat Menggunakan reporter alternatif dalam dokumentasi Istanbul.

Untuk menambahkan file package.json

1. Di repositori Anda, di File, pilih Buat file.
2. Untuk nama File, masukkan:

   ```
   package.json
   ```

3. Di kotak teks, masukkan kode berikut:

   ```
   {
     "name": "hello_world",
     "version": "1.0.0",
     "description": "hello world sample for NodeJS",
     "main": "app.js",
     "repository": "https://github.com/awslabs/aws-sam-cli/tree/develop/samcli/local/
   init/templates/cookiecutter-aws-sam-hello-nodejs",
     "author": "SAM CLI",
     "license": "MIT",
     "dependencies": {
       "axios": "^0.21.1",
       "nyc": "^15.1.0"
     },
   ```

```
  "scripts": {
    "test": "nyc --reporter=clover mocha hello-world/tests/unit/ --reporter xunit
 --reporter-option output=junit.xml"
  },
  "devDependencies": {
    "aws-sdk": "^2.815.0",
    "chai": "^4.2.0",
    "mocha": "^8.2.1"
  }
}
```

4. Pilih Komit, lalu pilih Komit lagi.

   Anda sekarang telah menambahkan file yang dipanggil `package.json` ke root repositori.

file sam-template.yml

`sam-template.yml`File berisi instruksi untuk menerapkan fungsi Lambda dan API Gateway dan mengonfigurasinya bersama-sama. Ini mengikuti spesifikasi AWS Serverless Application Model template, yang memperluas spesifikasi AWS CloudFormation template.

Anda menggunakan AWS SAM template dalam tutorial ini alih-alih AWS CloudFormation template biasa karena AWS SAM menawarkan tipe sumber daya AWS: :Serverless: :Function yang bermanfaat. Tipe ini melakukan banyak behind-the-scenes konfigurasi yang biasanya harus Anda tulis untuk menggunakan CloudFormation sintaks dasar. Misalnya, `AWS::Serverless::Function` membuat fungsi Lambda, peran eksekusi Lambda, dan pemetaan sumber peristiwa yang memulai fungsi. Anda harus mengkodekan semua ini jika Anda ingin menulisnya menggunakan dasar CloudFormation.

Meskipun tutorial ini menggunakan template yang telah ditulis sebelumnya, Anda dapat membuatnya sebagai bagian dari alur kerja Anda menggunakan tindakan build. Untuk informasi selengkapnya, lihat Menerapkan AWS CloudFormation tumpukan dengan alur kerja.

Untuk menambahkan file sam-template.yml.

1. Di repositori Anda, di File, pilih Buat file.
2. Untuk nama File, masukkan:

   ```
   sam-template.yml
   ```

3. Di kotak teks, masukkan kode berikut:

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Description: >
  serverless-api

  Sample SAM Template for serverless-api

# More info about Globals: https://github.com/awslabs/serverless-application-model/
blob/master/docs/globals.rst
Globals:
  Function:
    Timeout: 3

Resources:
  HelloWorldFunction:
    Type: AWS::Serverless::Function # For details on this resource type,
 see https://github.com/awslabs/serverless-application-model/blob/master/
versions/2016-10-31.md#awsserverlessfunction
    Properties:
      CodeUri: hello-world/
      Handler: app.lambdaHandler
      Runtime: nodejs12.x
      Events:
        HelloWorld:
          Type: Api # For details on this event source type, see
 https://github.com/awslabs/serverless-application-model/blob/master/
versions/2016-10-31.md#api
          Properties:
            Path: /hello
            Method: get

Outputs:
  # ServerlessRestApi is an implicit API created out of the events key under
 Serverless::Function
  # Find out about other implicit resources you can reference within AWS SAM at
  # https://github.com/awslabs/serverless-application-model/blob/master/docs/
internals/generated_resources.rst#api
  HelloWorldApi:
    Description: "API Gateway endpoint URL for the Hello World function"
    Value: !Sub "https://${ServerlessRestApi}.execute-api.
${AWS::Region}.amazonaws.com/Prod/hello/"
  HelloWorldFunction:
    Description: "Hello World Lambda function ARN"
```

```
    Value: !GetAtt HelloWorldFunction.Arn
  HelloWorldFunctionIamRole:
    Description: "Implicit Lambda execution role created for the Hello World
 function"
    Value: !GetAtt HelloWorldFunctionRole.Arn
```

4. Pilih Komit, lalu pilih Komit lagi.

   Anda sekarang telah menambahkan file yang disebut `sam-template.yml` di bawah folder root repositori Anda.

setup-sam.sh berkas

`setup-sam.sh`File berisi instruksi untuk mengunduh dan menginstal utilitas AWS SAM CLI. Alur kerja menggunakan utilitas ini untuk mengemas `hello-world` sumber.

Untuk menambahkan file setup-sam.sh

1. Di repositori Anda, di File, pilih Buat file.
2. Untuk nama File, masukkan:

   ```
   setup-sam.sh
   ```

3. Di kotak teks, masukkan kode berikut:

   ```
   #!/usr/bin/env bash
   echo "Setting up sam"

   yum install unzip -y

   curl -LO https://github.com/aws/aws-sam-cli/releases/latest/download/aws-sam-cli-
   linux-x86_64.zip
   unzip -qq aws-sam-cli-linux-x86_64.zip -d sam-installation-directory

   ./sam-installation-directory/install; export AWS_DEFAULT_REGION=us-west-2
   ```

   Pada kode sebelumnya, ganti us-west-2 dengan Region Anda. AWS

4. Pilih Komit, lalu pilih Komit lagi.

   Anda sekarang telah menambahkan file yang dipanggil `setup-sam.sh` ke root repositori.

app.js berkas

`app.js`Berisi kode fungsi Lambda. Dalam tutorial ini, kode mengembalikan teks`hello world`.

Untuk menambahkan file app.js

1. Di repositori Anda, di File, pilih Buat file.
2. Untuk nama File, masukkan:

   hello-world/app.js

3. Di kotak teks, masukkan kode berikut:

```
// const axios = require('axios')
// const url = 'http://checkip.amazonaws.com/';
let response;

/**
 *
 * Event doc: https://docs.aws.amazon.com/apigateway/latest/developerguide/set-up-
lambda-proxy-integrations.html#api-gateway-simple-proxy-for-lambda-input-format
 * @param {Object} event - API Gateway Lambda Proxy Input Format
 *
 * Context doc: https://docs.aws.amazon.com/lambda/latest/dg/nodejs-prog-model-
context.html
 * @param {Object} context
 *
 * Return doc: https://docs.aws.amazon.com/apigateway/latest/developerguide/set-up-
lambda-proxy-integrations.html
 * @returns {Object} object - API Gateway Lambda Proxy Output Format
 *
 */
exports.lambdaHandler = async (event, context) => {
    try {
        // const ret = await axios(url);
        response = {
            'statusCode': 200,
            'body': JSON.stringify({
                message: 'hello world',
                // location: ret.data.trim()
            })
        }
    } catch (err) {
```

```
        console.log(err);
        return err;
    }

    return response
};
```

4. Pilih Komit, lalu pilih Komit lagi.

   Anda sekarang telah membuat folder bernama `hello-world` dan file bernama`app.js`.

test-handler.js berkas

`test-handler.js`File berisi pengujian unit untuk fungsi Lambda.

Untuk menambahkan file test-handler.js

1. Di repositori Anda, di File, pilih Buat file.
2. Untuk nama File, masukkan:

   ```
   hello-world/tests/unit/test-handler.js
   ```

3. Di kotak teks, masukkan kode berikut:

   ```
   'use strict';

   const app = require('../../app.js');
   const chai = require('chai');
   const expect = chai.expect;
   var event, context;

   describe('Tests index', function () {
       it('verifies successful response', async () => {
           const result = await app.lambdaHandler(event, context)

           expect(result).to.be.an('object');
           expect(result.statusCode).to.equal(200);
           expect(result.body).to.be.an('string');

           let response = JSON.parse(result.body);

           expect(response).to.be.an('object');
   ```

```
        expect(response.message).to.be.equal("hello world");
        // expect(response.location).to.be.an("string");
    });
});
```

4. Pilih Komit, lalu pilih Komit lagi.

   Anda sekarang telah menambahkan file yang disebut `test-handler.js` di bawah `hello-world/tests/unit` folder.

Anda sekarang telah menambahkan semua file sumber Anda.

Luangkan waktu sejenak untuk memeriksa ulang pekerjaan Anda dan pastikan Anda menempatkan semua file di folder yang benar. Struktur folder adalah sebagai berikut:

```
.
|— hello-world
|  |— tests
|     |— unit
|        |— test-handler.js
|  |— app.js
|— .npmignore
|— README.md
|— package.json
|— sam-template.yml
|— setup-sam.sh
```

Langkah 6: Buat dan jalankan alur kerja

Pada langkah ini, Anda membuat alur kerja yang mengemas kode sumber Lambda Anda dan menerapkannya. Alur kerja terdiri dari blok bangunan berikut yang berjalan secara berurutan:

- Pemicu - Pemicu ini memulai alur kerja yang dijalankan secara otomatis saat Anda mendorong perubahan ke repositori sumber Anda. Untuk informasi lebih lanjut tentang menggunakan pemicu, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.
- Tindakan pengujian (`Test`) — Pada pemicu, tindakan ini menginstal manajer paket Node (npm), dan kemudian menjalankan perintah. `npm run test` Perintah ini memberitahu npm untuk menjalankan `test` script didefinisikan dalam `package.json` file. `test`Skrip, pada gilirannya, menjalankan pengujian unit dan menghasilkan dua laporan: laporan pengujian (`junit.xml`) dan laporan cakupan kode (`clover.xml`). Untuk informasi selengkapnya, lihat berkas package.json.

Selanjutnya, tindakan pengujian mengubah laporan XHTML menjadi CodeCatalyst laporan dan menampilkannya di CodeCatalyst konsol, di bawah tab Laporan tindakan pengujian.

Untuk informasi lebih lanjut tentang tindakan pengujian, lihatPengujian dengan alur kerja.

- Tindakan build (`BuildBackend`) — Setelah menyelesaikan tindakan pengujian, tindakan build mengunduh dan menginstal AWS SAM CLI, mengemas sumbernya, dan menyalin paket `hello-world` ke bucket Amazon S3 Anda, tempat layanan Lambda mengharapkannya. Tindakan ini juga mengeluarkan file AWS SAM template baru yang disebut `sam-template-packaged.yml` dan menempatkannya dalam artefak keluaran yang disebut. `buildArtifact`

  Untuk informasi selengkapnya tentang tindakan build, lihatMembangun dengan alur kerja.
- Tindakan penerapan (`DeployCloudFormationStack`) — Setelah menyelesaikan aksi build, tindakan deploy mencari artefak keluaran yang dihasilkan oleh build action (`buildArtifact`), menemukan AWS SAM template di dalamnya, dan kemudian menjalankan template. AWS SAM Template membuat tumpukan yang menyebarkan aplikasi tanpa server.

Untuk membuat alur kerja

1. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih Buat alur kerja.
3. Untuk repositori Sumber, pilih. `codecatalyst-cfn-source-repository`
4. Untuk Cabang, pilih`main`.
5. Pilih Buat.
6. Hapus kode sampel YAMAL.
7. Tambahkan kode YAML berikut:

```
Name: codecatalyst-cfn-workflow
SchemaVersion: 1.0

Triggers:
  - Type: PUSH
    Branches:
      - main
Actions:
  Test:
    Identifier: aws/managed-test@v1
    Inputs:
```

```
      Sources:
        - WorkflowSource
    Outputs:
      Reports:
        CoverageReport:
          Format: CLOVERXML
          IncludePaths:
            - "coverage/*"
        TestReport:
          Format: JUNITXML
          IncludePaths:
            - junit.xml
    Configuration:
      Steps:
        - Run: npm install
        - Run: npm run test
  BuildBackend:
    Identifier: aws/build@v1
    DependsOn:
      - Test
    Environment:
      Name: codecatalyst-cfn-environment
      Connections:
        - Name: codecatalyst-account-connection
          Role: codecatalyst-build-role
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      Steps:
        - Run: . ./setup-sam.sh
        - Run: sam package --template-file sam-template.yml --s3-
bucket codecatalyst-cfn-s3-bucket --output-template-file sam-template-packaged.yml
 --region us-west-2
    Outputs:
      Artifacts:
        - Name: buildArtifact
          Files:
            - "**/*"
  DeployCloudFormationStack:
    Identifier: aws/cfn-deploy@v1
    DependsOn:
      - BuildBackend
    Environment:
```

```
      Name: codecatalyst-cfn-environment
      Connections:
        - Name: codecatalyst-account-connection
          Role: codecatalyst-deploy-role
    Inputs:
      Artifacts:
        - buildArtifact
      Sources: []
    Configuration:
      name: codecatalyst-cfn-stack
      region: us-west-2
      role-arn: arn:aws:iam::111122223333:role/StackRole
      template: ./sam-template-packaged.yml
      capabilities: CAPABILITY_IAM,CAPABILITY_AUTO_EXPAND
```

Pada kode sebelumnya, ganti:

- Kedua contoh *codecatalyst-cfn-environment*dengan nama lingkungan Anda.
- Kedua contoh *codecatalyst-account-connection*dengan nama tampilan koneksi akun Anda. Nama tampilan mungkin nomor. Untuk informasi selengkapnya, lihat Langkah 3: Tambahkan AWS peran ke CodeCatalyst.
- *codecatalyst-build-role*dengan nama peran build yang Anda buatLangkah 2: Buat AWS peran.
- *codecatalyst-cfn-s3-bucket* dengan nama bucket Amazon S3 yang Anda buat. Langkah 4: Buat ember Amazon S3
- Kedua instance *us-west-2* dengan Wilayah tempat bucket Amazon S3 Anda berada (instance pertama) dan tempat tumpukan Anda akan diterapkan (instance kedua). Daerah ini bisa berbeda. Tutorial ini mengasumsikan bahwa kedua Wilayah diatur keus-west-2. Untuk detail tentang Wilayah yang didukung oleh Amazon S3 dan AWS CloudFormation, lihat Titik akhir dan kuota layanan di. Referensi Umum AWS
- *codecatalyst-deploy-role*dengan nama peran penerapan yang Anda buat. Langkah 2: Buat AWS peran
- *codecatalyst-cfn-environment*dengan nama lingkungan yang Anda buatPrasyarat.
- *arn:aws:iam: :111122223333:role/ dengan Amazon Resource Name (ARN) StackRole dari peran* stack yang Anda buat. Langkah 2: Buat AWS peran

> **Note**
>
> Jika Anda memutuskan untuk tidak membuat peran build, deploy, dan stack, replace *codecatalyst-build-rolecodecatalyst-deploy-role*, dan *arn:aws:iam:StackRole :111122223333:role/* dengan nama atau ARN peran tersebut. `CodeCatalystWorkflowDevelopmentRole-`*spaceName* Untuk informasi selengkapnya tentang peran ini, silakan lihat Langkah 2: Buat AWS peran.

Untuk informasi tentang properti dalam kode yang ditunjukkan sebelumnya, lihatTindakan "Terapkan AWS CloudFormation tumpukan" definisi YAMAL.

8. (Opsional) Pilih Validasi untuk memastikan kode YAMAL valid sebelum melakukan.
9. Pilih Terapkan.
10. Pada kotak dialog Commit workflow, masukkan yang berikut ini:

    a. Untuk nama file Workflow, pertahankan default,`codecatalyst-cfn-workflow`.

    b. Untuk pesan Commit, masukkan:

    ```
    add initial workflow file
    ```

    c. Untuk Repositori, pilih. codecatalyst-cfn-source-repository

    d. Untuk nama cabang, pilih main.

    e. Pilih Terapkan.

Anda sekarang telah membuat alur kerja. Jalankan alur kerja dimulai secara otomatis karena pemicu yang ditentukan di bagian atas alur kerja. Khususnya, ketika Anda melakukan (dan mendorong) `codecatalyst-cfn-workflow.yaml` file ke repositori sumber Anda, pemicu memulai alur kerja dijalankan.

Untuk melihat alur kerja yang sedang berjalan

1. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih alur kerja yang baru saja Anda buat:`codecatalyst-cfn-workflow`.
3. Pilih tab Runs.

4. Di kolom Run ID, pilih run ID.
5. Pilih Uji untuk melihat kemajuan tes.
6. Pilih BuildBackenduntuk melihat kemajuan pembangunan.
7. Pilih DeployCloudFormationStackuntuk melihat kemajuan penerapan.

   Untuk informasi selengkapnya tentang melihat detail run, lihatMelihat alur kerja menjalankan status dan detail.
8. Saat DeployCloudFormationStacktindakan selesai, lakukan hal berikut:

   - Jika alur kerja berjalan berhasil, pergi ke prosedur berikutnya.
   - Jika alur kerja gagal pada Pengujian atau BuildBackendtindakan, pilih Log untuk memecahkan masalah.
   - Jika alur kerja berjalan gagal pada DeployCloudFormationStacktindakan, pilih tindakan penerapan, lalu pilih tab Ringkasan. Gulir ke bagian CloudFormation peristiwa untuk melihat pesan kesalahan terperinci. Jika terjadi rollback, hapus `codecatalyst-cfn-stack` tumpukan melalui AWS CloudFormation konsol AWS sebelum menjalankan kembali alur kerja.

Untuk memverifikasi penyebaran

1. Setelah penerapan berhasil, pilih Variabel (7) dari bilah menu horizontal di dekat bagian atas. (Jangan memilih Variabel di panel di sebelah kanan.)
2. Di samping HelloWorldApi, tempel `https://` URL ke browser.

   Pesan JSON hello world dari fungsi Lambda ditampilkan, menunjukkan bahwa alur kerja berhasil digunakan dan dikonfigurasi fungsi Lambda dan API Gateway.

   > **Tip**
   >
   > Anda dapat CodeCatalyst menampilkan URL ini dalam diagram alur kerja dengan beberapa konfigurasi kecil. Untuk informasi selengkapnya, lihat Menampilkan URL aplikasi yang digunakan dalam diagram alur kerja.

Untuk memverifikasi hasil pengujian unit dan cakupan kode

1. Dalam diagram alur kerja, pilih Uji, lalu pilih Laporan.

2. Pilih TestReportuntuk melihat hasil pengujian unit, atau pilih CoverageReportuntuk melihat detail cakupan kode file yang sedang diuji, dalam hal ini, `app.js` dan`test-handler.js`.

Untuk memverifikasi sumber daya yang digunakan

1. Masuk ke AWS Management Console dan buka konsol API Gateway di https://console.aws.amazon.com/apigateway/.
2. Amati codecatalyst-cfn-stackAPI yang dibuat AWS SAM template. Nama API berasal dari `Configuration/name` nilai dalam file definisi alur kerja (`codecatalyst-cfn-workflow.yaml`).
3. Buka AWS Lambda konsol di https://console.aws.amazon.com/lambda/.
4. Di panel navigasi, pilih Fungsi.
5. Pilih fungsi Lambda Anda,. `codecatalyst-cfn-stack-HelloWorldFunction-`*`string`*
6. Anda dapat melihat bagaimana API Gateway adalah pemicu fungsi tersebut. Integrasi ini secara otomatis dikonfigurasi oleh jenis AWS SAM `AWS::Serverless::Function` sumber daya.

Langkah 7: Buat perubahan

Pada langkah ini, Anda membuat perubahan pada kode sumber Lambda Anda dan melakukan itu. Komit ini memulai alur kerja baru. Proses ini menerapkan fungsi Lambda baru dalam skema biru-hijau yang menggunakan konfigurasi perpindahan lalu lintas default yang ditentukan di konsol Lambda.

Untuk membuat perubahan pada sumber Lambda Anda

1. Masuk CodeCatalyst, navigasikan ke proyek Anda.
2. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
3. Pilih repositori `codecatalyst-cfn-source-repository` sumber Anda.
4. Ubah file aplikasi:
    a. Pilih `hello-world` foldernya.
    b. Pilih `app.js` file.
    c. Pilih Edit.
    d. Pada baris 23, ubah `hello world` ke**`Tutorial complete!`**.
    e. Pilih Komit, lalu pilih Komit lagi.

Komit menyebabkan alur kerja dijalankan. Proses ini akan gagal karena Anda belum memperbarui pengujian unit untuk mencerminkan perubahan nama.

5. Perbarui pengujian unit:
   a. Pilih `hello-world\tests\unit\test-handler.js`.
   b. Pilih Edit.
   c. Pada baris 19, ubah `hello world` ke**`Tutorial complete!`**.
   d. Pilih Komit, lalu pilih Komit lagi.

   Komit menyebabkan alur kerja lain dijalankan. Lari ini akan berhasil.
6. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
7. Pilih`codecatalyst-cfn-workflow`, lalu pilih Runs.
8. Pilih ID run dari run terbaru. Seharusnya masih dalam proses.
9. Pilih Uji, BuildBackend, dan DeployCloudFormationStackuntuk melihat alur kerja berjalan kemajuan.
10. Saat alur kerja selesai, pilih Variabel (7) di dekat bagian atas.
11. Di samping HelloWorldApi, tempel `https://` URL ke browser.

    Sebuah `Tutorial complete!` pesan muncul di browser, menunjukkan bahwa aplikasi baru Anda berhasil digunakan.

Bersihkan

Bersihkan file dan layanan yang digunakan dalam tutorial ini untuk menghindari biaya untuk mereka.

Untuk membersihkan di CodeCatalyst konsol

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Hapus`codecatalyst-cfn-workflow`.
3. Hapus`codecatalyst-cfn-environment`.
4. Hapus`codecatalyst-cfn-source-repository`.
5. Hapus`codecatalyst-cfn-project`.

Untuk membersihkan di AWS Management Console

1. Bersihkan CloudFormation, sebagai berikut:

   a. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.

   b. Hapus`codecatalyst-cfn-stack`.

      Menghapus tumpukan menghapus semua sumber daya tutorial dari API Gateway dan layanan Lambda.

2. Bersihkan di Amazon S3, sebagai berikut:

   a. Buka konsol Amazon S3 di https://console.aws.amazon.com/s3/.

   b. Pilih `codecatalyst-cfn-s3-bucket`.

   c. Hapus isi ember.

   d. Hapus bucket.

3. Bersihkan di IAM, sebagai berikut:

   a. Buka konsol IAM di https://console.aws.amazon.com/iam/.

   b. Hapus`codecatalyst-deploy-policy`.

   c. Hapus`codecatalyst-build-policy`.

   d. Hapus`codecatalyst-stack-policy`.

   e. Hapus`codecatalyst-deploy-role`.

   f. Hapus`codecatalyst-build-role`.

   g. Hapus`codecatalyst-stack-role`.

Dalam tutorial ini, Anda belajar bagaimana menerapkan aplikasi tanpa server sebagai CloudFormation tumpukan menggunakan CodeCatalyst alur kerja dan tindakan tumpukan Deploy. AWS CloudFormation

Menambahkan tindakan “Deploy AWS CloudFormation stack”

Gunakan petunjuk berikut untuk menambahkan tindakan AWS CloudFormation tumpukan Deploy ke alur kerja Anda.

Visual

Untuk menambahkan tindakan “Deploy AWS CloudFormation stack” menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan AWS CloudFormation tumpukan Deploy, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih Deploy AWS CloudFormation stack. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Unduh untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Pada tab Input dan Konfigurasi, lengkapi bidang sesuai dengan kebutuhan Anda. Untuk deskripsi setiap bidang, lihatTindakan “Terapkan AWS CloudFormation tumpukan” definisi YAMAL. Referensi ini memberikan informasi rinci tentang setiap bidang (dan nilai properti YAMAL yang sesuai) seperti yang muncul di YAMAL dan editor visual.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menambahkan tindakan “Deploy AWS CloudFormation stack” menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan AWS CloudFormation tumpukan Deploy, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih Deploy AWS CloudFormation stack. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Unduh untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Ubah properti dalam kode YAMAL sesuai dengan kebutuhan Anda. Penjelasan tentang setiap properti yang tersedia disediakan diTindakan “Terapkan AWS CloudFormation tumpukan” definisi YAMAL.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Mengkonfigurasi rollback

Secara default, jika tindakan AWS CloudFormation tumpukan Deploy gagal, itu akan menyebabkan AWS CloudFormation untuk memutar kembali tumpukan ke status stabil terakhir yang diketahui. Anda dapat mengubah perilaku sehingga rollback terjadi tidak hanya ketika tindakan gagal, tetapi

juga ketika CloudWatch alarm Amazon tertentu terjadi. Untuk informasi selengkapnya tentang CloudWatch alarm, lihat Menggunakan CloudWatch alarm Amazon di CloudWatch Panduan Pengguna Amazon.

Anda juga dapat mengubah perilaku default sehingga CloudFormation tidak memutar kembali tumpukan ketika tindakan gagal.

Gunakan petunjuk berikut untuk mengkonfigurasi rollback.

> **Note**
>
> Anda tidak dapat memulai rollback secara manual.

Visual

Sebelum Anda mulai

1. Pastikan Anda memiliki alur kerja yang menyertakan tindakan AWS CloudFormation tumpukan Deploy yang berfungsi. Untuk informasi selengkapnya, lihat Menerapkan AWS CloudFormation tumpukan dengan alur kerja.
2. Dalam peran yang ditentukan dalam peran Stack - bidang opsional dari tindakan AWS CloudFormation tumpukan Deploy, pastikan untuk menyertakan CloudWatchFullAccessizin. Untuk informasi tentang membuat peran ini dengan izin yang sesuai, lihatLangkah 2: Buat AWS peran.

Untuk mengonfigurasi alarm rollback untuk tindakan " AWS CloudFormation Deploy stack"

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja yang menyertakan tindakan AWS CloudFormation tumpukan Deploy. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Pilih tindakan AWS CloudFormation tumpukan Deploy Anda.

8. Di panel detail, pilih Konfigurasi.
9. Di bagian bawah, perluas Advanced.
10. Di bawah Memantau ARN alarm, pilih Tambahkan alarm.
11. Masukkan informasi ke dalam bidang berikut.

    - Alarm ARN

      Tentukan Nama Sumber Daya Amazon (ARN) dari CloudWatch alarm Amazon untuk digunakan sebagai pemicu rollback. Misalnya, `arn:aws:cloudwatch::123456789012:alarm/MyAlarm`. Anda dapat memiliki maksimal lima pemicu rollback.

      > **Note**
      >
      > Jika Anda menentukan ARN CloudWatch alarm, Anda juga harus mengonfigurasi izin tambahan untuk mengaktifkan tindakan untuk mengakses. CloudWatch Untuk informasi selengkapnya, lihat Mengkonfigurasi rollback.

    - Waktu pemantauan

      Tentukan jumlah waktu, dari 0 hingga 180 menit, di mana CloudFormation memantau alarm yang ditentukan. Pemantauan dimulai setelah semua sumber daya tumpukan telah digunakan. Jika alarm terjadi dalam waktu pemantauan yang ditentukan, maka penerapan gagal, dan CloudFormation memutar kembali seluruh operasi tumpukan.

      Default: 0. CloudFormation hanya memantau alarm saat sumber daya tumpukan sedang digunakan, bukan setelahnya.

YAML

Untuk mengonfigurasi pemicu rollback untuk tindakan “ AWS CloudFormation Deploy stack”

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.

4. Pilih nama alur kerja yang menyertakan tindakan AWS CloudFormation tumpukan Deploy. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMG.
7. Tambahkan `monitor-alarm-arns` dan `monitor-timeout-in-minutes` properti dalam kode YAMAL untuk menambahkan pemicu rollback. Untuk penjelasan masing-masing properti, lihatTindakan “Terapkan AWS CloudFormation tumpukan” definisi YAMAL.
8. Dalam peran yang ditentukan dalam `role-arn` properti tindakan AWS CloudFormation tumpukan Deploy, pastikan untuk menyertakan CloudWatchFullAccessizin. Untuk informasi tentang membuat peran ini dengan izin yang sesuai, lihatLangkah 2: Buat AWS peran.

Visual

Untuk mematikan rollback untuk tindakan “Deploy stack AWS CloudFormation ”

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja yang menyertakan tindakan AWS CloudFormation tumpukan Deploy. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Pilih tindakan AWS CloudFormation tumpukan Deploy Anda.
8. Di panel detail, pilih Konfigurasi.
9. Di bagian bawah, perluas Advanced.
10. Nyalakan Nonaktifkan rollback.

YAML

Untuk mematikan rollback untuk tindakan “Deploy stack AWS CloudFormation ”

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.

2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja yang menyertakan tindakan AWS CloudFormation tumpukan Deploy. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMG.
7. Tambahkan `disable-rollback: 1` properti dalam kode YAMAL untuk menghentikan rollback. Untuk penjelasan tentang properti ini, lihatTindakan “Terapkan AWS CloudFormation tumpukan” definisi YAMAL.

Variabel yang dihasilkan oleh tindakan “Deploy AWS CloudFormation stack”

Tindakan AWS CloudFormation tumpukan Deploy menghasilkan dan menetapkan variabel berikut pada waktu berjalan. Ini dikenal sebagai variabel yang telah ditentukan.

Untuk informasi tentang mereferensikan variabel-variabel ini dalam alur kerja, lihat. Menggunakan variabel yang telah ditentukan

Kunci	Nilai
platform penyebaran	Nama platform penyebaran. Hardcode ke. `AWS:CloudFormation`
region	Kode wilayah Wilayah AWS yang digunakan selama alur kerja dijalankan. Contoh: `us-west-2`
tumpukan-id	Nama Sumber Daya Amazon (ARN) dari tumpukan yang digunakan. Contoh: `arn:aws:cloudformation:us-west-2:111122223333:stack/codecatalyst-cfn-stack/6aad4380-100a-11ec-a10a-03b8a84d40df`

Tindakan “Terapkan AWS CloudFormation tumpukan” definisi YAMAL

Berikut ini adalah definisi YAMAL dari tindakan AWS CloudFormation tumpukan Deploy. Untuk mempelajari cara menggunakan tindakan ini, lihatMenerapkan AWS CloudFormation tumpukan dengan alur kerja.

Definisi tindakan ini ada sebagai bagian dalam file definisi alur kerja yang lebih luas. Untuk informasi selengkapnya tentang file ini, lihatAlur kerja definisi YAMAL.

> **Note**
>
> Sebagian besar properti YAMAL yang mengikuti memiliki elemen UI yang sesuai di editor visual. Untuk mencari elemen UI, gunakan Ctrl+F. Elemen akan terdaftar dengan properti YAMLnya yang terkait.

```
# The workflow definition starts here.
# See Properti tingkat atas for details.

Name: MyWorkflow
SchemaVersion: 1.0
Actions:

# The action definition starts here.
  DeployCloudFormationStack:
    Identifier: aws/cfn-deploy@v1
    DependsOn:
      - build-action
    Compute:
      Type: EC2 | Lambda
      Fleet: fleet-name
    Timeout: timeout-minutes
    Environment:
      Name: environment-name
      Connections:
        - Name: account-connection-name
          Role: DeployRole
    Inputs:
      Sources:
        - source-name-1
      Artifacts:
        - CloudFormation-artifact
```

```
    Configuration:
      name: stack-name
      region: us-west-2
      template: template-path
      role-arn: arn:aws:iam::123456789012:role/StackRole
      capabilities: CAPABILITY_IAM,CAPABILITY_NAMED_IAM,CAPABILITY_AUTO_EXPAND
      parameter-overrides: KeyOne=ValueOne,KeyTwo=ValueTwo | path-to-JSON-file
      no-execute-changeset: 1|0
      fail-on-empty-changeset: 1|0
      disable-rollback: 1|0
      termination-protection: 1|0
      timeout-in-minutes: minutes
      notification-arns: arn:aws:sns:us-east-1:123456789012:MyTopic,arn:aws:sns:us-
east-1:123456789012:MyOtherTopic
      monitor-alarm-arns: arn:aws:cloudwatch::123456789012:alarm/
MyAlarm,arn:aws:cloudwatch::123456789012:alarm/MyOtherAlarm
      monitor-timeout-in-minutes: minutes
      tags: '[{"Key":"MyKey1","Value":"MyValue1"},{"Key":"MyKey2","Value":"MyValue2"}]'
```

DeployCloudFormationStack

(Diperlukan)

Tentukan nama tindakan. Semua nama tindakan harus unik dalam alur kerja. Nama aksi terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama tindakan.

Default: `DeployCloudFormationStack_nn`.

UI yang sesuai: Tab konfigurasi/Nama tampilan tindakan

Identifier

(*DeployCloudFormationStack*/Identifier)

(Diperlukan)

Mengidentifikasi tindakan. Jangan mengubah properti ini kecuali Anda ingin mengubah versi. Untuk informasi selengkapnya, lihat Menentukan versi mayor, minor, atau patch dari suatu tindakan.

Default: `aws/cfn-deploy@v1`.

UI yang sesuai: Diagram alur DeployCloudFormationStack kerja/_nn/ aws/cfn-deploy @v1 label

DependsOn

(*DeployCloudFormationStack*/DependsOn)

(Opsional)

Tentukan tindakan, grup tindakan, atau gerbang yang harus berjalan dengan sukses agar tindakan ini berjalan.

Untuk informasi selengkapnya tentang fungsionalitas 'tergantung pada', lihat. Mengkonfigurasi tindakan untuk bergantung pada tindakan lain

UI yang sesuai: Tab masukan/Tergantung pada - opsional

Compute

(*DeployCloudFormationStack*/Compute)

(Opsional)

Mesin komputasi yang digunakan untuk menjalankan tindakan alur kerja Anda. Anda dapat menentukan komputasi baik di tingkat alur kerja atau di tingkat tindakan, tetapi tidak keduanya. Ketika ditentukan pada tingkat alur kerja, konfigurasi komputasi berlaku untuk semua tindakan yang ditentukan dalam alur kerja. Pada tingkat alur kerja, Anda juga dapat menjalankan beberapa tindakan pada instance yang sama. Untuk informasi selengkapnya, lihat Berbagi komputasi di seluruh tindakan.

UI yang sesuai: tidak ada

Type

(*DeployCloudFormationStack*/Compute/Type)

(Diperlukan Compute jika disertakan)

Jenis mesin komputasi. Anda dapat menggunakan salah satu nilai berikut:

- EC2 (editor visual) atau EC2 (editor YAMG)

  Dioptimalkan untuk fleksibilitas selama aksi berjalan.

- Lambda (editor visual) atau `Lambda` (editor YAMG)

  Kecepatan start-up aksi yang dioptimalkan.

Untuk informasi selengkapnya tentang jenis komputasi, lihatJenis komputasi.

UI yang sesuai: Tab Konfigurasi/Tingkat Lanjut - opsional/Jenis komputasi

Fleet

(*`DeployCloudFormationStack`*/Compute/Fleet)

(Opsional)

Tentukan mesin atau armada yang akan menjalankan alur kerja atau tindakan alur kerja Anda. Dengan armada sesuai permintaan, ketika suatu tindakan dimulai, alur kerja menyediakan sumber daya yang dibutuhkan, dan mesin dihancurkan ketika tindakan selesai. Contoh armada sesuai permintaan:`Linux.x86-64.Large`,. `Linux.x86-64.XLarge` Untuk informasi lebih lanjut tentang armada sesuai permintaan, lihat. Properti armada sesuai permintaan

Dengan armada yang disediakan, Anda mengonfigurasi satu set mesin khusus untuk menjalankan tindakan alur kerja Anda. Mesin-mesin ini tetap menganggur, siap untuk memproses tindakan segera. Untuk informasi lebih lanjut tentang armada yang disediakan, lihat. Properti armada yang disediakan

Jika `Fleet` dihilangkan, defaultnya adalah. `Linux.x86-64.Large`

UI yang sesuai: Tab Konfigurasi/Advanced - armada opsional/Komputasi

Timeout

(*`DeployCloudFormationStack`*/Timeout)

(Opsional)

Tentukan jumlah waktu dalam menit (editor YAMG), atau jam dan menit (editor visual), bahwa tindakan dapat berjalan sebelum CodeCatalyst mengakhiri tindakan. Minimal adalah 5 menit dan maksimum dijelaskan dalamKuota untuk alur kerja. Batas waktu default sama dengan batas waktu maksimum.

UI yang sesuai: Tab konfigurasi/Batas waktu dalam hitungan menit - opsional

Environment

(*DeployCloudFormationStack*/Environment)

(Diperlukan)

Tentukan CodeCatalyst lingkungan yang akan digunakan dengan tindakan.

Untuk informasi lebih lanjut tentang lingkungan, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst danPembuatan lingkungan.

UI yang sesuai: Tab Konfigurasi/'lingkungan/akun/peran'/Lingkungan

Name

(*DeployCloudFormationStack*/Environment/Name)

(Diperlukan Environment jika disertakan)

Tentukan nama lingkungan yang ada yang ingin Anda kaitkan dengan tindakan.

UI yang sesuai: Tab Konfigurasi/'lingkungan/akun/peran'/Lingkungan

Connections

(*DeployCloudFormationStack*/Environment/Connections)

(Diperlukan Environment jika disertakan)

Tentukan koneksi akun untuk dikaitkan dengan tindakan. Anda dapat menentukan maksimum satu koneksi akun di bawah`Environment`.

Untuk informasi selengkapnya tentang koneksi akun, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS. Untuk informasi tentang cara mengaitkan koneksi akun dengan lingkungan Anda, lihatPembuatan lingkungan.

UI yang sesuai: Tab konfigurasi/'lingkungan/akun/peran'/koneksi akun AWS

Name

(*DeployCloudFormationStack*/Environment/Connections/Name)

(Diperlukan)

Tentukan nama koneksi akun.

UI yang sesuai: Tab konfigurasi/'lingkungan/akun/peran'/koneksi akun AWS

Role

(*DeployCloudFormationStack*/Environment/Connections/Role)

(Diperlukan)

Tentukan nama peran IAM yang digunakan tindakan AWS CloudFormation tumpukan Deploy untuk mengakses AWS dan layanan. AWS CloudFormation

Pastikan peran ini mencakup kebijakan berikut:

- Kebijakan izin berikut:

> **Warning**
>
> Batasi izin untuk yang ditampilkan dalam kebijakan berikut. Menggunakan peran dengan izin yang lebih luas dapat menimbulkan risiko keamanan.

```
{
    "Version": "2012-10-17",
    "Statement": [{
    "Action": [
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack",
        "cloudformation:Describe*",
        "cloudformation:UpdateStack",
        "cloudformation:CreateChangeSet",
        "cloudformation:DeleteChangeSet",
        "cloudformation:ExecuteChangeSet",
        "cloudformation:SetStackPolicy",
        "cloudformation:ValidateTemplate",
        "cloudformation:List*",
        "iam:PassRole"
    ],
    "Resource": "*",
    "Effect": "Allow"
```

```
}]
}
```

> **Note**
>
> Pertama kali peran digunakan, gunakan wildcard berikut dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

- Kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                 ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Pastikan peran ini ditambahkan ke koneksi akun di ruang Anda. Untuk mempelajari selengkapnya tentang menambahkan peran IAM ke koneksi akun, lihatMenambahkan peran IAM ke koneksi akun.

> **Note**
>
> Anda dapat menentukan nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran di sini, jika Anda mau. Untuk informasi selengkapnya tentang peran ini, silakan lihat Membuat CodeCatalystWorkflowDevelopmentRole-***spaceName***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami

menyarankan Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian.

UI yang sesuai: Tab Konfigurasi/'lingkungan/akun/peran'/ Peran

Inputs

(*DeployCloudFormationStack*/Inputs)

(Opsional)

`Inputs`Bagian ini mendefinisikan data yang `DeployCloudFormationStack` dibutuhkan selama menjalankan alur kerja.

> **Note**
>
> Maksimal empat input (satu sumber dan tiga artefak) diizinkan per tindakan tumpukan Deploy AWS CloudFormation .

Jika Anda perlu merujuk ke file yang berada di input yang berbeda (katakanlah sumber dan artefak), input sumber adalah input utama, dan artefak adalah input sekunder. Referensi ke file dalam input sekunder mengambil awalan khusus untuk menyisihkannya dari primer. Lihat perinciannya di Contoh: Merujuk file dalam beberapa artefak.

UI yang sesuai: Tab input

Sources

(*DeployCloudFormationStack*/Inputs/Sources)

(Diperlukan jika CloudFormation atau AWS SAM template Anda disimpan dalam repositori sumber)

Jika AWS SAM template CloudFormation atau Anda disimpan dalam repositori sumber, tentukan label repositori sumber tersebut. Saat ini, satu-satunya label yang didukung adalah`WorkflowSource`.

Jika Anda CloudFormation atau AWS SAM template tidak terkandung dalam repositori sumber, itu harus berada dalam artefak yang dihasilkan oleh tindakan lain, atau dalam bucket Amazon S3.

Untuk informasi selengkapnya tentang sumber, lihat Menghubungkan alur kerja ke repositori sumber.

UI yang sesuai: Tab/Sumber Input - opsional

Artifacts - input

(*DeployCloudFormationStack*/Inputs/Artifacts)

(Diperlukan jika CloudFormation atau AWS SAM template Anda disimpan dalam artefak keluaran dari tindakan sebelumnya)

Jika AWS SAM templat CloudFormation atau yang ingin Anda terapkan terkandung dalam artefak yang dihasilkan oleh tindakan sebelumnya, tentukan artefak tersebut di sini. Jika CloudFormation template Anda tidak terkandung dalam artefak, itu harus berada di repositori sumber Anda atau di bucket Amazon S3.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab konfigurasi/Artefak - opsional

Configuration

(*DeployCloudFormationStack*/Configuration)

(Diperlukan)

Bagian di mana Anda dapat menentukan properti konfigurasi tindakan.

UI yang sesuai: Tab konfigurasi

name

(*DeployCloudFormationStack*/Configuration/name)

(Diperlukan)

Tentukan nama untuk CloudFormation tumpukan yang dibuat atau diperbarui oleh tindakan AWS CloudFormation tumpukan Deploy.

UI yang sesuai: Tab konfigurasi/Nama tumpukan

region

(*DeployCloudFormationStack*/Configuration/region)

(Diperlukan)

Tentukan Wilayah AWS ke mana tumpukan akan digunakan. Untuk daftar kode Wilayah, lihat Titik akhir Regional.

UI yang sesuai: Tab konfigurasi/Wilayah tumpukan

template

(*DeployCloudFormationStack*/Configuration/template)

(Diperlukan)

Tentukan nama dan jalur ke file CloudFormation atau AWS SAM template Anda. Template dapat dalam format JSON atau YAMG, dan dapat berada di repositori sumber, artefak dari tindakan sebelumnya, atau bucket Amazon S3. Jika file template berada dalam repositori sumber atau artefak, jalurnya relatif terhadap sumber atau akar artefak. Jika template berada di bucket Amazon S3, jalurnya adalah nilai URL Objek template.

Contoh:

```
./MyFolder/MyTemplate.json
```

```
MyFolder/MyTemplate.yml
```

```
https://MyBucket.s3.us-west-2.amazonaws.com/MyTemplate.yml
```

> **Note**
>
> Anda mungkin perlu menambahkan awalan ke jalur file template untuk menunjukkan artefak atau sumber mana yang akan menemukannya. Untuk informasi selengkapnya, lihat Mereferensikan file dalam repositori sumber dan Merujuk file dalam artefak.

UI yang sesuai: Tab/Template Konfigurasi

role-arn

(*DeployCloudFormationStack*/Configuration/role-arn)

(Diperlukan)

Tentukan Nama Sumber Daya Amazon (ARN) dari peran tumpukan. CloudFormation menggunakan peran ini untuk mengakses dan memodifikasi sumber daya di tumpukan Anda. Misalnya: `arn:aws:iam::123456789012:role/StackRole`.

Pastikan peran tumpukan meliputi:

- Satu atau beberapa kebijakan izin. Kebijakan tergantung pada sumber daya yang Anda miliki di tumpukan Anda. Misalnya, jika tumpukan menyertakan AWS Lambda fungsi, Anda perlu menambahkan izin yang memberikan akses ke Lambda. Jika Anda mengikuti tutorial yang dijelaskanTutorial: Menyebarkan aplikasi tanpa server menggunakan AWS CloudFormation, itu termasuk prosedur berjudul, Untuk membuat peran tumpukan yang mencantumkan izin yang dibutuhkan peran tumpukan jika Anda menerapkan tumpukan aplikasi tanpa server yang khas.

  > **Warning**
  >
  > Batasi izin untuk yang diperlukan oleh CloudFormation layanan untuk mengakses sumber daya di tumpukan Anda. Menggunakan peran dengan izin yang lebih luas dapat menimbulkan risiko keamanan.

- Kebijakan kepercayaan berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudformation.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Secara opsional, kaitkan peran ini dengan koneksi akun Anda. Untuk mempelajari lebih lanjut tentang mengaitkan peran IAM dengan koneksi akun, lihat. Menambahkan peran IAM ke koneksi akun Jika Anda tidak mengaitkan peran tumpukan dengan koneksi akun, maka peran tumpukan tidak akan

muncul di daftar drop-down peran Stack di editor visual; namun, peran ARN masih dapat ditentukan di `role-arn` bidang menggunakan editor YAMG.

> **Note**
>
> Anda dapat menentukan nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran di sini, jika Anda mau. Untuk informasi selengkapnya tentang peran ini, silakan lihat Membuat CodeCatalystWorkflowDevelopmentRole-***spaceName***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami menyarankan Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian.

UI yang sesuai: Tab konfigurasi/Peran tumpukan - opsional

capabilities

(*`DeployCloudFormationStack`*/Configuration/capabilities)

(Diperlukan)

Tentukan daftar kemampuan IAM yang diperlukan untuk memungkinkan AWS CloudFormation untuk membuat tumpukan tertentu. Dalam kebanyakan kasus, Anda dapat meninggalkan `capabilities` dengan nilai default`CAPABILITY_IAM,CAPABILITY_NAMED_IAM,CAPABILITY_AUTO_EXPAND`.

Jika Anda melihat `##[error] requires capabilities: [`*`capability-name`*`]` log tindakan AWS CloudFormation tumpukan Deploy, lihat Bagaimana cara memperbaiki kesalahan kemampuan IAM? informasi tentang cara memperbaiki masalah.

Untuk informasi selengkapnya tentang kemampuan IAM, lihat Mengakui sumber daya IAM dalam AWS CloudFormation templat di Panduan Pengguna IAM.

UI yang sesuai: Tab Konfigurasi/Lanjutan/Kemampuan

parameter-overrides

(*`DeployCloudFormationStack`*/Configuration/parameter-overrides)

(Opsional)

Tentukan parameter dalam AWS CloudFormation atau AWS SAM templat yang tidak memiliki nilai default, atau yang ingin Anda tentukan nilai non-default. Untuk informasi selengkapnya tentang parameter, lihat Parameter di Panduan AWS CloudFormation Pengguna.

`parameter-overrides`Properti menerima:

- File JSON yang berisi parameter dan nilai.
- Daftar parameter dan nilai yang terpisah koma.

Untuk menentukan file JSON

1. Pastikan file JSON menggunakan salah satu sintaks berikut:

```
{
  "Parameters": {
    "Param1": "Value1",
    "Param2": "Value2",
    ...
  }
}
```

Atau...

```
[
  {
     "ParameterKey": "Param1",
     "ParameterValue": "Value1"
  },
  ...
]
```

(Ada sintaks lain, tetapi tidak didukung oleh CodeCatalyst pada saat penulisan.) Untuk informasi selengkapnya tentang menentukan CloudFormation parameter dalam file JSON, lihat Sintaks JSON yang didukung di Referensi Perintah.AWS CLI

2. Tentukan jalur ke file JSON menggunakan salah satu format berikut:

   - Jika file JSON Anda berada dalam artefak keluaran dari tindakan sebelumnya, gunakan:

     `file:///artifacts/`*`current-action-name`*`/`*`output-artifact-name`*`/`*`path-to-json-file`*

Lihat Contoh 1 untuk detailnya.

- Jika file JSON Anda berada di repositori sumber Anda, gunakan:

  `file:///sources/WorkflowSource/`*`path-to-json-file`*

  Lihat Contoh 2 untuk detailnya.

  Contoh 1 - File JSON berada di artefak keluaran

```
##My workflow YAML
...
Actions:
  MyBuildAction:
    Identifier: aws/build@v1
    Outputs:
      Artifacts:
        - Name: ParamArtifact
          Files:
            - params.json
    Configuration:
    ...
  MyDeployCFNStackAction:
    Identifier: aws/cfn-deploy@v1
    Configuration:
      parameter-overrides: file:///artifacts/MyDeployCFNStackAction/
ParamArtifact/params.json
```

  Contoh 2 - File JSON berada di repositori sumber Anda, dalam folder bernama `my/folder`

```
##My workflow YAML
...
Actions:
  MyDeployCloudFormationStack:
    Identifier: aws/cfn-deploy@v1
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      parameter-overrides: file:///sources/WorkflowSource/my/folder/params.json
```

Untuk menggunakan daftar parameter yang terpisah koma

- Tambahkan pasangan nama-nilai parameter di `parameter-overrides` properti menggunakan format berikut:

  param-1=*value-1*,*param-2*=*value-2*

  Misalnya, dengan asumsi AWS CloudFormation template berikut:

  ```
  ##My CloudFormation template

  Description: My AWS CloudFormation template

  Parameters:
    InstanceType:
      Description: Defines the Amazon EC2 compute for the production server.
      Type: String
      Default: t2.micro
      AllowedValues:
        - t2.micro
        - t2.small
        - t3.medium

  Resources:
  ...
  ```

  ... Anda dapat mengatur `parameter-overrides` properti sebagai berikut:

  ```
  ##My workflow YAML
  ...
  Actions:
  ...
    DeployCloudFormationStack:
      Identifier: aws/cfn-deploy@v1
      Configuration:
        parameter-overrides: InstanceType=t3.medium,UseVPC=true
  ```

  > **Note**
  >
  > Anda dapat menentukan nama parameter tanpa nilai yang sesuai menggunakan `undefined` sebagai nilai. Sebagai contoh:

`parameter-overrides: MyParameter=undefined`
Efeknya adalah bahwa selama pembaruan tumpukan, CloudFormation menggunakan nilai parameter yang ada untuk nama parameter yang diberikan.

UI yang sesuai:

- Tab konfigurasi/Lanjutan/Parameter menimpa
- Tab konfigurasi/Advanced/Parameter overrides/ Tentukan penggantian menggunakan file
- Tab konfigurasi/Advanced/Parameter overrides/ Tentukan penggantian menggunakan set nilai

no-execute-changeset

(*DeployCloudFormationStack*/Configuration/no-execute-changeset)

(Opsional)

Tentukan apakah Anda CodeCatalyst ingin membuat set CloudFormation perubahan dan kemudian berhenti sebelum menjalankannya. Ini memberi Anda kesempatan untuk meninjau perubahan yang ditetapkan di CloudFormation konsol. Jika Anda menentukan bahwa set perubahan terlihat bagus, nonaktifkan opsi ini dan kemudian jalankan kembali alur kerja sehingga CodeCatalyst dapat membuat dan menjalankan set perubahan tanpa henti. Defaultnya adalah membuat dan menjalankan set perubahan tanpa henti. Untuk informasi selengkapnya, lihat parameter AWS CloudFormation penerapan di Referensi AWS CLI Perintah. Untuk informasi selengkapnya tentang melihat set perubahan, lihat Melihat set perubahan di Panduan AWS CloudFormation Pengguna.

UI yang sesuai: Tab konfigurasi/Lanjutan/Tidak ada set perubahan eksekusi

fail-on-empty-changeset

(*DeployCloudFormationStack*/Configuration/fail-on-empty-changeset)

(Opsional)

Tentukan apakah Anda CodeCatalyst ingin gagal dalam tindakan AWS CloudFormation tumpukan Deploy jika set CloudFormation perubahan kosong. (Jika set perubahan kosong, itu berarti tidak ada perubahan yang dibuat pada tumpukan selama penerapan terbaru.) Defaultnya adalah mengizinkan tindakan untuk melanjutkan jika set perubahan kosong, dan mengembalikan `UPDATE_COMPLETE` pesan meskipun tumpukan tidak diperbarui.

Untuk informasi selengkapnya tentang setelan ini, lihat parameter AWS CloudFormation penerapan di Referensi AWS CLI Perintah. Untuk informasi selengkapnya tentang set perubahan, lihat Memperbarui tumpukan menggunakan set perubahan di Panduan AWS CloudFormation Pengguna.

UI yang sesuai: Tab konfigurasi/Lanjutan/Gagal pada set perubahan kosong

disable-rollback

(*DeployCloudFormationStack*/Configuration/disable-rollback)

(Opsional)

Tentukan apakah Anda CodeCatalyst ingin memutar kembali penerapan tumpukan jika gagal. Rollback mengembalikan tumpukan ke keadaan stabil terakhir yang diketahui. Defaultnya adalah mengaktifkan rollback. Untuk informasi selengkapnya tentang setelan ini, lihat parameter AWS CloudFormation penerapan di Referensi AWS CLI Perintah.

Untuk informasi selengkapnya tentang cara tindakan AWS CloudFormation tumpukan Deploy menangani rollback, lihat. Mengkonfigurasi rollback

Untuk informasi selengkapnya tentang memutar kembali tumpukan, lihat Opsi kegagalan tumpukan di Panduan AWS CloudFormation Pengguna.

UI yang sesuai: Tab Konfigurasi/Lanjutan/Nonaktifkan rollback

termination-protection

(*DeployCloudFormationStack*/Configuration/termination-protection)

(Opsional)

Tentukan apakah Anda ingin AWS CloudFormation tumpukan Deploy menambahkan perlindungan terminasi ke tumpukan yang disebarkan. Jika pengguna mencoba menghapus tumpukan dengan perlindungan pengakhiran diaktifkan, penghapusan gagal dan tumpukan, termasuk statusnya, tetap tidak berubah. Defaultnya adalah menonaktifkan perlindungan terminasi. Untuk informasi selengkapnya, lihat Melindungi tumpukan agar tidak dihapus di Panduan AWS CloudFormation Pengguna.

UI yang sesuai: Tab Konfigurasi/Perlindungan lanjutan/Penghentian

timeout-in-minutes

(*DeployCloudFormationStack*/Configuration/timeout-in-minutes)

(Opsional)

Tentukan jumlah waktu, dalam menit, yang CloudFormation harus dialokasikan sebelum mengatur waktu operasi pembuatan tumpukan dan mengatur status tumpukan ke`CREATE_FAILED`. Jika tidak CloudFormation dapat membuat seluruh tumpukan dalam waktu yang ditentukan, itu gagal pembuatan tumpukan karena batas waktu dan memutar kembali tumpukan.

Secara default, tidak ada waktu habis untuk pembuatan tumpukan. Namun, sumber daya individu mungkin memiliki waktu habis sendiri berdasarkan sifat layanan yang diterapkan. Misalnya, jika waktu sumber daya individual di tumpukan Anda habis, waktu pembuatan tumpukan juga akan habis meskipun waktu habis yang Anda tentukan untuk pembuatan tumpukan belum tercapai.

UI yang sesuai: Tab konfigurasi/lanjutan/batas waktu CloudFormation

notification-arns

(*`DeployCloudFormationStack`*/Configuration/notification-arns)

(Opsional)

Tentukan ARN topik Amazon SNS yang CodeCatalyst ingin Anda kirimi pesan notifikasi. Misalnya, `arn:aws:sns:us-east-1:111222333:MyTopic`. Saat tindakan AWS CloudFormation tumpukan Deploy berjalan, CodeCatalyst berkoordinasi dengan CloudFormation untuk mengirim satu notifikasi per AWS CloudFormation peristiwa yang terjadi selama proses pembuatan atau pembaruan tumpukan. (Peristiwa terlihat di tab Peristiwa AWS CloudFormation konsol untuk tumpukan.) Anda dapat menentukan hingga lima topik. Untuk informasi lebih lanjut, lihat Apa itu Amazon SNS? .

UI yang sesuai: Tab Konfigurasi/Lanjutan/ARN Pemberitahuan

monitor-alarm-arns

(*`DeployCloudFormationStack`*/Configuration/monitor-alarm-arns)

(Opsional)

Tentukan Nama Sumber Daya Amazon (ARN) dari CloudWatch alarm Amazon untuk digunakan sebagai pemicu rollback. Misalnya, `arn:aws:cloudwatch::123456789012:alarm/MyAlarm`. Anda dapat memiliki maksimal lima pemicu rollback.

> **Note**
>
> Jika Anda menentukan ARN CloudWatch alarm, Anda juga harus mengonfigurasi izin tambahan untuk mengaktifkan tindakan untuk mengakses. CloudWatch Untuk informasi selengkapnya, lihat Mengkonfigurasi rollback.

UI yang sesuai: Tab Konfigurasi/Lanjutan/Monitor ARN alarm

monitor-timeout-in-minutes

(*DeployCloudFormationStack*/Configuration/monitor-timeout-in-minutes)

(Opsional)

Tentukan jumlah waktu, dari 0 hingga 180 menit, di mana CloudFormation memantau alarm yang ditentukan. Pemantauan dimulai setelah semua sumber daya tumpukan telah digunakan. Jika alarm terjadi dalam waktu pemantauan yang ditentukan, maka penerapan gagal, dan CloudFormation memutar kembali seluruh operasi tumpukan.

Default: 0. CloudFormation hanya memantau alarm saat sumber daya tumpukan sedang digunakan, bukan setelahnya.

UI yang sesuai: Tab konfigurasi/Lanjutan/Waktu pemantauan

tags

(*DeployCloudFormationStack*/Configuration/tags)

(Opsional)

Tentukan tag untuk dilampirkan ke CloudFormation tumpukan Anda. Tag adalah pasangan nilai kunci arbitrer yang dapat Anda gunakan untuk mengidentifikasi tumpukan Anda untuk tujuan seperti alokasi biaya. Untuk informasi lebih lanjut tentang apa itu tanda dan bagaimana cara penggunaannya, lihat Menandai sumber daya Anda di Panduan Pengguna Amazon EC2. Untuk informasi selengkapnya tentang penandaan CloudFormation, lihat Menyetel opsi AWS CloudFormation tumpukan di Panduan AWS CloudFormation Pengguna.

Sebuah kunci dapat memiliki karakter alfanumerik atau spasi, dan dapat memiliki hingga 127 karakter. Nilai dapat memiliki karakter alfanumerik atau spasi, dan dapat memiliki hingga 255 karakter.

Anda dapat menambahkan hingga 50 tag unik untuk setiap tumpukan.

UI yang sesuai: Tab Konfigurasi/Lanjutan/Tag

Menerapkan AWS Cloud Development Kit (AWS CDK) aplikasi dengan alur kerja

Bagian ini menjelaskan cara menerapkan AWS CDK aplikasi ke AWS akun Anda menggunakan alur kerja. Untuk mencapai ini, Anda harus menambahkan tindakan AWS CDK penerapan ke alur kerja Anda. Tindakan AWS CDK penerapan mensintesis dan menerapkan aplikasi Anda AWS Cloud Development Kit (AWS CDK) ke dalamnya. AWS Jika aplikasi Anda sudah ada AWS, tindakan akan memperbaruinya jika perlu.

Untuk informasi umum tentang menulis aplikasi menggunakan aplikasi AWS CDK, lihat Apa itu AWS CDK? di Panduan AWS Cloud Development Kit (AWS CDK) Pengembang.

Kapan menggunakan tindakan "AWS CDK menyebarkan”

Gunakan tindakan ini jika Anda telah mengembangkan aplikasi menggunakan aplikasi AWS CDK, dan sekarang Anda ingin menerapkannya secara otomatis sebagai bagian dari alur kerja integrasi dan pengiriman berkelanjutan (CI/CD) otomatis. Misalnya, Anda mungkin ingin menerapkan AWS CDK aplikasi secara otomatis setiap kali seseorang menggabungkan permintaan tarik yang terkait dengan sumber AWS CDK aplikasi Anda.

Bagaimana tindakan "AWS CDK menyebarkan” bekerja

AWS CDK Penyebaran berfungsi sebagai berikut:

1. Saat runtime, jika Anda menentukan versi 1.0.12 atau tindakan sebelumnya, tindakan akan mengunduh CDK CLI terbaru (juga disebut AWS CDK Tookit) ke image build. CodeCatalyst

   Jika Anda menentukan versi 1.0.13 atau yang lebih baru, tindakan tersebut dibundel dengan versi tertentu dari CDK CLI, jadi tidak ada unduhan yang terjadi.
2. Tindakan menggunakan CDK CLI untuk menjalankan `cdk deploy` perintah. Perintah ini mensintesis dan menerapkan AWS CDK aplikasi Anda ke dalam. AWS Untuk informasi selengkapnya tentang perintah ini, lihat topik AWS CDK Toolkit (perintah cdk) di Panduan AWS Cloud Development Kit (AWS CDK) Pengembang.

Versi CDK CLI yang digunakan oleh tindakan AWS CDK "menyebarkan”

Tabel berikut menunjukkan versi CDK CLI mana yang digunakan secara default oleh versi tindakan penerapan yang berbeda.AWS CDK

> **Note**
>
> Anda mungkin dapat mengganti default. Untuk informasi selengkapnya, lihat CdkCliVersion di Tindakan “AWS CDK menyebarkan” definisi YAMAL.

“AWS CDK menyebarkan” versi tindakan	AWS CDK Versi CLI
1.0.0 — 1.0.12	terbaru
1.0.13 atau lebih baru	2.99.1

Berapa banyak tumpukan yang dapat diterapkan tindakan?

AWS CDK Penyebaran hanya dapat menyebarkan satu tumpukan. Jika AWS CDK aplikasi Anda terdiri dari beberapa tumpukan, Anda harus membuat tumpukan induk dengan tumpukan bersarang, dan menerapkan induk menggunakan tindakan ini.

Topik

- Contoh alur kerja yang menerapkan aplikasi AWS CDK
- Menambahkan tindakan "AWS CDK menyebarkan”
- Variabel yang dihasilkan oleh tindakan "AWS CDK menyebarkan”
- Tindakan “AWS CDK menyebarkan” definisi YAMAL

Contoh alur kerja yang menerapkan aplikasi AWS CDK

Contoh alur kerja berikut mencakup tindakan AWS CDK penerapan, bersama dengan tindakan AWS CDK bootstrap. Alur kerja terdiri dari blok bangunan berikut yang berjalan secara berurutan:

> **Note**
>
> Contoh alur kerja berikut adalah untuk tujuan ilustrasi, dan tidak akan berfungsi tanpa konfigurasi tambahan. Ini dimaksudkan untuk memberi Anda contoh seperti apa alur kerja saat dikonfigurasi dengan tindakan AWS CDK penerapan.

- Pemicu - Pemicu ini memulai alur kerja yang dijalankan secara otomatis saat Anda mendorong perubahan ke repositori sumber Anda. Repositori ini berisi aplikasi Anda AWS CDK . Untuk informasi lebih lanjut tentang menggunakan pemicu, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.
- Tindakan AWS CDK bootstrap (`CDKBootstrap`) — Pada pemicu, tindakan menyebarkan tumpukan `CDKToolkit` bootstrap ke dalam AWS. Jika `CDKToolkit` tumpukan sudah ada di lingkungan, itu akan ditingkatkan jika perlu; jika tidak, tidak ada yang terjadi, dan tindakan ditandai sebagai berhasil.
- Tindakan AWS CDK penerapan (`AWS CDK Deploy`) — Setelah menyelesaikan tindakan AWS CDK bootstrap, tindakan penerapan mensintesis kode AWS CDK aplikasi Anda ke dalam AWS CloudFormation template dan AWS CDK menerapkan tumpukan yang ditentukan dalam template ke dalamnya. AWS

```
Name: codecatalyst-cdk-deploy-workflow
SchemaVersion: 1.0

Triggers:
  - Type: PUSH
    Branches:
      - main
Actions:
  CDKBootstrap:
    Identifier: aws/cdk-bootstrap@v1
    Inputs:
      Sources:
        - WorkflowSource
    Environment:
      Name: codecatalyst-cdk-deploy-environment
      Connections:
        - Name: codecatalyst-account-connection
          Role: codecatalyst-cdk-bootstrap-role
    Configuration:
```

```
      Region: us-west-2

  CDKDeploy:
    Identifier: aws/cdk-deploy@v1
    DependsOn:
      - CDKBootstrap
    Environment:
      Name: codecatalyst-cdk-deploy-environment
      Connections:
        - Name: codecatalyst-account-connection
          Role: codecatalyst-cdk-deploy-role
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      StackName: my-app-stack
      Region: us-west-2
```

Menambahkan tindakan "AWS CDK menyebarkan”

Gunakan petunjuk berikut untuk menambahkan tindakan AWS CDK penerapan ke alur kerja Anda.

Sebelum Anda memulai

Sebelum Anda dapat menambahkan tindakan AWS CDK penerapan ke alur kerja Anda, selesaikan tugas-tugas berikut:

1. Siapkan AWS CDK aplikasi. Anda dapat menulis AWS CDK aplikasi Anda menggunakan AWS CDK v1 atau v2, dalam bahasa pemrograman apa pun yang didukung oleh. AWS CDK Pastikan file AWS CDK aplikasi Anda tersedia di:
   - Sebuah repositori CodeCatalyst sumber, atau
   - Artefak CodeCatalyst keluaran yang dihasilkan oleh aksi alur kerja lain
2. Bootstrap AWS lingkungan Anda. Untuk bootstrap, Anda dapat:
   - Gunakan salah satu metode yang dijelaskan dalam Cara bootstrap di Panduan AWS Cloud Development Kit (AWS CDK) Pengembang.
   - Gunakan tindakan AWS CDK bootstrap. Anda dapat menambahkan tindakan ini dalam alur kerja yang sama dengan AWS CDK penerapan Anda, atau di alur kerja yang berbeda. Pastikan saja tindakan bootstrap berjalan setidaknya sekali sebelum menjalankan tindakan AWS CDK penerapan sehingga sumber daya yang diperlukan ada. Untuk informasi selengkapnya tentang tindakan AWS CDK bootstrap, lihatBootstrapping AWS CDK aplikasi dengan alur kerja.

Untuk informasi selengkapnya tentang bootstrap, lihat Bootstrapping di Panduan Pengembang.AWS Cloud Development Kit (AWS CDK)

Visual

Untuk menambahkan tindakan "AWS CDK menyebarkan” menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan AWS CDK penerapan, dan lakukan salah satu hal berikut:
   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih AWS CDK menyebarkan. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Unduh untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Pada tab Input dan Konfigurasi, lengkapi bidang sesuai dengan kebutuhan Anda. Untuk deskripsi setiap bidang, lihatTindakan “AWS CDK menyebarkan” definisi YAMAL. Referensi ini memberikan informasi rinci tentang setiap bidang (dan nilai properti YAMAL yang sesuai) seperti yang muncul di YAMAL dan editor visual.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, lalu pilih Komit lagi.

> **Note**
>
> Jika tindakan AWS CDK penerapan Anda gagal Bagaimana cara memperbaiki kesalahan “npm install”? karena `npm install` kesalahan, lihat informasi tentang cara memperbaiki kesalahan.

YAML

Untuk menambahkan tindakan "AWS CDK deploy” menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan AWS CDK penerapan, dan lakukan salah satu hal berikut:
   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau
   - Pilih AWS CDK menyebarkan. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Unduh untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Ubah properti dalam kode YAMAL sesuai dengan kebutuhan Anda. Penjelasan tentang setiap properti yang tersedia disediakan diTindakan “AWS CDK menyebarkan” definisi YAMAL.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.

12. Pilih Komit, masukkan pesan komit, lalu pilih Komit lagi.

> **Note**
>
> Jika tindakan AWS CDK penerapan Anda gagal Bagaimana cara memperbaiki kesalahan “npm install”? karena `npm install` kesalahan, lihat informasi tentang cara memperbaiki kesalahan.

Variabel yang dihasilkan oleh tindakan "AWS CDK menyebarkan”

Tindakan AWS CDK deploy menghasilkan dan menetapkan variabel berikut pada waktu berjalan. Ini dikenal sebagai variabel yang telah ditentukan.

Untuk informasi tentang mereferensikan variabel-variabel ini dalam alur kerja, lihat. Menggunakan variabel yang telah ditentukan

Kunci	Nilai
tumpukan-id	Nama Sumber Daya Amazon (ARN) dari tumpukan AWS CDK aplikasi yang digunakan selama alur kerja dijalankan. Contoh: `arn:aws:cloudformation:us-west-2:111122223333:stack/codecatalyst-cdk-app-stack/6aad4380-100a-11ec-a10a-03b8a84d40df`
platform penyebaran	Nama platform penyebaran. Hardcode ke. `AWS:CloudFormation`
region	Kode wilayah Wilayah AWS yang digunakan selama alur kerja dijalankan. Contoh: `us-west-2`
LEWATI PENYEBARAN	Nilai `true` menunjukkan bahwa penerapan tumpukan AWS CDK aplikasi Anda dilewati

Kunci	Nilai
	selama alur kerja dijalankan. Penerapan tumpukan akan dilewati jika tidak ada perubahan dalam tumpukan sejak penerapan terakhir. Variabel ini hanya diproduksi jika nilainya`true`. Hardcode ke. `true`
AWS CloudFormation variabel	Selain menghasilkan variabel yang tercantum sebelumnya, tindakan AWS CDK pener apan juga mengekspos variabel CloudForm ationkeluaran sebagai variabel alur kerja untuk digunakan dalam tindakan alur kerja berikutny a. Secara default, tindakan hanya mengekspo s empat (atau lebih sedikit) CloudForm ation variabel pertama yang ditemukannya. Untuk menentukan mana yang diekspos, jalankan tindakan AWS CDK deploy sekali, lalu lihat di tab Variables pada halaman run details. Jika variabel yang tercantum pada tab Variabel tidak seperti yang Anda inginkan, Anda dapat mengonfigurasi variabel yang berbeda menggunakan properti `CfnOutput Variables` YAMAL. Untuk informasi selengkapnya, lihat deskripsi CfnOutput Variables properti diTindakan “AWS CDK menyebarkan” definisi YAMAL.

Tindakan “AWS CDK menyebarkan” definisi YAMAL

Berikut ini adalah definisi YAMAL dari tindakan AWS CDK penerapan. Untuk mempelajari cara menggunakan tindakan ini, lihatMenerapkan AWS Cloud Development Kit (AWS CDK) aplikasi dengan alur kerja.

Definisi tindakan ini ada sebagai bagian dalam file definisi alur kerja yang lebih luas. Untuk informasi selengkapnya tentang file ini, lihatAlur kerja definisi YAMAL.

> **Note**
>
> Sebagian besar properti YAMB yang mengikuti memiliki elemen UI yang sesuai di editor visual. Untuk mencari elemen UI, gunakan Ctrl+F. Elemen akan terdaftar dengan properti YAMLnya yang terkait.

```
# The workflow definition starts here.
# See Properti tingkat atas for details.

Name: MyWorkflow
SchemaVersion: 1.0
Actions:

# The action definition starts here.
  CDKDeploy_nn:
    Identifier: aws/cdk-deploy@v1
    DependsOn:
      - CDKBootstrap
    Compute:
      Type: EC2 | Lambda
      Fleet: fleet-name
    Timeout: timeout-minutes
    Inputs:
      # Specify a source or an artifact, but not both.
      Sources:
        - source-name-1
      Artifacts:
        - artifact-name
    Outputs:
      Artifacts:
        - Name: cdk_artifact
          Files:
            - "cdk.out/**/*"
    Environment:
      Name: environment-name
      Connections:
        - Name: account-connection-name
          Role: iam-role-name
```

```
    Configuration:
      StackName: my-cdk-stack
      Region: us-west-2
      Tags: '{"key1": "value1", "key2": "value2"}'
      Context: '{"key1": "value1", "key2": "value2"}'
      CdkCliVersion: version
      CdkRootPath: directory-containing-cdk.json-file
      CfnOutputVariables: '["CnfOutputKey1","CfnOutputKey2","CfnOutputKey3"]'
      CloudAssemblyRootPath: path-to-cdk.out
```

CDKDeploy

(Diperlukan)

Tentukan nama tindakan. Semua nama tindakan harus unik dalam alur kerja. Nama aksi terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama tindakan.

Default: `CDKDeploy_nn`.

UI yang sesuai: Tab konfigurasi/Nama tindakan

Identifier

(*CDKDeploy*/Identifier)

(Diperlukan)

Mengidentifikasi tindakan. Jangan mengubah properti ini kecuali Anda ingin mengubah versi. Untuk informasi selengkapnya, lihat Menentukan versi mayor, minor, atau patch dari suatu tindakan.

Default: `aws/cdk-deploy@v1`.

UI yang sesuai: Diagram alur CDKDeploy kerja/_nn/ aws/cdk-deploy @v1 label

DependsOn

(*CDKDeploy*/DependsOn)

(Opsional)

Tentukan grup tindakan atau tindakan yang harus berjalan dengan sukses agar tindakan AWS CDK penerapan berjalan. Kami merekomendasikan untuk menentukan tindakan AWS CDK bootstrap di `DependsOn` properti, seperti ini:

```
CDKDeploy:
  Identifier: aws/cdk-deploy@v1
  DependsOn:
    - CDKBootstrap
```

> **Note**
>
> Bootstrapping adalah prasyarat wajib untuk menerapkan aplikasi. AWS CDK Jika Anda tidak menyertakan tindakan AWS CDK Bootstrap dalam alur kerja Anda, maka Anda harus menemukan cara lain untuk menyebarkan tumpukan AWS CDK bootstrap sebelum menjalankan tindakan AWS CDK penerapan Anda. Untuk informasi selengkapnya, lihat Menambahkan tindakan "AWS CDK menyebarkan" di Menerapkan AWS Cloud Development Kit (AWS CDK) aplikasi dengan alur kerja.

Untuk informasi selengkapnya tentang fungsionalitas 'tergantung pada', lihat. Mengkonfigurasi tindakan untuk bergantung pada tindakan lain

UI yang sesuai: Tab masukan/Tergantung pada - opsional

Compute

(*CDKDeploy*/Compute)

(Opsional)

Mesin komputasi yang digunakan untuk menjalankan tindakan alur kerja Anda. Anda dapat menentukan komputasi baik di tingkat alur kerja atau di tingkat tindakan, tetapi tidak keduanya. Ketika ditentukan pada tingkat alur kerja, konfigurasi komputasi berlaku untuk semua tindakan yang ditentukan dalam alur kerja. Pada tingkat alur kerja, Anda juga dapat menjalankan beberapa tindakan pada instance yang sama. Untuk informasi selengkapnya, lihat Berbagi komputasi di seluruh tindakan.

UI yang sesuai: tidak ada

Type

(*CDKDeploy*/Compute/Type)

(Diperlukan Compute jika disertakan)

Jenis mesin komputasi. Anda dapat menggunakan salah satu nilai berikut:

- EC2 (editor visual) atau `EC2` (editor YAMB)

  Dioptimalkan untuk fleksibilitas selama aksi berjalan.
- Lambda (editor visual) atau `Lambda` (editor YAMB)

  Kecepatan start-up aksi yang dioptimalkan.

Untuk informasi selengkapnya tentang jenis komputasi, lihatJenis komputasi.

UI yang sesuai: Tab Konfigurasi/Tingkat Lanjut - opsional/Jenis komputasi

Fleet

(*CDKDeploy*/Compute/Fleet)

(Opsional)

Tentukan mesin atau armada yang akan menjalankan alur kerja atau tindakan alur kerja Anda. Dengan armada sesuai permintaan, ketika suatu tindakan dimulai, alur kerja menyediakan sumber daya yang dibutuhkan, dan mesin dihancurkan ketika tindakan selesai. Contoh armada sesuai permintaan:`Linux.x86-64.Large`,. `Linux.x86-64.XLarge` Untuk informasi lebih lanjut tentang armada sesuai permintaan, lihat. Properti armada sesuai permintaan

Dengan armada yang disediakan, Anda mengonfigurasi satu set mesin khusus untuk menjalankan tindakan alur kerja Anda. Mesin-mesin ini tetap menganggur, siap untuk memproses tindakan segera. Untuk informasi selengkapnya tentang armada yang disediakan, lihat. Properti armada yang disediakan

Jika `Fleet` dihilangkan, defaultnya adalah. `Linux.x86-64.Large`

UI yang sesuai: Tab Konfigurasi/Advanced - armada opsional/Komputasi

Timeout

(*CDKDeploy*/Timeout)

(Diperlukan)

Tentukan jumlah waktu dalam menit (editor YAMB), atau jam dan menit (editor visual), bahwa tindakan dapat berjalan sebelum CodeCatalyst mengakhiri tindakan. Minimal adalah 5 menit dan maksimum dijelaskan dalamKuota untuk alur kerja. Batas waktu default sama dengan batas waktu maksimum.

UI yang sesuai: Tab konfigurasi/Timeout - opsional

Inputs

(*CDKDeploy*/Inputs)

(Opsional)

`Inputs`Bagian ini mendefinisikan data yang `CDKDeploy` dibutuhkan selama menjalankan alur kerja.

> **Note**
>
> Hanya satu input (baik sumber atau artefak) yang diizinkan untuk setiap tindakan AWS CDK penerapan.

UI yang sesuai: Tab input

Sources

(*CDKDeploy*/Inputs/Sources)

(Diperlukan jika AWS CDK aplikasi yang ingin Anda terapkan disimpan dalam repositori sumber)

Jika AWS CDK aplikasi Anda disimpan dalam repositori sumber, tentukan label repositori sumber tersebut. Tindakan AWS CDK penerapan mensintesis aplikasi di repositori ini sebelum memulai proses penerapan. Saat ini, satu-satunya label yang didukung adalah`WorkflowSource`.

Jika AWS CDK aplikasi Anda tidak terkandung dalam repositori sumber, itu harus berada dalam artefak yang dihasilkan oleh tindakan lain.

Untuk informasi selengkapnya tentang sumber, lihat Menghubungkan alur kerja ke repositori sumber.

UI yang sesuai: Tab/Sumber Input - opsional

Artifacts - input

(*CDKDeploy*/Inputs/Artifacts)

(Diperlukan jika AWS CDK aplikasi yang ingin Anda terapkan disimpan dalam artefak keluaran dari tindakan sebelumnya)

Jika AWS CDK aplikasi Anda terkandung dalam artefak yang dihasilkan oleh tindakan sebelumnya, tentukan artefak tersebut di sini. Tindakan AWS CDK penerapan mensintesis aplikasi dalam artefak yang ditentukan ke dalam CloudFormation templat sebelum memulai proses penerapan. Jika AWS CDK aplikasi Anda tidak terkandung dalam artefak, itu harus berada di repositori sumber Anda.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Input tab/Artefak - opsional

Outputs

(*CDKDeploy*/Outputs)

(Opsional)

Mendefinisikan data yang dihasilkan oleh tindakan selama menjalankan alur kerja.

UI yang sesuai: Tab keluaran

Artifacts - output

(*CDKDeploy*/Outputs/Artifacts

(Opsional)

Tentukan artefak yang dihasilkan oleh tindakan. Anda dapat mereferensikan artefak ini sebagai masukan dalam tindakan lain.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab keluaran/Artefak

Name

(*CDKDeploy*/Outputs/Artifacts/Name)

(Diperlukan Artifacts - output jika disertakan)

Tentukan nama artefak yang akan berisi AWS CloudFormation template yang disintesis oleh tindakan AWS CDK penerapan saat runtime. Nilai default-nya adalah `cdk_artifact`. Jika Anda tidak menentukan artefak, maka tindakan mensintesis template tetapi tidak akan menyimpannya dalam artefak. Pertimbangkan untuk menyimpan template yang disintesis dalam artefak untuk menyimpan catatannya untuk tujuan pengujian atau pemecahan masalah.

UI yang sesuai: Tab Keluaran/Artefak/Tambahkan artefak/Bangun nama artefak

Files

(*CDKDeploy*/Outputs/Artifacts/Files)

(Diperlukan Artifacts - output jika disertakan)

Tentukan file yang akan disertakan dalam artefak. Anda harus menentukan `"cdk.out/**/*"` untuk menyertakan AWS CloudFormation template yang disintesis AWS CDK aplikasi Anda.

> **Note**
>
> `cdk.out`adalah direktori default tempat file yang disintesis disimpan. Jika Anda menentukan direktori keluaran selain `cdk.out` di `cdk.json` file Anda, tentukan direktori itu di sini, bukan`cdk.out`.

UI yang sesuai: Tab Keluaran/Artefak/Tambahkan artefak/File yang dihasilkan oleh build

Environment

(*CDKDeploy*/Environment)

(Diperlukan)

Tentukan CodeCatalyst lingkungan yang akan digunakan dengan tindakan.

Untuk informasi lebih lanjut tentang lingkungan, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst danPembuatan lingkungan.

UI yang sesuai: Tab Konfigurasi/'Lingkungan/Akun/Peran'/Lingkungan

Name

(*CDKDeploy*/Environment/Name)

(Diperlukan Environment jika disertakan)

Tentukan nama lingkungan yang ada yang ingin Anda kaitkan dengan tindakan.

UI yang sesuai: Tab Konfigurasi/'Lingkungan/Akun/Peran'/Lingkungan

Connections

(*CDKDeploy*/Environment/Connections)

(Diperlukan Environment jika disertakan)

Tentukan koneksi akun untuk dikaitkan dengan tindakan. Anda dapat menentukan maksimum satu koneksi akun di bawah`Environment`.

Untuk informasi selengkapnya tentang koneksi akun, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS. Untuk informasi tentang cara mengaitkan koneksi akun dengan lingkungan Anda, lihatPembuatan lingkungan.

UI yang sesuai: Tab konfigurasi/'lingkungan/akun/peran'/koneksi akun AWS

Name

(*CDKDeploy*/Environment/Connections/Name)

(Diperlukan)

Tentukan nama koneksi akun.

UI yang sesuai: Tab konfigurasi/'lingkungan/akun/peran'/koneksi akun AWS

Role

(*CDKDeploy*/Environment/Connections/Role)

(Diperlukan)

Tentukan nama peran IAM yang AWS CDK digunakan tindakan penerapan untuk mengakses AWS dan menyebarkan tumpukan aplikasi. AWS CDK Pastikan peran ini meliputi:

- Kebijakan izin berikut:

> **Warning**
>
> Batasi izin untuk yang ditampilkan dalam kebijakan berikut. Menggunakan peran dengan izin yang lebih luas dapat menimbulkan risiko keamanan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "cloudformation:DescribeStackEvents",
                "cloudformation:DescribeChangeSet",
                "cloudformation:DescribeStacks",
                "cloudformation:ListStackResources"
            ],
            "Resource": "*"
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::aws-account:role/cdk-*"
        }
    ]
}
```

- Kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
```

```
                    "codecatalyst.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Pastikan peran ini ditambahkan ke koneksi akun Anda. Untuk mempelajari selengkapnya tentang menambahkan peran IAM ke koneksi akun, lihatMenambahkan peran IAM ke koneksi akun.

> **Note**
>
> Anda dapat menentukan nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran di sini, jika Anda mau. Untuk informasi selengkapnya tentang peran ini, silakan lihat Membuat CodeCatalystWorkflowDevelopmentRole-***spaceName***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami menyarankan Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian.

UI yang sesuai: Tab Konfigurasi/'lingkungan/akun/peran'/ Peran

Configuration

(*CDKDeploy*/Configuration)

(Diperlukan)

Bagian di mana Anda dapat menentukan properti konfigurasi tindakan.

UI yang sesuai: Tab konfigurasi

StackName

(*CDKDeploy*/Configuration/StackName)

(Diperlukan)

Nama tumpukan AWS CDK aplikasi Anda, seperti yang muncul di file entrypoint di direktori AWS CDK aplikasi Anda. `bin` Contoh berikut menunjukkan isi dari file TypeScript entrypoint, dengan nama

stack disorot dalam huruf miring *merah*. Jika file entrypoint Anda dalam bahasa yang berbeda, itu akan terlihat serupa.

```
import * as cdk from 'aws-cdk-lib';
import { CdkWorksopTypescriptStack } from '../lib/cdk_workshop_typescript-stack';

const app = new cdk.App();
new CdkWorkshopTypescriptStack(app, 'CdkWorkshopTypescriptStack');
```

Anda hanya dapat menentukan satu tumpukan.

> **Tip**
>
> Jika Anda memiliki beberapa tumpukan, Anda dapat membuat tumpukan induk dengan tumpukan bersarang. Anda kemudian dapat menentukan tumpukan induk dalam tindakan ini untuk menerapkan semua tumpukan.

UI yang sesuai: Tab konfigurasi/Nama tumpukan

Region

(*CDKDeploy*/Configuration/Region)

(Diperlukan)

Tentukan Wilayah AWS ke mana tumpukan AWS CDK aplikasi akan digunakan. Untuk daftar kode Wilayah, lihat Titik akhir Regional.

UI yang sesuai: Tab konfigurasi/Wilayah

Tags

(*CDKDeploy*/Configuration/Tags)

(Opsional)

Tentukan tag yang ingin Anda terapkan ke AWS sumber daya di tumpukan AWS CDK aplikasi. Tag diterapkan ke tumpukan itu sendiri serta sumber daya individu di tumpukan. Untuk informasi selengkapnya tentang penandaan, lihat Menandai di Panduan AWS Cloud Development Kit (AWS CDK) Pengembang.

UI yang sesuai: Tab Konfigurasi/Lanjutan - opsional/Tag

Context

(*CDKDeploy*/Configuration/Context)

(Opsional)

Tentukan konteks, dalam bentuk pasangan kunci-nilai, untuk dikaitkan dengan tumpukan aplikasi. AWS CDK Untuk informasi selengkapnya tentang konteks, lihat konteks Runtime di Panduan Pengembang.AWS Cloud Development Kit (AWS CDK)

UI yang sesuai: Tab Konfigurasi/Lanjutan - opsional/Konteks

CdkCliVersion

(*CDKDeploy*/Configuration/CdkCliVersion)

(Opsional)

Properti ini tersedia dengan versi 1.0.13 atau yang lebih baru dari tindakan AWS CDK penerapan, dan versi 1.0.8 atau yang lebih baru dari tindakan bootstrap.AWS CDK

Tentukan satu dari yang berikut ini:

- Versi lengkap dari AWS Cloud Development Kit (AWS CDK) Command Line Interface (CLI) (juga disebut AWS CDK Toolkit) yang Anda ingin tindakan ini untuk digunakan. Contoh:`2.102.1`. Pertimbangkan untuk menentukan versi lengkap untuk memastikan konsistensi dan stabilitas saat membangun dan menerapkan aplikasi Anda.

  Atau

- `latest`. Pertimbangkan `latest` untuk menentukan untuk memanfaatkan fitur dan perbaikan terbaru CDK CLI.

Tindakan akan mengunduh versi AWS CDK CLI yang ditentukan (atau versi terbaru) ke gambar CodeCatalyst build, dan kemudian menggunakan versi ini untuk menjalankan perintah yang diperlukan untuk menyebarkan aplikasi CDK Anda atau mem-bootstrap lingkungan Anda. AWS

Untuk daftar versi CDK CLI yang didukung yang dapat Anda gunakan,AWS CDK lihat Versi.

Jika Anda menghilangkan properti ini, tindakan menggunakan versi AWS CDK CLI default yang dijelaskan dalam salah satu topik berikut:

- Versi CDK CLI yang digunakan oleh tindakan AWS CDK "menyebarkan”
- Versi CDK CLI yang digunakan oleh tindakan AWS CDK "bootstrap”

UI yang sesuai: Tab konfigurasi/versi AWS CDK CLI

CdkRootPath

(*CDKDeploy*/Configuration/CdkRootPath)

(Opsional)

Path ke direktori yang berisi `cdk.json` file AWS CDK proyek Anda. Tindakan AWS CDK penyebaran berjalan dari folder ini, dan output apa pun yang dibuat oleh tindakan akan ditambahkan ke direktori ini. Jika tidak ditentukan, tindakan AWS CDK penerapan mengasumsikan bahwa `cdk.json` file tersebut ada di root proyek Anda. AWS CDK

UI yang sesuai: Tab konfigurasi/Direktori tempat cdk.json berada

CfnOutputVariables

(*CDKDeploy*/Configuration/CfnOutputVariables)

(Opsional)

Tentukan `CfnOutput` konstruksi mana dalam kode AWS CDK aplikasi yang ingin Anda paparkan sebagai variabel keluaran alur kerja. Anda kemudian dapat mereferensikan variabel keluaran alur kerja dalam tindakan selanjutnya dalam alur kerja Anda. Untuk informasi lebih lanjut tentang variabel di CodeCatalyst, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

Misalnya, jika kode AWS CDK aplikasi Anda terlihat seperti ini:

```
import { Duration, Stack, StackProps, CfnOutput, RemovalPolicy} from 'aws-cdk-lib';
import * as dynamodb from 'aws-cdk-lib/aws-dynamodb';
import * as s3 from 'aws-cdk-lib/aws-s3';
import { Construct } from 'constructs';
import * as cdk from 'aws-cdk-lib';
export class HelloCdkStack extends Stack {
  constructor(scope: Construct, id: string, props?: StackProps) {
    super(scope, id, props);
    const bucket = new s3.Bucket(this, 'my-bucket', {
      removalPolicy: RemovalPolicy.DESTROY,
    });
    new CfnOutput(this, 'bucketName', {
```

```
      value: bucket.bucketName,
      description: 'The name of the s3 bucket',
      exportName: 'myBucket',
    });
    const table = new dynamodb.Table(this, 'todos-table', {
      partitionKey: {name: 'todoId', type: dynamodb.AttributeType.NUMBER},
      billingMode: dynamodb.BillingMode.PAY_PER_REQUEST,
      removalPolicy: RemovalPolicy.DESTROY,
    })
    new CfnOutput(this, 'tableName', {
      value: table.tableName,
      description: 'The name of the dynamodb table',
      exportName: 'myDynamoDbTable',
    });
    ...
  }
}
```

... dan `CfnOutputVariables` properti Anda terlihat seperti ini:

```
Configuration:
  ...
  CfnOutputVariables: '["bucketName","tableName"]'
```

... maka tindakan menghasilkan variabel output alur kerja berikut:

Kunci	Nilai
bucketName	`bucket.bucketName`
tableName	`table.tableName`

Anda kemudian dapat mereferensikan `tableName` variabel `bucketName` dan dalam tindakan selanjutnya. Untuk mempelajari cara mereferensikan variabel keluaran alur kerja dalam tindakan selanjutnya, lihatMereferensikan variabel yang telah ditentukan.

Jika Anda tidak menentukan `CfnOutput` konstruksi apa pun di `CfnOutputVariables` properti, maka tindakan tersebut mengekspos empat (atau lebih sedikit) variabel CloudFormation keluaran pertama yang ditemukannya sebagai variabel keluaran alur kerja. Untuk informasi selengkapnya, lihat Variabel yang dihasilkan oleh tindakan "AWS CDK menyebarkan" .

> **Tip**
>
> Untuk mendapatkan daftar semua variabel CloudFormation keluaran yang dihasilkan tindakan, jalankan alur kerja yang berisi tindakan AWS CDK penerapan sekali, lalu lihat di tab Log tindakan. Log berisi daftar semua variabel CloudFormation keluaran yang terkait dengan AWS CDK aplikasi Anda. Setelah Anda tahu apa semua CloudFormation variabel, Anda dapat menentukan mana yang ingin Anda konversi ke variabel output alur kerja menggunakan `CfnOutputVariables` properti.

Untuk informasi selengkapnya tentang variabel AWS CloudFormation keluaran, lihat dokumentasi untuk `CfnOutput` konstruksi, yang tersedia di class CfnOutput (construct) di Referensi AWS Cloud Development Kit (AWS CDK) API.

UI yang sesuai: Variabel AWS CloudFormation tab/output konfigurasi

CloudAssemblyRootPath

(*CDKDeploy*/Configuration/CloudAssemblyRootPath)

(Opsional)

Jika Anda telah mensintesis tumpukan AWS CDK aplikasi ke dalam rakitan cloud (menggunakan `cdk synth` operasi), tentukan jalur root direktori perakitan cloud (`cdk.out`). AWS CloudFormation Template yang terletak di direktori perakitan cloud yang ditentukan akan digunakan oleh tindakan AWS CDK penerapan ke dalam Anda Akun AWS menggunakan perintah. `cdk deploy --app` Ketika `--app` opsi hadir, `cdk synth` operasi tidak terjadi.

Jika Anda tidak menentukan direktori perakitan cloud, maka tindakan AWS CDK penerapan akan menjalankan `cdk deploy` perintah tanpa `--app` opsi. Tanpa `--app` opsi, `cdk deploy` operasi akan mensintesis (`cdk synth`) dan menerapkan aplikasi Anda ke dalam AWS CDK aplikasi Anda. Akun AWS

Mengapa saya menentukan rakitan cloud yang sudah ada dan disintesis ketika tindakan "AWS CDK penerapan” dapat melakukan sintesis pada waktu berjalan?

Anda mungkin ingin menentukan rakitan cloud yang sudah ada dan disintesis ke:

- Pastikan bahwa kumpulan sumber daya yang sama persis digunakan setiap kali tindakan "AWS CDK penerapan” berjalan

Jika Anda tidak menentukan rakitan cloud, tindakan AWS CDK penerapan dapat mensintesis dan menyebarkan file yang berbeda tergantung pada kapan dijalankan. Misalnya, tindakan AWS CDK penerapan mungkin mensintesis perakitan cloud dengan satu set dependensi selama tahap pengujian, dan kumpulan dependensi lainnya selama tahap produksi (jika dependensi tersebut berubah antar tahapan). Untuk menjamin paritas yang tepat antara apa yang diuji dan apa yang diterapkan, kami sarankan untuk mensintesis sekali dan kemudian menggunakan bidang direktori perakitan Path to cloud (editor visual) atau `CloudAssemblyRootPath` properti (editor YAMAL) untuk menentukan rakitan cloud yang sudah disintesis.

- Gunakan pengelola dan perkakas paket non-standar dengan aplikasi AWS CDK

  Selama `synth` operasi, tindakan AWS CDK penerapan mencoba menjalankan aplikasi Anda menggunakan alat standar seperti npm atau pip. Jika tindakan tidak berhasil menjalankan aplikasi Anda menggunakan alat tersebut, sintesis tidak akan terjadi dan tindakan akan gagal. Untuk mengatasi masalah ini, Anda dapat menentukan perintah persis yang diperlukan untuk menjalankan aplikasi dengan sukses di `cdk.json` file AWS CDK aplikasi, lalu mensintesis aplikasi Anda menggunakan metode yang tidak melibatkan tindakan AWS CDK penerapan. Setelah perakitan cloud dibuat, Anda dapat menentukannya di bidang direktori Path to cloud assembly (editor visual) atau `CloudAssemblyRootPath` properti (editor YAMAL) dari tindakan AWS CDK penerapan.

Untuk informasi tentang mengonfigurasi `cdk.json` file agar menyertakan perintah untuk menginstal dan menjalankan AWS CDK aplikasi, lihat Menentukan perintah aplikasi.

Untuk informasi tentang *`cdk synth`* perintah *`cdk deploy`* dan, serta *`--app`* opsi, lihat Menerapkan tumpukan, Mensintesis tumpukan, dan Melewati sintesis di Panduan Pengembang.AWS Cloud Development Kit (AWS CDK)

Untuk informasi tentang rakitan cloud, lihat Cloud Assembly di Referensi AWS Cloud Development Kit (AWS CDK) API.

UI yang sesuai: Tab konfigurasi/Jalur ke direktori perakitan cloud

Bootstrapping AWS CDK aplikasi dengan alur kerja

Bagian ini menjelaskan cara bootstrap AWS CDK aplikasi menggunakan CodeCatalyst alur kerja. Untuk mencapai ini, Anda harus menambahkan tindakan AWS CDK bootstrap ke alur kerja Anda. Tindakan AWS CDK bootstrap menyediakan tumpukan bootstrap di AWS lingkungan

Anda menggunakan template modern. Jika tumpukan bootstrap sudah ada, tindakan akan memperbaruinya jika perlu. Memiliki tumpukan bootstrap yang ada di dalamnya AWS adalah prasyarat untuk menerapkan aplikasi. AWS CDK

Untuk informasi selengkapnya tentang bootstrap, lihat Bootstrapping di Panduan Pengembang.AWS Cloud Development Kit (AWS CDK)

Kapan menggunakan tindakan "AWS CDK bootstrap”

Gunakan tindakan ini jika Anda memiliki alur kerja yang menerapkan AWS CDK aplikasi, dan Anda ingin menerapkan (dan memperbarui, jika perlu) tumpukan bootstrap secara bersamaan. Dalam kasus ini, Anda akan menambahkan tindakan AWS CDK bootstrap ke alur kerja yang sama dengan yang menerapkan aplikasi Anda AWS CDK .

Jangan gunakan tindakan ini jika salah satu dari berikut ini berlaku:

- Anda sudah menerapkan tumpukan bootstrap menggunakan mekanisme lain, dan Anda ingin menjaganya tetap utuh (tidak ada pembaruan).
- Anda ingin menggunakan template bootstrap khusus, yang tidak didukung dengan tindakan AWS CDK bootstrap.

Bagaimana tindakan "AWS CDK bootstrap” bekerja

AWS CDK Bootstrap bekerja sebagai berikut:

1. Saat runtime, jika Anda menentukan versi 1.0.7 atau tindakan sebelumnya, tindakan akan mengunduh CDK CLI terbaru (juga disebut AWS CDK Tookit) ke image build. CodeCatalyst

   Jika Anda menentukan versi 1.0.8 atau yang lebih baru, tindakan tersebut dibundel dengan versi tertentu dari CDK CLI, jadi tidak ada unduhan yang terjadi.
2. Tindakan ini menggunakan CDK CLI untuk menjalankan `cdk bootstrap` perintah. Perintah ini melakukan tugas bootstrap yang dijelaskan dalam topik Bootstrapping di Panduan Pengembang.AWS Cloud Development Kit (AWS CDK)

Versi CDK CLI yang digunakan oleh tindakan AWS CDK "bootstrap”

Tabel berikut menunjukkan versi CDK CLI mana yang digunakan secara default oleh berbagai versi tindakan bootstrap AWS CDK .

> **Note**
>
> Anda mungkin dapat mengganti default. Untuk informasi selengkapnya, lihat CdkCliVersion di Tindakan “AWS CDK bootstrap” definisi YAMAL.

“AWS CDK bootstrap” versi tindakan	AWS CDK Versi CLI
1.0.0 — 1.0.7	terbaru
1.0.8 atau yang lebih baru	2.99.1

Topik

- Contoh alur kerja yang mem-bootstrap aplikasi AWS CDK
- Menambahkan tindakan "AWS CDK bootstrap”
- Variabel yang dihasilkan oleh tindakan "AWS CDK bootstrap”
- Tindakan “AWS CDK bootstrap” definisi YAMAL

Contoh alur kerja yang mem-bootstrap aplikasi AWS CDK

Lihat Contoh alur kerja yang menerapkan aplikasi AWS CDK di Menerapkan AWS Cloud Development Kit (AWS CDK) aplikasi dengan alur kerja untuk alur kerja yang mencakup tindakan AWS CDK bootstrap.

Menambahkan tindakan "AWS CDK bootstrap”

Gunakan petunjuk berikut untuk menambahkan tindakan AWS CDK bootstrap ke alur kerja Anda.

Sebelum Anda memulai

Sebelum Anda dapat menggunakan tindakan AWS CDK bootstrap, pastikan Anda memiliki AWS CDK aplikasi yang siap. Tindakan bootstrap akan mensintesis AWS CDK aplikasi sebelum bootstrap. Anda dapat menulis aplikasi Anda dalam bahasa pemrograman apa pun yang didukung oleh AWS CDK.

Pastikan file AWS CDK aplikasi Anda tersedia di:

- Sebuah repositori CodeCatalyst sumber, atau
- Artefak CodeCatalyst keluaran yang dihasilkan oleh aksi alur kerja lain

Visual

Untuk menambahkan tindakan "AWS CDK bootstrap” menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan AWS CDK bootstrap, dan lakukan salah satu hal berikut:
   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih AWS CDK bootstrap. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Lihat sumber untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Pada tab Input, Konfigurasi, dan Output, lengkapi bidang sesuai dengan kebutuhan Anda. Untuk deskripsi setiap bidang, lihatTindakan “AWS CDK bootstrap” definisi YAMAL. Referensi ini memberikan informasi rinci tentang setiap bidang (dan nilai properti YAMAL yang sesuai) seperti yang muncul di YAMAL dan editor visual.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, lalu pilih Komit lagi.

> **Note**
>
> Jika tindakan AWS CDK bootstrap Anda gagal dengan `npm install` kesalahan, lihat Bagaimana cara memperbaiki kesalahan “npm install”? untuk informasi tentang cara memperbaiki kesalahan.

YAML

Untuk menambahkan tindakan "AWS CDK bootstrap” menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan AWS CDK bootstrap, dan pilih + untuk menambahkannya ke diagram alur kerja dan buka panel konfigurasinya.
10. Ubah properti dalam kode YAMAL sesuai dengan kebutuhan Anda. Penjelasan tentang setiap properti yang tersedia disediakan diTindakan “AWS CDK bootstrap” definisi YAMAL.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, lalu pilih Komit lagi.

> **Note**
>
> Jika tindakan AWS CDK bootstrap Anda gagal dengan `npm install` kesalahan, lihat Bagaimana cara memperbaiki kesalahan “npm install”? untuk informasi tentang cara memperbaiki kesalahan.

Variabel yang dihasilkan oleh tindakan "AWS CDK bootstrap”

Tindakan AWS CDK bootstrap menghasilkan dan menetapkan variabel berikut pada waktu berjalan. Ini dikenal sebagai variabel yang telah ditentukan.

Untuk informasi tentang mereferensikan variabel-variabel ini dalam alur kerja, lihat. Menggunakan variabel yang telah ditentukan

Kunci	Nilai
platform penyebaran	Nama platform penyebaran. Hardcode ke. `AWS:CloudFormation`
region	Kode wilayah tempat Wilayah AWS tumpukan AWS CDK bootstrap digunakan selama alur kerja dijalankan. Contoh: `us-west-2`
tumpukan-id	Nama Sumber Daya Amazon (ARN) dari tumpukan bootstrap yang diterapkan AWS CDK . Contoh: `arn:aws:cloudformation:us-west-2:111122223333:stack/codecatalyst-cdk-bootstrap-stack/6aad4380-100a-11ec-a10a-03b8a84d40df`
LEWATI PENYEBARAN	Nilai `true` menunjukkan bahwa penerapan tumpukan AWS CDK bootstrap Anda dilewati selama alur kerja dijalankan. Penerapan tumpukan akan dilewati jika tidak ada perubahan dalam tumpukan sejak penerapan terakhir. Variabel ini hanya diproduksi jika nilainya`true`. Hardcode ke. `true`

Tindakan “AWS CDK bootstrap” definisi YAMAL

Berikut ini adalah definisi YAMAL dari tindakan AWS CDK bootstrap. Untuk mempelajari cara menggunakan tindakan ini, lihatBootstrapping AWS CDK aplikasi dengan alur kerja.

Definisi tindakan ini ada sebagai bagian dalam file definisi alur kerja yang lebih luas. Untuk informasi selengkapnya tentang file ini, lihatAlur kerja definisi YAMAL.

> **Note**
>
> Sebagian besar properti YAMB yang mengikuti memiliki elemen UI yang sesuai di editor visual. Untuk mencari elemen UI, gunakan Ctrl+F. Elemen akan terdaftar dengan properti YAMLnya yang terkait.

```
# The workflow definition starts here.
# See Properti tingkat atas for details.

Name: MyWorkflow
SchemaVersion: 1.0
Actions:

# The action definition starts here.
  CDKBootstrapAction_nn:
    Identifier: aws/cdk-bootstrap@v1
    DependsOn:
      - action-name
    Compute:
      Type: EC2 | Lambda
      Fleet: fleet-name
    Timeout: timeout-minutes
    Inputs:
      # Specify a source or an artifact, but not both.
      Sources:
        - source-name-1
      Artifacts:
        - artifact-name
    Outputs:
      Artifacts:
        - Name: cdk_bootstrap_artifacts
          Files:
            - "cdk.out/**/*"
```

```
    Environment:
      Name: environment-name
      Connections:
        - Name: account-connection-name
          Role: iam-role-name
    Configuration:
      Region: us-west-2
      CdkCliVersion: version
```

CDKBootstrapAction

(Diperlukan)

Tentukan nama tindakan. Semua nama tindakan harus unik dalam alur kerja. Nama aksi terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama tindakan.

Default: `CDKBootstrapAction_nn`.

UI yang sesuai: Tab konfigurasi/Nama tampilan tindakan

Identifier

(*CDKBootstrapAction*/Identifier)

(Diperlukan)

Mengidentifikasi tindakan. Jangan mengubah properti ini kecuali Anda ingin mengubah versi. Untuk informasi selengkapnya, lihat Menentukan versi mayor, minor, atau patch dari suatu tindakan.

Default: `aws/cdk-bootstrap@v1`.

UI yang sesuai: Diagram alur CDKBootstrapAction kerja/_nn/ aws/cdk-bootstrap @v1 label

DependsOn

(*CDKBootstrapAction*/DependsOn)

(Opsional)

Tentukan tindakan, grup tindakan, atau gerbang yang harus berjalan dengan sukses agar tindakan ini berjalan.

Untuk informasi selengkapnya tentang fungsionalitas 'tergantung pada', lihat. Mengkonfigurasi tindakan untuk bergantung pada tindakan lain

UI yang sesuai: Tab masukan/Tergantung pada - opsional

Compute

(*CDKBootstrapAction*/Compute)

(Opsional)

Mesin komputasi yang digunakan untuk menjalankan tindakan alur kerja Anda. Anda dapat menentukan komputasi baik di tingkat alur kerja atau di tingkat tindakan, tetapi tidak keduanya. Ketika ditentukan pada tingkat alur kerja, konfigurasi komputasi berlaku untuk semua tindakan yang ditentukan dalam alur kerja. Pada tingkat alur kerja, Anda juga dapat menjalankan beberapa tindakan pada instance yang sama. Untuk informasi selengkapnya, lihat Berbagi komputasi di seluruh tindakan.

UI yang sesuai: tidak ada

Type

(*CDKBootstrapAction*/Compute/Type)

(Diperlukan Compute jika disertakan)

Jenis mesin komputasi. Anda dapat menggunakan salah satu nilai berikut:

- EC2 (editor visual) atau `EC2` (editor YAMG)

  Dioptimalkan untuk fleksibilitas selama aksi berjalan.
- Lambda (editor visual) atau `Lambda` (editor YAMG)

  Kecepatan start-up aksi yang dioptimalkan.

Untuk informasi selengkapnya tentang jenis komputasi, lihatJenis komputasi.

UI yang sesuai: Tab Konfigurasi/Tingkat Lanjut - opsional/Jenis komputasi

Fleet

(*CDKBootstrapAction*/Compute/Fleet)

(Opsional)

Tentukan mesin atau armada yang akan menjalankan alur kerja atau tindakan alur kerja Anda. Dengan armada sesuai permintaan, ketika suatu tindakan dimulai, alur kerja menyediakan sumber daya yang dibutuhkan, dan mesin dihancurkan ketika tindakan selesai. Contoh armada sesuai permintaan:`Linux.x86-64.Large`,. `Linux.x86-64.XLarge` Untuk informasi lebih lanjut tentang armada sesuai permintaan, lihat. Properti armada sesuai permintaan

Dengan armada yang disediakan, Anda mengonfigurasi satu set mesin khusus untuk menjalankan tindakan alur kerja Anda. Mesin-mesin ini tetap menganggur, siap untuk memproses tindakan segera. Untuk informasi selengkapnya tentang armada yang disediakan, lihat. Properti armada yang disediakan

Jika `Fleet` dihilangkan, defaultnya adalah. `Linux.x86-64.Large`

UI yang sesuai: Tab Konfigurasi/Advanced - armada opsional/Komputasi

Timeout

(*`CDKBootstrapAction`*/Timeout)

(Diperlukan)

Tentukan jumlah waktu dalam menit (editor YAMB), atau jam dan menit (editor visual), bahwa tindakan dapat berjalan sebelum CodeCatalyst mengakhiri tindakan. Minimal adalah 5 menit dan maksimum dijelaskan dalamKuota untuk alur kerja. Batas waktu default sama dengan batas waktu maksimum.

UI yang sesuai: Tab konfigurasi/Timeout - opsional

Inputs

(*`CDKBootstrapAction`*/Inputs)

(Opsional)

`Inputs`Bagian ini mendefinisikan data yang dibutuhkan tindakan AWS CDK bootstrap selama menjalankan alur kerja.

UI yang sesuai: Tab input

> **Note**
>
> Hanya satu input (baik sumber atau artefak) yang diizinkan untuk setiap tindakan AWS CDK bootstrap.

Sources

(*CDKBootstrapAction*/Inputs/Sources)

(Diperlukan jika AWS CDK aplikasi Anda disimpan dalam repositori sumber)

Jika AWS CDK aplikasi Anda disimpan dalam repositori sumber, tentukan label repositori sumber tersebut. Tindakan AWS CDK bootstrap mensintesis aplikasi di repositori ini sebelum memulai proses bootstrap. Saat ini, satu-satunya label repositori yang didukung adalah. `WorkflowSource`

Jika AWS CDK aplikasi Anda tidak terkandung dalam repositori sumber, itu harus berada dalam artefak yang dihasilkan oleh tindakan lain.

Untuk informasi selengkapnya tentang sumber, lihat Menghubungkan alur kerja ke repositori sumber.

UI yang sesuai: Tab/Sumber Input - opsional

Artifacts - input

(*CDKBootstrapAction*/Inputs/Artifacts)

(Diperlukan jika AWS CDK aplikasi Anda disimpan dalam artefak keluaran dari tindakan sebelumnya)

Jika AWS CDK aplikasi Anda terkandung dalam artefak yang dihasilkan oleh tindakan sebelumnya, tentukan artefak tersebut di sini. Tindakan AWS CDK bootstrap mensintesis aplikasi dalam artefak yang ditentukan ke dalam CloudFormation template sebelum memulai proses bootstrap. Jika AWS CDK aplikasi Anda tidak terkandung dalam artefak, itu harus berada di repositori sumber Anda.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Input tab/Artefak - opsional

Outputs

(*CDKBootstrapAction*/Outputs)

(Opsional)

Mendefinisikan data yang dihasilkan oleh tindakan selama menjalankan alur kerja.

UI yang sesuai: Tab keluaran

Artifacts - output

(*CDKBootstrapAction*/Outputs/Artifacts)

(Opsional)

Tentukan artefak yang dihasilkan oleh tindakan. Anda dapat mereferensikan artefak ini sebagai masukan dalam tindakan lain.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab keluaran/Artefak

Name

(*CDKBootstrapAction*/Outputs/Artifacts/Name)

(Diperlukan Artifacts - output jika disertakan)

Tentukan nama artefak yang akan berisi AWS CloudFormation template yang disintesis oleh aksi AWS CDK bootstrap saat runtime. Nilai default-nya adalah `cdk_bootstrap_artifacts`. Jika Anda tidak menentukan artefak, maka tindakan mensintesis template, tetapi tidak akan menyimpannya dalam artefak. Pertimbangkan untuk menyimpan template yang disintesis dalam artefak untuk menyimpan catatannya untuk tujuan pengujian atau pemecahan masalah.

UI yang sesuai: Tab Keluaran/Artefak/Tambahkan artefak/Bangun nama artefak

Files

(*CDKBootstrapAction*/Outputs/Artifacts/Files)

(Diperlukan Artifacts - output jika disertakan)

Tentukan file yang akan disertakan dalam artefak. Anda harus menentukan `"cdk.out/**/*"` untuk menyertakan AWS CloudFormation template yang disintesis AWS CDK aplikasi Anda.

> **Note**
>
> `cdk.out`adalah direktori default tempat file yang disintesis disimpan. Jika Anda menentukan direktori keluaran selain `cdk.out` di `cdk.json` file Anda, tentukan direktori itu di sini, bukan`cdk.out`.

UI yang sesuai: Tab Keluaran/Artefak/Tambahkan artefak/File yang dihasilkan oleh build

Environment

(*CDKBootstrapAction*/Environment)

(Diperlukan)

Tentukan CodeCatalyst lingkungan yang akan digunakan dengan tindakan.

Untuk informasi lebih lanjut tentang lingkungan, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst danPembuatan lingkungan.

UI yang sesuai: Tab Konfigurasi/'Lingkungan/Akun/Peran'/Lingkungan

Name

(*CDKBootstrapAction*/Environment/Name)

(Diperlukan Environment jika disertakan)

Tentukan nama lingkungan yang ada yang ingin Anda kaitkan dengan tindakan.

UI yang sesuai: Tab Konfigurasi/'Lingkungan/Akun/Peran'/Lingkungan

Connections

(*CDKBootstrapAction*/Environment/Connections)

(Diperlukan Environment jika disertakan)

Tentukan koneksi akun untuk dikaitkan dengan tindakan. Anda dapat menentukan maksimum satu koneksi akun di bawah`Environment`.

Untuk informasi selengkapnya tentang koneksi akun, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS. Untuk informasi tentang cara mengaitkan koneksi akun dengan lingkungan Anda, lihatPembuatan lingkungan.

UI yang sesuai: Tab konfigurasi/'lingkungan/akun/peran'/koneksi akun AWS

Name

(*CDKBootstrapAction*/Environment/Connections/Name)

(Diperlukan)

Tentukan nama koneksi akun.

UI yang sesuai: Tab konfigurasi/'lingkungan/akun/peran'/koneksi akun AWS

Role

(*CDKBootstrapAction*/Environment/Connections/Role)

(Diperlukan)

Tentukan nama peran IAM yang digunakan tindakan AWS CDK bootstrap untuk mengakses AWS dan menambahkan tumpukan bootstrap. Pastikan bahwa peran ini mencakup kebijakan berikut:

> **Note**
>
> Izin yang ditampilkan dalam kebijakan izin berikut adalah izin yang diperlukan oleh `cdk bootstrap` perintah untuk melakukan bootstrap. Izin ini dapat berubah jika AWS CDK perubahan perintah bootstrap nya.

> **Warning**
>
> Hanya gunakan peran ini dengan tindakan AWS CDK bootstrap. Ini sangat permisif, dan menggunakannya dengan tindakan lain dapat menimbulkan risiko keamanan.

- Kebijakan izin berikut:

  > **Warning**
  >
  > Batasi izin untuk yang ditampilkan dalam kebijakan berikut. Menggunakan peran dengan izin yang lebih luas dapat menimbulkan risiko keamanan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "ssm:GetParameterHistory",
                "ecr:PutImageScanningConfiguration",
                "cloudformation:*",
                "iam:CreateRole",
                "iam:AttachRolePolicy",
                "ssm:GetParameters",
                "iam:PutRolePolicy",
                "ssm:GetParameter",
                "ssm:DeleteParameters",
                "ecr:DeleteRepository",
                "ssm:PutParameter",
                "ssm:DeleteParameter",
                "iam:PassRole",
                "ecr:SetRepositoryPolicy",
                "ssm:GetParametersByPath",
                "ecr:DescribeRepositories",
                "ecr:GetLifecyclePolicy"
            ],
            "Resource": [
                "arn:aws:ssm:aws-region:aws-account:parameter/cdk-bootstrap/*",
                "arn:aws:cloudformation:aws-region:aws-account:stack/CDKToolkit/*",
                "arn:aws:ecr:aws-region:aws-account:repository/cdk-*",
                "arn:aws:iam::aws-account:role/cdk-*"
            ]
        },
        {
            "Sid": "VisualEditor1",
            "Effect": "Allow",
            "Action": [
                "cloudformation:RegisterType",
                "cloudformation:CreateUploadBucket",
                "cloudformation:ListExports",
                "cloudformation:DescribeStackDriftDetectionStatus",
                "cloudformation:SetTypeDefaultVersion",
```

```
                "cloudformation:RegisterPublisher",
                "cloudformation:ActivateType",
                "cloudformation:ListTypes",
                "cloudformation:DeactivateType",
                "cloudformation:SetTypeConfiguration",
                "cloudformation:DeregisterType",
                "cloudformation:ListTypeRegistrations",
                "cloudformation:EstimateTemplateCost",
                "cloudformation:DescribeAccountLimits",
                "cloudformation:BatchDescribeTypeConfigurations",
                "cloudformation:CreateStackSet",
                "cloudformation:ListStacks",
                "cloudformation:DescribeType",
                "cloudformation:ListImports",
                "s3:*",
                "cloudformation:PublishType",
                "ecr:CreateRepository",
                "cloudformation:DescribePublisher",
                "cloudformation:DescribeTypeRegistration",
                "cloudformation:TestType",
                "cloudformation:ValidateTemplate",
                "cloudformation:ListTypeVersions"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan, gunakan wildcard berikut dalam pernyataan kebijakan sumber daya dan kemudian cakupkan kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

- Kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Pastikan peran ini ditambahkan ke koneksi akun Anda. Untuk mempelajari selengkapnya tentang menambahkan peran IAM ke koneksi akun, lihatMenambahkan peran IAM ke koneksi akun.

> **Note**
>
> Anda dapat menentukan nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran di sini, jika Anda mau. Untuk informasi selengkapnya tentang peran ini, silakan lihat Membuat CodeCatalystWorkflowDevelopmentRole-***spaceName***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami menyarankan Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian.

UI yang sesuai: Tab Konfigurasi/'lingkungan/akun/peran'/ Peran

Configuration

(*`CDKBootstrapAction`*/Configuration)

(Diperlukan)

Bagian di mana Anda dapat menentukan properti konfigurasi tindakan.

UI yang sesuai: Tab konfigurasi

Region

(*CDKBootstrapAction*/Configuration/Region)

(Diperlukan)

Tentukan Wilayah AWS ke mana tumpukan bootstrap akan digunakan. Wilayah ini harus cocok dengan yang digunakan AWS CDK aplikasi Anda. Untuk daftar kode Wilayah, lihat Titik akhir Regional.

UI yang sesuai: Tab konfigurasi/Wilayah

CdkCliVersion

(*CDKBootstrapAction*/Configuration/CdkCliVersion)

(Opsional)

Properti ini tersedia dengan versi 1.0.13 atau yang lebih baru dari tindakan AWS CDK penerapan, dan versi 1.0.8 atau yang lebih baru dari tindakan bootstrap.AWS CDK

Tentukan satu dari yang berikut ini:

- Versi lengkap dari AWS Cloud Development Kit (AWS CDK) Command Line Interface (CLI) (juga disebut AWS CDK Toolkit) yang Anda ingin tindakan ini untuk digunakan. Contoh:`2.102.1`. Pertimbangkan untuk menentukan versi lengkap untuk memastikan konsistensi dan stabilitas saat membangun dan menerapkan aplikasi Anda.

  Atau

- `latest`. Pertimbangkan `latest` untuk menentukan untuk memanfaatkan fitur dan perbaikan terbaru CDK CLI.

Tindakan akan mengunduh versi AWS CDK CLI yang ditentukan (atau versi terbaru) ke gambar CodeCatalyst build, dan kemudian menggunakan versi ini untuk menjalankan perintah yang diperlukan untuk menyebarkan aplikasi CDK Anda atau mem-bootstrap lingkungan Anda. AWS

Untuk daftar versi CDK CLI yang didukung yang dapat Anda gunakan,AWS CDK lihat Versi.

Jika Anda menghilangkan properti ini, tindakan menggunakan versi AWS CDK CLI default yang dijelaskan dalam salah satu topik berikut:

- Versi CDK CLI yang digunakan oleh tindakan AWS CDK "menyebarkan”
- Versi CDK CLI yang digunakan oleh tindakan AWS CDK "bootstrap”

UI yang sesuai: Tab konfigurasi/versi AWS CDK CLI

Menerbitkan file ke Amazon S3 dengan alur kerja

Bagian ini menjelaskan cara mempublikasikan file ke Amazon S3 menggunakan alur kerja. CodeCatalyst Untuk mencapai ini, Anda harus menambahkan tindakan publikasi Amazon S3 ke alur kerja Anda. Tindakan Amazon S3 menerbitkan tindakan menyalin file dari direktori sumber ke bucket Amazon S3. Direktori sumber dapat berada di:

- Sebuah repositori sumber, atau
- Artefak keluaran yang dihasilkan oleh aksi alur kerja lain

Kapan menggunakan tindakan “Amazon S3 publish”

Gunakan tindakan ini jika:

- Anda memiliki alur kerja yang menghasilkan file yang ingin Anda simpan di Amazon S3.

  Misalnya, Anda mungkin memiliki alur kerja yang membangun situs web statis yang ingin Anda host di Amazon S3. Dalam hal ini, alur kerja Anda akan menyertakan tindakan build untuk membangun HTML situs dan file pendukung, dan tindakan publikasi Amazon S3 untuk menyalin file ke Amazon S3.
- Anda memiliki repositori sumber yang berisi file yang ingin Anda simpan di Amazon S3.

  Misalnya, Anda mungkin memiliki repositori sumber dengan file sumber aplikasi yang ingin Anda arsipkan setiap malam ke Amazon S3.

Topik

- Contoh alur kerja yang menerbitkan file ke Amazon S3
- Menambahkan tindakan “Amazon S3 publish”
- Tindakan “Amazon S3 menerbitkan” definisi YAMAL tindakan

Contoh alur kerja yang menerbitkan file ke Amazon S3

Alur kerja contoh berikut mencakup tindakan publikasi Amazon S3, bersama dengan tindakan build. Alur kerja membangun situs web dokumentasi statis dan kemudian menerbitkannya ke Amazon S3, di mana ia di-host. Alur kerja terdiri dari blok bangunan berikut yang berjalan secara berurutan:

> **Note**
>
> Contoh alur kerja berikut adalah untuk tujuan ilustrasi, dan hanya akan bekerja dengan konfigurasi tambahan.

- Pemicu - Pemicu ini memulai alur kerja yang dijalankan secara otomatis saat Anda mendorong perubahan ke repositori sumber Anda. Untuk informasi lebih lanjut tentang menggunakan pemicu, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.
- Tindakan build (`BuildDocs`) — Pada pemicu, tindakan membangun situs web dokumentasi statis (`mkdocs build`) dan menambahkan file HTML terkait dan metadata pendukung ke artefak yang disebut. `MyDocsSite` Untuk informasi selengkapnya tentang tindakan build, lihatMembangun dengan alur kerja.
- Tindakan publikasi Amazon S3 (`PublishToS3`) - Setelah menyelesaikan tindakan pembuatan, tindakan ini menyalin situs dalam `MyDocsSite` artefak ke Amazon S3 untuk hosting.

```
Name: codecatalyst-s3-publish-workflow
SchemaVersion: 1.0

Triggers:
  - Type: PUSH
    Branches:
      - main
Actions:
  BuildDocs:
    Identifier: aws/build@v1
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      Steps:
        - Run: echo BuildDocs started on `date`
        - Run: pip install --upgrade pip
```

```
      - Run: pip install mkdocs
      - Run: mkdocs build
      - Run: echo BuildDocs completed on `date`
  Outputs:
    Artifacts:
    - Name: MyDocsSite
      Files:
        - "site/**/*"

PublishToS3:
  Identifier: aws/s3-publish@v1
  Environment:
    Name: codecatalyst-s3-publish-environment
    Connections:
      - Name: codecatalyst-account-connection
        Role: codecatalyst-s3-publish-build-role
  Inputs:
    Sources:
      - WorkflowSource
    Artifacts:
      - MyDocsSite
  Configuration:
    DestinationBucketName: my-bucket
    SourcePath: /artifacts/PublishToS3/MyDocSite/site
    TargetPath: my/docs/site
```

Menambahkan tindakan “Amazon S3 publish”

Gunakan petunjuk berikut untuk menambahkan tindakan publikasi Amazon S3 ke alur kerja Anda.

Visual

Untuk menambahkan tindakan “Amazon S3 publish” menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.

7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan publikasi Amazon S3, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih Amazon S3 terbitkan. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Lihat sumber untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Pada tab Input, Konfigurasi, dan Output, lengkapi bidang sesuai dengan kebutuhan Anda. Untuk deskripsi setiap bidang, lihatTindakan “Amazon S3 menerbitkan” definisi YAMAL tindakan. Referensi ini memberikan informasi rinci tentang setiap bidang (dan nilai properti YAMAL yang sesuai) seperti yang muncul di YAML dan editor visual.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, lalu pilih Komit lagi.

YAML

Untuk menambahkan tindakan “Amazon S3 publish” menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan publikasi Amazon S3, dan lakukan salah satu hal berikut:

- Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

  Atau

- Pilih Amazon S3 terbitkan. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
  - (Opsional) Pilih Lihat sumber untuk melihat kode sumber tindakan.
  - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

10. Ubah properti dalam kode YAMAL sesuai dengan kebutuhan Anda. Penjelasan tentang setiap properti yang tersedia disediakan diTindakan "Amazon S3 menerbitkan" definisi YAMAL tindakan.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, lalu pilih Komit lagi.

Tindakan "Amazon S3 menerbitkan" definisi YAMAL tindakan

Berikut ini adalah definisi YAMAL dari tindakan publikasi Amazon S3. Untuk mempelajari cara menggunakan tindakan ini, lihatMenerbitkan file ke Amazon S3 dengan alur kerja.

Definisi tindakan ini ada sebagai bagian dalam file definisi alur kerja yang lebih luas. Untuk informasi selengkapnya tentang file ini, lihatAlur kerja definisi YAMAL.

> **Note**
>
> Sebagian besar properti YAMAL yang mengikuti memiliki elemen UI yang sesuai di editor visual. Untuk mencari elemen UI, gunakan Ctrl+F. Elemen akan terdaftar dengan properti YAMLnya yang terkait.

```
# The workflow definition starts here.
# See Properti tingkat atas for details.

Name: MyWorkflow
SchemaVersion: 1.0
Actions:

# The action definition starts here.
```

```
  S3Publish_nn:
    Identifier: aws/s3-publish@v1
    DependsOn:
      - build-action
    Compute:
      Type: EC2 | Lambda
      Fleet: fleet-name
    Timeout: timeout-minutes
    Inputs:
      Sources:
        - source-name-1
      Artifacts:
        - artifact-name
      Variables:
        - Name: variable-name-1
          Value: variable-value-1
        - Name: variable-name-2
          Value: variable-value-2
    Environment:
      Name: environment-name
      Connections:
        - Name: account-connection-name
          Role: iam-role-name
    Configuration:
      SourcePath: my/source
      DestinationBucketName: s3-bucket-name
      TargetPath: my/target
```

S3Publish

(Diperlukan)

Tentukan nama tindakan. Semua nama tindakan harus unik dalam alur kerja. Nama aksi terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama tindakan.

Default: `S3Publish_nn`.

UI yang sesuai: Tab konfigurasi/Nama tindakan

Identifier

(*S3Publish*/Identifier)

(Diperlukan)

Mengidentifikasi tindakan. Jangan mengubah properti ini kecuali Anda ingin mengubah versi. Untuk informasi selengkapnya, lihat Menentukan versi mayor, minor, atau patch dari suatu tindakan.

Default: `aws/s3-publish@v1`.

UI yang sesuai: Diagram alur kerja/ S3Publish _nn/ aws/s3-publish @v1 label

DependsOn

(*S3Publish*/DependsOn)

(Opsional)

Tentukan tindakan, grup tindakan, atau gerbang yang harus berjalan dengan sukses agar tindakan ini berjalan.

Untuk informasi selengkapnya tentang fungsionalitas 'tergantung pada', lihat. Mengkonfigurasi tindakan untuk bergantung pada tindakan lain

UI yang sesuai: Tab masukan/Tergantung pada - opsional

Compute

(*S3Publish*/Compute)

(Opsional)

Mesin komputasi yang digunakan untuk menjalankan tindakan alur kerja Anda. Anda dapat menentukan komputasi baik di tingkat alur kerja atau di tingkat tindakan, tetapi tidak keduanya. Ketika ditentukan pada tingkat alur kerja, konfigurasi komputasi berlaku untuk semua tindakan yang ditentukan dalam alur kerja. Pada tingkat alur kerja, Anda juga dapat menjalankan beberapa tindakan pada instance yang sama. Untuk informasi selengkapnya, lihat Berbagi komputasi di seluruh tindakan.

UI yang sesuai: tidak ada

Type

(*S3Publish*/Compute/Type)

(Diperlukan Compute jika disertakan)

Jenis mesin komputasi. Anda dapat menggunakan salah satu nilai berikut:

- EC2 (editor visual) atau `EC2` (editor YAMG)

  Dioptimalkan untuk fleksibilitas selama aksi berjalan.
- Lambda (editor visual) atau `Lambda` (editor YAMG)

  Kecepatan start-up aksi yang dioptimalkan.

Untuk informasi selengkapnya tentang jenis komputasi, lihatJenis komputasi.

UI yang sesuai: Tab konfigurasi/Jenis komputasi

Fleet

(*`S3Publish`*/Compute/Fleet)

(Opsional)

Tentukan mesin atau armada yang akan menjalankan alur kerja atau tindakan alur kerja Anda. Dengan armada sesuai permintaan, ketika suatu tindakan dimulai, alur kerja menyediakan sumber daya yang dibutuhkan, dan mesin dihancurkan ketika tindakan selesai. Contoh armada sesuai permintaan:`Linux.x86-64.Large`,. `Linux.x86-64.XLarge` Untuk informasi lebih lanjut tentang armada sesuai permintaan, lihat. Properti armada sesuai permintaan

Dengan armada yang disediakan, Anda mengonfigurasi satu set mesin khusus untuk menjalankan tindakan alur kerja Anda. Mesin-mesin ini tetap menganggur, siap untuk memproses tindakan segera. Untuk informasi lebih lanjut tentang armada yang disediakan, lihat. Properti armada yang disediakan

Jika `Fleet` dihilangkan, defaultnya adalah. `Linux.x86-64.Large`

UI yang sesuai: Tab konfigurasi/Armada komputasi

Timeout

(*`S3Publish`*/Timeout)

(Diperlukan)

Tentukan jumlah waktu dalam menit (editor YAMAL), atau jam dan menit (editor visual), bahwa tindakan dapat berjalan sebelum CodeCatalyst mengakhiri tindakan. Minimal adalah 5 menit dan

maksimum dijelaskan dalamKuota untuk alur kerja. Batas waktu default sama dengan batas waktu maksimum.

UI yang sesuai: Tab konfigurasi/Timeout - opsional

Inputs

(*S3Publish*/Inputs)

(Opsional)

`Inputs`Bagian ini mendefinisikan data yang `S3Publish` dibutuhkan selama menjalankan alur kerja.

> **Note**
>
> Maksimal empat input (satu sumber dan tiga artefak) diizinkan untuk setiap tindakan AWS CDK penerapan. Variabel tidak dihitung terhadap total ini.

Jika Anda perlu merujuk ke file yang berada di input yang berbeda (katakanlah sumber dan artefak), input sumber adalah input utama, dan artefak adalah input sekunder. Referensi ke file dalam input sekunder mengambil awalan khusus untuk menyisihkannya dari primer. Lihat perinciannya di Contoh: Merujuk file dalam beberapa artefak.

UI yang sesuai: Tab input

Sources

(*S3Publish*/Inputs/Sources)

(Diperlukan jika file yang ingin Anda publikasikan ke Amazon S3 disimpan dalam repositori sumber)

Jika file yang ingin Anda publikasikan ke Amazon S3 disimpan dalam repositori sumber, tentukan label repositori sumber tersebut. Saat ini, satu-satunya label yang didukung adalah`WorkflowSource`.

Jika file yang ingin Anda publikasikan ke Amazon S3 tidak terkandung dalam repositori sumber, mereka harus berada dalam artefak yang dihasilkan oleh tindakan lain.

Untuk informasi selengkapnya tentang sumber, lihat Menghubungkan alur kerja ke repositori sumber.

UI yang sesuai: Tab/Sumber Input - opsional

Artifacts - input

(*S3Publish*/Inputs/Artifacts)

(Diperlukan jika file yang ingin Anda publikasikan ke Amazon S3 disimpan dalam artefak keluaran dari tindakan sebelumnya)

Jika file yang ingin Anda publikasikan ke Amazon S3 terkandung dalam artefak yang dihasilkan oleh tindakan sebelumnya, tentukan artefak tersebut di sini. Jika file Anda tidak terkandung dalam artefak, mereka harus berada di repositori sumber Anda.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab konfigurasi/Artefak - opsional

Variables - input

(*S3Publish*/Inputs/Variables)

(Opsional)

Tentukan urutan pasangan nama/nilai yang menentukan variabel input yang ingin Anda sediakan untuk tindakan. Nama variabel terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama variabel.

Untuk informasi selengkapnya tentang variabel, termasuk contoh, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

UI yang sesuai: Input tab/Variabel - opsional

Environment

(*S3Publish*/Environment)

(Diperlukan)

Tentukan CodeCatalyst lingkungan yang akan digunakan dengan tindakan.

Untuk informasi lebih lanjut tentang lingkungan, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst danPembuatan lingkungan.

UI yang sesuai: Tab konfigurasi/'lingkungan/koneksi/peran'/Lingkungan

Name

(*S3Publish*/Environment/Name)

(Diperlukan Environment jika disertakan)

Tentukan nama lingkungan yang ada yang ingin Anda kaitkan dengan tindakan.

UI yang sesuai: Tab konfigurasi/'lingkungan/koneksi/peran'/Lingkungan

Connections

(*S3Publish*/Environment/Connections)

(Diperlukan Environment jika disertakan)

Tentukan koneksi akun untuk dikaitkan dengan tindakan. Anda dapat menentukan maksimum satu koneksi akun di bawah`Environment`.

Untuk informasi selengkapnya tentang koneksi akun, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS. Untuk informasi tentang cara mengaitkan koneksi akun dengan lingkungan Anda, lihatPembuatan lingkungan.

UI yang sesuai: Tab Konfigurasi/'lingkungan/koneksi/peran'/ Koneksi

Name

(*S3Publish*/Environment/Connections/Name)

(Diperlukan)

Tentukan nama koneksi akun.

UI yang sesuai: Tab Konfigurasi/'lingkungan/koneksi/peran'/ Koneksi

Role

(*S3Publish*/Environment/Connections/Role)

(Diperlukan)

Tentukan nama peran IAM yang digunakan tindakan publikasi Amazon S3 untuk AWS mengakses dan menyalin file ke Amazon S3. Pastikan bahwa peran ini meliputi:

- Kebijakan izin berikut:

> **Warning**
>
> Batasi izin untuk yang ditampilkan dalam kebijakan berikut. Menggunakan peran dengan izin yang lebih luas dapat menimbulkan risiko keamanan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:ListBucket",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::bucket-name",
                "arn:aws:s3:::bucket-name/*"
            ]
        }
    ]
}
```

- Kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
```

```
                    "codecatalyst-runner.amazonaws.com",
                    "codecatalyst.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Pastikan peran ini terkait dengan koneksi akun Anda. Untuk mempelajari lebih lanjut tentang mengaitkan peran IAM dengan koneksi akun, lihat. Menambahkan peran IAM ke koneksi akun

> **Note**
>
> Anda dapat menentukan nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran di sini, jika Anda mau. Untuk informasi selengkapnya tentang peran ini, silakan lihat Membuat CodeCatalystWorkflowDevelopmentRole-***`spaceName`***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami menyarankan Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian.

UI yang sesuai: Tab Konfigurasi/'lingkungan/koneksi/peran'/Peran

Configuration

(*`S3Publish`*/Configuration)

(Diperlukan)

Bagian di mana Anda dapat menentukan properti konfigurasi tindakan.

UI yang sesuai: Tab konfigurasi

SourcePath

(*`S3Publish`*/Configuration/SourcePath)

(Diperlukan)

Tentukan nama dan jalur direktori atau file yang ingin Anda publikasikan ke Amazon S3. Direktori atau file dapat berada di repositori sumber atau artefak dari tindakan sebelumnya, dan relatif terhadap repositori sumber atau root artefak.

Contoh:

Menentukan `./myFolder/` salinan konten `/myFolder` ke Amazon S3, dan mempertahankan struktur direktori yang mendasarinya.

Menentukan `./myFolder/myfile.txt` salinan hanya `myfile.txt` untuk Amazon S3. (Struktur direktori dihapus.)

Anda tidak dapat menggunakan wildcard.

> **Note**
>
> Anda mungkin perlu menambahkan awalan ke direktori atau jalur file untuk menunjukkan artefak atau sumber mana yang akan menemukannya. Untuk informasi selengkapnya, lihat Mereferensikan file dalam repositori sumber dan Merujuk file dalam artefak.

UI yang sesuai: Tab konfigurasi/Jalur sumber

DestinationBucketName

(*S3Publish*/Configuration/DestinationBucketName)

(Diperlukan)

Tentukan nama bucket Amazon S3 tempat Anda ingin mempublikasikan file.

UI yang sesuai: Tab konfigurasi/ember Tujuan - opsional

TargetPath

(*S3Publish*/Configuration/TargetPath)

(Opsional)

Tentukan nama dan jalur direktori di Amazon S3 tempat Anda ingin mempublikasikan file Anda. Jika direktori tidak ada, itu akan dibuat. Jalur direktori tidak boleh menyertakan nama bucket.

Contoh:

`myS3Folder`

`./myS3Folder/myS3Subfolder`

UI yang sesuai: Tab konfigurasi/direktori Tujuan - opsional

Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst

Anda dapat meminta tindakan CodeCatalyst alur kerja untuk menyebarkan aplikasi dan sumber daya lainnya ke dalam VPC Anda atau Akun AWS Amazon. Untuk mencapai ini, Anda harus mengatur CodeCatalyst lingkungan.

Lingkungan, jangan dikelirukan dengan Lingkungan Pengembang, adalah tempat kode digunakan. Biasanya berisi instance aplikasi yang sedang berjalan bersama dengan infrastruktur yang terkait. Anda dapat memberi nama lingkungan Anda seperti pengembangan, pengujian, pementasan, atau produksi. Setiap penerapan yang dihasilkan oleh CodeCatalyst ke lingkungan akan muncul di halaman Lingkungan. Untuk mengatur lingkungan, Anda memberinya nama, sepertimy-`production-environment`, dan kemudian mengaitkannya dengan Anda Akun AWS.

Selain menampilkan informasi penyebaran, lingkungan juga berfungsi sebagai mekanisme untuk menetapkan peran AWS IAM untuk tindakan alur kerja.

Dapatkah beberapa lingkungan ada dalam satu alur kerja?

Ya. Jika alur kerja menyertakan beberapa tindakan, setiap tindakan tersebut dapat ditetapkan suatu lingkungan. Misalnya, Anda dapat memiliki alur kerja yang menyertakan dua tindakan penerapan, di mana satu ditetapkan `my-staging-enviroment` lingkungan dan yang lain ditetapkan lingkungan. `my-production-environment`

Tindakan apa yang mendukung lingkungan?

Tindakan berikut mendukung agar informasi penerapannya ditampilkan di halaman Lingkungan:

- Menyebarkan AWS CloudFormation tumpukan - Untuk informasi selengkapnya, lihat Menerapkan AWS CloudFormation tumpukan dengan alur kerja
- Menyebarkan ke Amazon ECS - Untuk informasi selengkapnya, lihat Menyebarkan aplikasi ke Amazon Elastic Container Service (ECS) dengan alur kerja
- Terapkan ke klaster Kubernetes — Untuk informasi selengkapnya, lihat Menerapkan aplikasi ke Amazon Elastic Kubernetes Service dengan alur kerja

- AWS CDK deploy — Untuk informasi selengkapnya, lihat Menerapkan AWS Cloud Development Kit (AWS CDK) aplikasi dengan alur kerja

> **Note**
>
> Jika Anda ingin mengizinkan tindakan untuk mengakses dan melakukan operasi di AWS akun Anda, Anda harus mengaitkannya dengan lingkungan. Banyak tindakan mendukung asosiasi lingkungan, termasuk, namun tidak terbatas pada, tindakan yang tercantum sebelumnya. Anda dapat mengetahui tindakan mana yang mendukung asosiasi lingkungan karena tindakan tersebut akan menyertakan daftar drop-down Lingkungan pada tab Konfigurasi mereka di editor visual.

Wilayah yang Didukung

Halaman Lingkungan dapat menampilkan sumber daya di AWS Wilayah mana pun.

Apakah lingkungan wajib?

Lingkungan adalah wajib jika tindakan alur kerja yang ditugaskan menyebarkan sumber daya ke AWS cloud, atau berkomunikasi dengan AWS layanan karena alasan lain (seperti pemantauan dan pelaporan).

Topik

- Pembuatan lingkungan
- Mengaitkan lingkungan, koneksi akun, dan peran IAM dengan tindakan alur kerja
- Mengaitkan koneksi VPC dengan lingkungan
- Mengasosiasikan Akun AWS dengan lingkungan

Pembuatan lingkungan

Gunakan petunjuk berikut untuk membuat lingkungan kosong yang nantinya dapat Anda kaitkan dengan tindakan alur kerja.

Sebelum Anda mulai

Anda membutuhkan yang berikut ini:

- Sebuah CodeCatalyst ruang. Untuk informasi selengkapnya, lihat Siapkan dan masuk ke CodeCatalyst.
- Sebuah CodeCatalyst proyek. Untuk informasi selengkapnya, lihat Membuat proyek dengan cetak biru.
- Sambungan AWS akun yang menyertakan peran IAM yang perlu diakses oleh tindakan alur kerja Anda. AWS Anda dapat menggunakan maksimal satu koneksi akun per lingkungan. Untuk informasi selengkapnya, lihat Memungkinkan akses ke AWS sumber daya yang terhubung Akun AWS.

> **Note**
>
> Anda dapat membuat lingkungan tanpa koneksi akun; Namun, Anda harus kembali dan menambahkan koneksi nanti.

Untuk membuat lingkungan

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Lingkungan.
4. Dalam nama Lingkungan, masukkan nama, seperti **`Production`** atau**`Staging`**.
5. Dalam jenis Lingkungan, pilih salah satu dari berikut ini:

   - Non-produksi — Lingkungan di mana Anda dapat menguji aplikasi Anda untuk memastikannya berfungsi sebagaimana dimaksud sebelum memindahkannya ke produksi.
   - Produksi - Lingkungan 'langsung' yang tersedia untuk umum dan menjadi tuan rumah aplikasi akhir Anda.

     Jika Anda memilih Produksi, lencana Produksi akan muncul di UI di samping tindakan apa pun yang terkait dengan lingkungan. Lencana membantu Anda dengan cepat melihat tindakan mana yang diterapkan ke produksi. Selain penampilan lencana, tidak ada perbedaan antara lingkungan produksi dan non-produksi.
6. (Opsional) Dalam koneksi VPC, pilih koneksi VPC yang ingin Anda kaitkan dengan lingkungan ini. Untuk informasi selengkapnya tentang membuat koneksi VPC ini, lihat Mengelola Awan Pribadi Virtual Amazon di Panduan CodeCatalyst Administrator.

7. (Opsional) Dalam Deskripsi, masukkan deskripsi seperti**`Production environment for the hello-world app`**.
8. Dalam Koneksi - opsional, pilih koneksi AWS akun yang ingin Anda kaitkan dengan lingkungan ini. Pastikan koneksi akun menyertakan peran IAM yang ingin Anda kaitkan dengan lingkungan. Untuk informasi selengkapnya tentang membuat koneksi ini, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS.
9. Pilih Buat lingkungan. CodeCatalyst menciptakan lingkungan yang kosong.

Langkah selanjutnya

- Sekarang Anda telah menciptakan lingkungan, Anda siap untuk mengaitkannya dengan tindakan alur kerja. Untuk informasi selengkapnya, lihat Mengaitkan lingkungan, koneksi akun, dan peran IAM dengan tindakan alur kerja.

Mengaitkan lingkungan, koneksi akun, dan peran IAM dengan tindakan alur kerja

Saat Anda mengaitkan lingkungan, koneksi akun, dan peran IAM dengan tindakan alur kerja yang didukung, peran IAM akan tersedia untuk digunakan oleh tindakan tersebut. Selain mendapatkan akses ke peran IAM, tindakan tersebut mungkin juga memiliki informasi penyebaran yang diimpor ke halaman Lingkungan. Untuk informasi selengkapnya, lihat Tindakan apa yang mendukung lingkungan?.

Gunakan petunjuk berikut untuk mengaitkan lingkungan, koneksi akun, dan peran IAM dengan tindakan.

Langkah 1: Kaitkan lingkungan, koneksi akun, dan peran ke tindakan alur kerja

Gunakan prosedur berikut untuk mengaitkan lingkungan, koneksi akun, dan peran dengan tindakan alur kerja.

Visual

Untuk mengaitkan lingkungan, koneksi akun, dan peran dengan tindakan alur kerja menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.

4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Dalam diagram alur kerja, pilih tindakan yang didukung dengan lingkungan. Untuk informasi selengkapnya, lihat Tindakan apa yang mendukung lingkungan?.
8. Pilih tab Konfigurasi, dan tentukan informasi ke dalam bidang, sebagai berikut.

   Lingkungan

   Tentukan CodeCatalyst lingkungan yang akan digunakan dengan tindakan.

   Untuk informasi lebih lanjut tentang lingkungan, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst danPembuatan lingkungan.

   Koneksi akun atau Koneksi - opsional (mana saja yang tersedia)

   Tentukan koneksi akun untuk dikaitkan dengan tindakan. Anda dapat menentukan maksimum satu koneksi akun di bawah`Environment`.

   Untuk informasi selengkapnya tentang koneksi akun, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS. Untuk informasi tentang cara mengaitkan koneksi akun dengan lingkungan Anda, lihatPembuatan lingkungan.

   Peran

   Tentukan nama peran IAM yang digunakan tindakan ini untuk mengakses dan beroperasi di AWS layanan seperti Amazon S3 dan Amazon ECR. Pastikan peran ini ditambahkan ke koneksi akun Anda. Untuk menambahkan peran IAM ke koneksi akun, lihatMenambahkan peran IAM ke koneksi akun.

   > **Note**
   >
   > Anda mungkin dapat menentukan nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran di sini, asalkan memiliki izin yang cukup. Untuk informasi selengkapnya tentang peran ini, silakan lihat Membuat CodeCatalystWorkflowDevelopmentRole-***`spaceName`***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`*

peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami menyarankan Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian.

Jika Anda tidak melihat peran dalam daftar, itu karena Anda belum mengaitkannya ke koneksi akun. Untuk informasi selengkapnya, lihat Menambahkan peran IAM ke koneksi akun.

9. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
10. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk mengaitkan lingkungan, koneksi akun, dan peran dengan tindakan alur kerja menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Dalam tindakan alur kerja yang ingin Anda kaitkan dengan lingkungan, tambahkan kode yang mirip dengan berikut ini:

```
action-name
  Environment:
    Name: environment-name
    Connections:
      - Name: account-connection-name
        Role: iam-role-name
```

Untuk informasi selengkapnya, lihat Alur kerja definisi YAMAL untuk tindakan Anda.

8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Langkah 2: Mengisi lingkungan

Setelah mengaitkan lingkungan, koneksi akun, dan peran ke tindakan alur kerja, Anda dapat mengisi halaman Lingkungan dengan informasi penerapan. Gunakan petunjuk berikut untuk mengisi halaman Lingkungan.

> **Note**
>
> Halaman Lingkungan didukung oleh hanya sebagian dari tindakan alur kerja. Untuk informasi selengkapnya, lihat Tindakan apa yang mendukung lingkungan?.

Untuk mengisi lingkungan

1. Jika alur kerja tidak dimulai secara otomatis saat Anda melakukan perubahanLangkah 1: Kaitkan lingkungan, koneksi akun, dan peran ke tindakan alur kerja, mulailah menjalankan secara manual sebagai berikut:

   a. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.

   b. Pilih nama alur kerja tempat Anda ingin memulai proses. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.

   c. Pilih Jalankan.

   Proses alur kerja memulai penerapan baru, yang menyebabkan CodeCatalyst untuk menambahkan informasi sumber daya aplikasi Anda di bawah Lingkungan.

2. Verifikasi bahwa sumber daya aplikasi Anda muncul di lingkungan Anda:

   a. Di panel navigasi, pilih CI/CD, lalu pilih Lingkungan.

   b. Pilih lingkungan Anda (misalnya,`Production`).

   c. Pilih tab aktivitas Deployment, dan verifikasi bahwa penerapan muncul dengan Status SUCCEEDED. Ini menunjukkan bahwa alur kerja berjalan berhasil menyebarkan sumber daya aplikasi Anda.

   d. Pilih tab Target Deployment, dan verifikasi apakah resource aplikasi Anda muncul.

Mengaitkan koneksi VPC dengan lingkungan

Ketika suatu tindakan dikonfigurasi dengan lingkungan yang memiliki koneksi VPC, tindakan akan berjalan terhubung ke VPC, mengikuti aturan jaringan dan mengakses sumber daya yang ditentukan oleh VPC terkait. Koneksi VPC yang sama dapat digunakan oleh satu atau lebih lingkungan.

Gunakan petunjuk berikut untuk mengaitkan koneksi VPC dengan lingkungan.

Untuk mengaitkan koneksi VPC dengan lingkungan

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Lingkungan.
4. Pilih lingkungan Anda (misalnya,`Production`).
5. Pilih tab Properti Lingkungan.
6. Pilih Kelola koneksi VPC, pilih koneksi VPC yang Anda inginkan, dan pilih Konfirmasi. Ini mengaitkan koneksi VPC pilihan Anda dengan lingkungan ini.

Untuk informasi selengkapnya, lihat Mengelola Awan Pribadi Virtual Amazon di Panduan CodeCatalyst Administrator.

Mengasosiasikan Akun AWS dengan lingkungan

Gunakan instruksi berikut untuk mengaitkan Akun AWS dengan lingkungan.

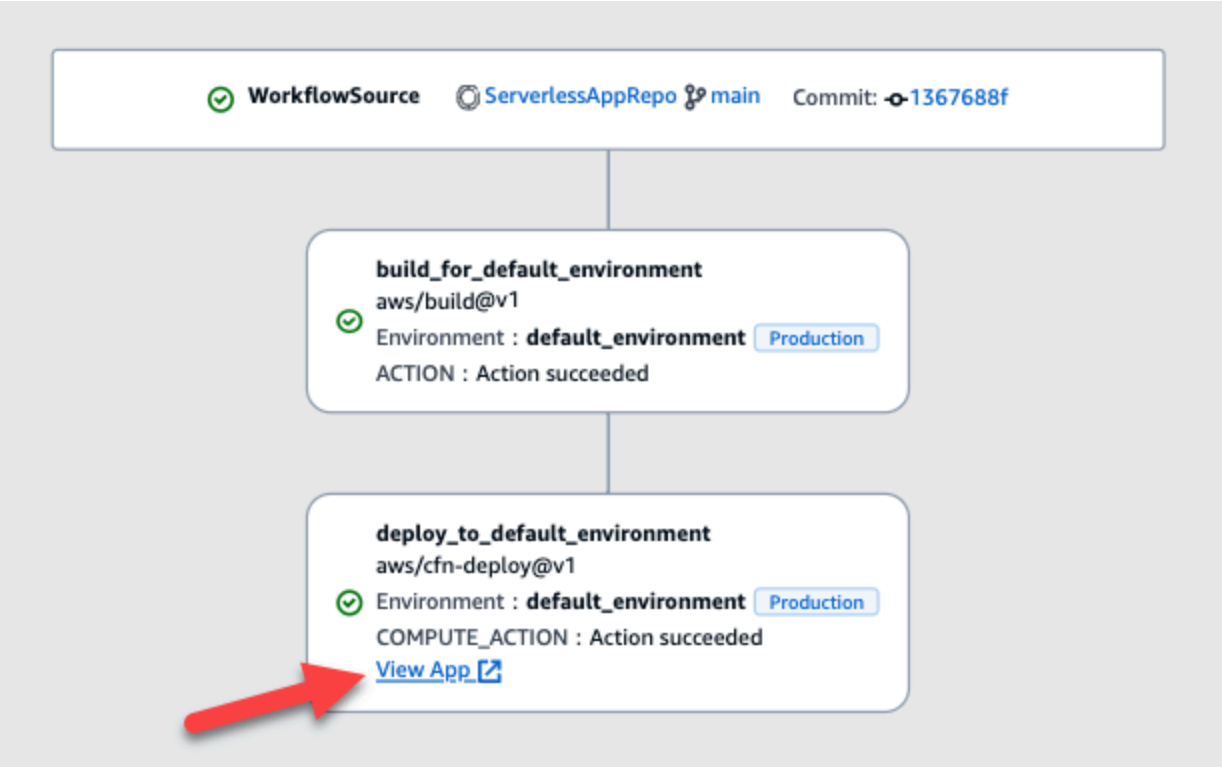
Untuk mengasosiasikan Akun AWS dengan lingkungan

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Lingkungan.
4. Pilih lingkungan Anda (misalnya,`Production`).
5. Pilih tab Properti Lingkungan.
6. Pilih Associate Akun AWS, pilih yang Anda inginkan Akun AWS, dan pilih Associate. Ini mengaitkan pilihan Anda Akun AWS dengan lingkungan ini.

Untuk informasi selengkapnya, lihat Memungkinkan akses ke AWS sumber daya yang terhubung Akun AWS.

Menampilkan URL aplikasi yang digunakan dalam diagram alur kerja

Jika alur kerja Anda menyebarkan aplikasi, Anda dapat mengonfigurasi Amazon CodeCatalyst untuk menampilkan URL aplikasi sebagai tautan yang dapat diklik. Tautan ini muncul di CodeCatalyst konsol, di dalam tindakan yang menerapkannya. Diagram alur kerja berikut menunjukkan URL Lihat Aplikasi yang muncul di bagian bawah tindakan.



Dengan membuat URL ini dapat diklik di CodeCatalyst konsol, Anda dapat dengan cepat memverifikasi penerapan aplikasi Anda.

> **Note**
>
> URL aplikasi tidak didukung dengan tindakan Deploy to Amazon ECS.

Untuk mengaktifkan fitur ini, tambahkan variabel output ke tindakan Anda dengan nama yang berisi`appurl`, atau`endpointurl`. Anda dapat menggunakan nama dengan atau tanpa tanda hubung yang bergabung (`-`), garis bawah (_), atau spasi ( ). String tidak peka huruf besar/kecil. Tetapkan nilai variabel ke `http` atau `https` URL aplikasi yang Anda gunakan.

> **Note**
>
> Jika Anda memperbarui variabel keluaran yang ada untuk menyertakan`app url`, atau `endpoint url` string, perbarui semua referensi ke variabel ini untuk menggunakan nama variabel baru.

Untuk langkah-langkah rinci, lihat salah satu prosedur berikut:

- Untuk menampilkan URL aplikasi dalam tindakan "AWS CDK deploy”
- Untuk menampilkan URL aplikasi dalam tindakan “Deploy AWS CloudFormation stack”
- Untuk menampilkan URL aplikasi di semua tindakan lainnya

Setelah selesai mengonfigurasi URL, verifikasi bahwa URL tersebut muncul seperti yang diharapkan dengan mengikuti petunjuk berikut:

- Untuk memverifikasi bahwa URL aplikasi telah ditambahkan

Untuk menampilkan URL aplikasi dalam tindakan "AWS CDK deploy”

1. Jika Anda menggunakan tindakan AWS CDK penerapan, tambahkan `CfnOutput` konstruksi (yang merupakan pasangan nilai kunci) dalam kode aplikasi Anda: AWS CDK

   - Nama kunci harus berisi`appurl`, atau`endpointurl`, dengan atau tanpa tanda hubung yang bergabung (`-`), garis bawah (`_`), atau spasi ( ). String tidak peka huruf besar/kecil.
   - Nilai harus berupa `http` atau `https` URL aplikasi yang Anda gunakan.

   Misalnya, AWS CDK kode Anda mungkin terlihat seperti ini:

```
import { Duration, Stack, StackProps, CfnOutput, RemovalPolicy} from 'aws-cdk-lib';
import * as dynamodb from 'aws-cdk-lib/aws-dynamodb';
import * as s3 from 'aws-cdk-lib/aws-s3';
import { Construct } from 'constructs';
import * as cdk from 'aws-cdk-lib';
export class HelloCdkStack extends Stack {
  constructor(scope: Construct, id: string, props?: StackProps) {
    super(scope, id, props);
    const bucket = new s3.Bucket(this, 'my-bucket', {
```

```
      removalPolicy: RemovalPolicy.DESTROY,
    });
    new CfnOutput(this, 'APP-URL', {
      value: https://mycompany.myapp.com,
      description: 'The URL of the deployed application',
      exportName: 'myApp',
    });
    ...
  }
}
```

Untuk informasi selengkapnya tentang `CfnOutput` konstruksi, lihat antarmuka CfnOutputProps di Referensi AWS Cloud Development Kit (AWS CDK) API.

2. Simpan dan komit kode Anda.
3. Lanjut ke Untuk memverifikasi bahwa URL aplikasi telah ditambahkan.

Untuk menampilkan URL aplikasi dalam tindakan “Deploy AWS CloudFormation stack”

1. Jika Anda menggunakan aksi AWS CloudFormation tumpukan Deploy, tambahkan output ke `Outputs` bagian di CloudFormation template atau AWS SAM template Anda dengan karakteristik berikut:

   - Kunci (juga disebut ID logis) harus berisi`appurl`, atau`endpointurl`, dengan atau tanpa tanda hubung yang bergabung (`-`), garis bawah (`_`), atau spasi ( ). String tidak peka huruf besar/kecil.
   - Nilai harus berupa `http` atau `https` URL aplikasi yang Anda gunakan.

   Misalnya, CloudFormation template Anda mungkin terlihat seperti ini:

```
"Outputs" : {
  "APP-URL" : {
    "Description" : "The URL of the deployed app",
    "Value" : "https://mycompany.myapp.com",
    "Export" : {
      "Name" : "My App"
    }
  }
}
```

Untuk informasi selengkapnya tentang CloudFormation output, lihat Output di AWS CloudFormation Panduan Pengguna.

2. Simpan dan komit kode Anda.
3. Lanjut ke Untuk memverifikasi bahwa URL aplikasi telah ditambahkan.

Untuk menampilkan URL aplikasi di semua tindakan lainnya

Jika Anda menggunakan tindakan lain untuk menerapkan aplikasi Anda, seperti tindakan build atau GitHub Tindakan, lakukan hal berikut agar URL aplikasi ditampilkan.

1. Tentukan variabel lingkungan di `Steps` bagian `Inputs` atau tindakan dalam file definisi alur kerja. Variabel harus memiliki karakteristik ini:

   - `name`Harus berisi`appurl`, atau`endpointurl`, dengan atau tanpa tanda hubung (`-`), garis bawah (_), atau spasi ( ). String tidak peka huruf besar/kecil.
   - Nilai harus berupa `http` atau `https` URL aplikasi yang Anda gunakan.

   Misalnya, tindakan build mungkin terlihat seperti ini:

```
Build-action:
  Identifier: aws/build@v1
  Inputs:
    Variables:
      - Name: APP-URL
        Value: https://mycompany.myapp.com
```

   ... atau ini:

```
Actions:
  Build:
    Identifier: aws/build@v1
    Configuration:
      Steps:
        - Run: APP-URL=https://mycompany.myapp.com
```

   Untuk informasi selengkapnya tentang mendefinisikan variabel lingkungan, lihatMendefinisikan variabel.

2. Ekspor variabel.

   Misalnya, tindakan build Anda mungkin terlihat seperti ini:

   ```
   Build-action:
     ...
     Outputs:
       Variables:
         - APP-URL
   ```

   Untuk informasi tentang mengekspor variabel, lihatMengekspor variabel sehingga tindakan lain dapat menggunakannya.

3. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
4. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.
5. Lanjut ke Untuk memverifikasi bahwa URL aplikasi telah ditambahkan.

Untuk memverifikasi bahwa URL aplikasi telah ditambahkan

- Mulai menjalankan alur kerja, jika belum dimulai secara otomatis. Proses baru harus menampilkan URL aplikasi sebagai tautan yang dapat diklik dalam diagram alur kerjanya. Untuk informasi selengkapnya tentang memulai proses, lihatMemulai alur kerja berjalan secara manual.

Menghapus target penerapan

Anda dapat menghapus target penerapan seperti kluster Amazon ECS atau AWS CloudFormation tumpukan dari halaman Lingkungan di CodeCatalyst konsol.

> **Important**
>
> Saat Anda menghapus target penerapan, target tersebut dihapus dari CodeCatalyst konsol, tetapi tetap tersedia di AWS layanan yang menghostingnya (jika masih ada).

Pertimbangkan untuk menghapus target penerapan jika target menjadi basi. CodeCatalyst Target mungkin menjadi basi jika:

- Anda menghapus alur kerja yang diterapkan ke target.

- Anda mengubah tumpukan atau cluster yang Anda gunakan.
- Anda menghapus tumpukan atau cluster dari CloudFormation atau layanan Amazon ECS di AWS konsol.

Untuk menghapus target penerapan

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Lingkungan.
4. Pilih nama lingkungan yang berisi target penyebaran yang ingin Anda hapus. Untuk informasi tentang lingkungan, lihatMenyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst.
5. Pilih tab Target Deployment.
6. Pilih tombol radio di sebelah target penyebaran yang ingin Anda hapus.
7. Pilih Hapus.

   Target dihapus dari halaman.

Melacak status penerapan dengan komit

Kapan saja dalam siklus hidup pengembangan, penting untuk mengetahui status penerapan komit tertentu, seperti perbaikan bug, fitur baru, atau perubahan berdampak lainnya. Pertimbangkan skenario berikut di mana kemampuan pelacakan status penerapan sangat membantu tim pengembangan:

- Sebagai pengembang, Anda telah membuat perbaikan untuk mengatasi bug dan Anda ingin melaporkan status penerapannya di seluruh lingkungan penerapan tim Anda.
- Sebagai pengelola rilis, Anda ingin melihat daftar komit yang diterapkan untuk melacak dan melaporkan status penerapannya.

CodeCatalyst menyediakan tampilan yang dapat Anda gunakan untuk menentukan sekilas di mana setiap komit atau perubahan telah diterapkan, dan ke lingkungan mana. Tampilan ini meliputi:

- Daftar komit.
- Status penerapan yang mencakup komit.

- Lingkungan di mana komit berhasil digunakan.
- Status pengujian apa pun dijalankan terhadap komit dalam alur kerja CI/CD Anda.

Prosedur berikut merinci cara menavigasi ke dan menggunakan tampilan ini untuk melacak perubahan dalam proyek Anda.

> **Note**
>
> Melacak status penerapan dengan komit hanya didukung dengan CodeCatalyst repositori. Anda tidak dapat menggunakan fitur ini dengan GitHub repositori atau Bitbucket.

Untuk melacak status penerapan dengan komit

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Ubah pelacakan.
4. Dalam dua daftar tarik-turun di bagian atas panel utama, pilih repositori sumber dan cabang yang berisi komit yang status rilisnya ingin Anda lihat.
5. Pilih Lihat perubahan.

   Daftar komit muncul.

   Untuk setiap komit, Anda dapat melihat yang berikut:

   - Komit informasi seperti ID, penulis, pesan, dan kapan itu dilakukan. Untuk informasi selengkapnya, lihat Simpan dan berkolaborasi pada kode dengan repositori sumber di CodeCatalyst.
   - Status penerapan ke setiap lingkungan. Untuk informasi selengkapnya, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst.
   - Hasil uji dan cakupan kode. Untuk informasi selengkapnya, lihat Pengujian dengan alur kerja.

   > **Note**
   >
   > Hasil Analisis Komposisi Perangkat Lunak (SCA) tidak ditampilkan.

6. (Opsional) Untuk melihat informasi selengkapnya tentang perubahan yang terkait dengan komit tertentu, termasuk penerapan terbaru dan cakupan kode terperinci serta informasi pengujian unit, pilih Lihat detail untuk komit tersebut.

Melihat log penerapan

Anda dapat melihat log yang terkait dengan tindakan penerapan tertentu untuk memecahkan masalah di Amazon. CodeCatalyst

Anda dapat melihat log mulai dari alur kerja, atau lingkungan.

Untuk melihat log tindakan penerapan yang dimulai dari alur kerja

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Runs.
6. Pilih alur kerja yang menjalankan aplikasi Anda.
7. Dalam diagram alur kerja, pilih tindakan yang lognya ingin Anda lihat.
8. Pilih tab Log dan perluas bagian untuk mengungkapkan pesan log.
9. Untuk melihat lebih banyak log, pilih tab Ringkasan, lalu pilih Lihat di CloudFormation (jika tersedia) untuk melihat lebih banyak log di sana. Anda mungkin perlu masuk ke AWS.

Untuk melihat log tindakan penerapan yang dimulai dari lingkungan

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Lingkungan.
4. Pilih lingkungan tempat aplikasi Anda digunakan.
5. Dalam aktivitas Deployment, temukan kolom Workflow Run ID, dan pilih alur kerja yang menerapkan tumpukan Anda.
6. Dalam diagram alur kerja, pilih tindakan yang lognya ingin Anda lihat.
7. Pilih tab Log dan perluas bagian untuk mengungkapkan pesan log.

8. Untuk melihat lebih banyak log, pilih tab Ringkasan, lalu pilih Lihat di CloudFormation (jika tersedia) untuk melihat lebih banyak log di sana. Anda mungkin perlu masuk ke AWS.

Melihat status penerapan, komit, dan permintaan tarik

Anda dapat melihat informasi berikut tentang penerapan di Amazon CodeCatalyst:

- Aktivitas penerapan, termasuk status penerapan, waktu mulai, waktu akhir, riwayat, dan durasi acara.
- Nama tumpukan, Wilayah AWS, waktu pembaruan terakhir, dan alur kerja terkait.
- Komit dan tarik permintaan.
- Informasi khusus tindakan, misalnya, CloudFormation peristiwa dan output.

Anda dapat melihat informasi penerapan mulai dari alur kerja, lingkungan, atau tindakan alur kerja.

Untuk melihat informasi penerapan mulai dari alur kerja

- Buka alur kerja yang menerapkan aplikasi Anda. Untuk petunjuk, lihat Melihat alur kerja menjalankan status dan detail.

Untuk melihat informasi penyebaran mulai dari lingkungan

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Lingkungan.
4. Pilih lingkungan tempat tumpukan Anda digunakan, misalnya,`Production`.
5. Pilih aktivitas Deployment untuk melihat riwayat penerapan tumpukan Anda, status penerapan (misalnya, BERHASIL atau GAGAL), dan informasi terkait penerapan lainnya.
6. Pilih target Deployment untuk melihat informasi tentang tumpukan, kluster, atau target lain yang diterapkan ke lingkungan. Anda dapat melihat informasi seperti nama tumpukan, Wilayah, penyedia, dan pengenal.

Untuk melihat informasi penyebaran mulai dari tindakan

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.

2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Dalam diagram alur kerja, pilih tindakan alur kerja yang menerapkan aplikasi Anda. Misalnya, Anda mungkin memilih DeployCloudFormationStack.
6. Tinjau konten di panel kanan untuk informasi penyebaran khusus tindakan.

Membuat alur kerja

Alur kerja adalah prosedur otomatis yang menjelaskan cara membangun, menguji, dan menyebarkan kode Anda sebagai bagian dari sistem integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD). Alur kerja mendefinisikan serangkaian langkah, atau tindakan, yang harus diambil selama menjalankan alur kerja. Alur kerja juga mendefinisikan peristiwa, atau pemicu, yang menyebabkan alur kerja dimulai. Untuk menyiapkan alur kerja, Anda membuat file definisi alur kerja menggunakan editor visual atau YAMAL CodeCatalyst konsol.

> **Tip**
>
> Untuk melihat sekilas bagaimana Anda dapat menggunakan alur kerja dalam sebuah proyek, buat proyek dengan cetak biru. Setiap cetak biru menerapkan alur kerja yang berfungsi yang dapat Anda tinjau, jalankan, dan bereksperimen.

Gunakan prosedur berikut untuk membuat alur kerja di CodeCatalyst.

Untuk informasi lebih lanjut tentang alur kerja, lihat Bangun, uji, dan terapkan dengan alur kerja di CodeCatalyst.

Visual

Untuk membuat alur kerja menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.

4. Pilih Buat alur kerja.

   Kotak dialog Buat alur kerja muncul.

5. Di bidang repositori Sumber, pilih repositori sumber tempat file definisi alur kerja akan berada. File akan disimpan di `~/.codecatalyst/workflows/` folder di repositori yang dipilih. Jika tidak ada repositori sumber, buat satu.
6. Di bidang Cabang, pilih cabang tempat file definisi alur kerja akan berada.
7. Pilih Buat.

   Amazon CodeCatalyst menyimpan informasi repositori dan cabang dalam memori, tetapi alur kerja belum dilakukan.

8. Pilih Visual.
9. Bangun alur kerja:

   a. (Opsional) Dalam diagram alur kerja, pilih kotak Sumber dan Pemicu. Panel Pemicu muncul. Pilih Tambah pemicu untuk menambahkan pemicu. Untuk informasi selengkapnya, lihat Menambahkan pemicu push, pull, atau schedule.

   b. Pilih + Tindakan (kiri atas). Katalog Tindakan muncul.

   c. Pilih tanda plus (+) di dalam tindakan untuk menambahkannya ke alur kerja. Gunakan panel di sebelah kanan untuk mengonfigurasi tindakan. Untuk informasi selengkapnya, lihat Menambahkan tindakan ke CodeCatalyst alur kerja.

   d. (Opsional) Pilih properti Alur Kerja (kanan atas). Sebuah panel properti alur kerja muncul. Konfigurasikan mode jalankan nama alur kerja, dan hitung. Untuk informasi selengkapnya, lihat Mengonfigurasi perilaku antrian run dan Mengkonfigurasi gambar Docker lingkungan komputasi dan runtime untuk alur kerja.

10. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
11. Pilih Commit, dan pada kotak dialog Commit workflow, lakukan hal berikut:

    a. Untuk nama file Workflow, tinggalkan nama default atau masukkan nama Anda sendiri.

    b. Untuk pesan Commit, tinggalkan pesan default atau masukkan pesan Anda sendiri.

    c. Untuk Repositori dan Cabang, pilih repositori sumber dan cabang untuk file definisi alur kerja. Bidang ini harus diatur ke repositori dan cabang yang Anda tentukan sebelumnya di kotak dialog Buat alur kerja. Anda dapat mengubah repositori dan cabang sekarang, jika Anda mau.

> **Note**
>
> Setelah melakukan file definisi alur kerja Anda, itu tidak dapat dikaitkan dengan repositori atau cabang lain, jadi pastikan untuk memilihnya dengan hati-hati.

d. Pilih Komit untuk melakukan berkas definisi alur kerja.

YAML

Untuk membuat alur kerja menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih Buat alur kerja.

   Kotak dialog Buat alur kerja muncul.
5. Di bidang repositori Sumber, pilih repositori sumber tempat file definisi alur kerja akan berada. File akan disimpan di `~/.codecatalyst/workflows/` folder di repositori yang dipilih. Jika tidak ada repositori sumber, buat satu.
6. Di bidang Cabang, pilih cabang tempat file definisi alur kerja akan berada.
7. Pilih Buat.

   Amazon CodeCatalyst menyimpan informasi repositori dan cabang dalam memori, tetapi alur kerja belum dilakukan.
8. Pilih YAMAL.
9. Bangun alur kerja:

   a. (Opsional) Tambahkan pemicu ke kode YAMAL. Untuk informasi selengkapnya, lihat Menambahkan pemicu push, pull, atau schedule.

   b. Pilih + Tindakan (kiri atas). Katalog Tindakan muncul.

   c. Pilih tanda plus (+) di dalam tindakan untuk menambahkannya ke alur kerja. Gunakan panel di sebelah kanan untuk mengonfigurasi tindakan. Untuk informasi selengkapnya, lihat Menambahkan tindakan ke CodeCatalyst alur kerja.

d. (Opsional) Pilih properti Alur Kerja (kanan atas). Sebuah panel properti alur kerja muncul. Konfigurasikan nama alur kerja, mode jalankan, dan komputasi. Untuk informasi selengkapnya, lihat Mengonfigurasi perilaku antrian run dan Mengkonfigurasi gambar Docker lingkungan komputasi dan runtime untuk alur kerja.

10. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
11. Pilih Commit, dan pada kotak dialog Commit workflow, lakukan hal berikut:

    a. Untuk nama file Workflow, tinggalkan nama default atau masukkan nama Anda sendiri.

    b. Untuk pesan Commit, tinggalkan pesan default atau masukkan pesan Anda sendiri.

    c. Untuk Repositori dan Cabang, pilih repositori sumber dan cabang untuk file definisi alur kerja. Bidang ini harus diatur ke repositori dan cabang yang Anda tentukan sebelumnya di kotak dialog Buat alur kerja. Anda dapat mengubah repositori dan cabang sekarang, jika Anda mau.

    > **Note**
    >
    > Setelah melakukan file definisi alur kerja Anda, itu tidak dapat dikaitkan dengan repositori atau cabang lain, jadi pastikan untuk memilihnya dengan hati-hati.

    d. Pilih Komit untuk melakukan berkas definisi alur kerja.

Menjalankan alur kerja

Run adalah iterasi tunggal dari alur kerja. Selama menjalankan, CodeCatalyst melakukan tindakan yang ditentukan dalam file konfigurasi alur kerja dan mengeluarkan log, artefak, dan variabel terkait.

Anda dapat memulai proses secara manual, atau Anda dapat memulainya secara otomatis, melalui pemicu alur kerja. Contoh pemicu alur kerja mungkin pengembang perangkat lunak mendorong komit ke cabang utama Anda.

Anda juga dapat menghentikan alur kerja secara manual di tengah pemrosesannya jika Anda memulainya secara tidak sengaja.

Jika beberapa alur kerja berjalan dimulai sekitar waktu yang sama, Anda dapat mengonfigurasi bagaimana Anda ingin proses ini diantrian. Anda dapat menggunakan perilaku antrian default, di mana proses diantrian satu demi satu dalam urutan di mana mereka dimulai, atau Anda dapat memiliki run yang lebih baru menggantikan (atau 'mengambil alih') dari yang sebelumnya untuk

mempercepat proses Anda. Menyiapkan alur kerja Anda agar terjadi secara paralel, sehingga tidak ada proses menunggu yang lain, juga dimungkinkan.

Setelah memulai alur kerja, baik secara manual maupun otomatis, Anda dapat melihat status proses dan detail lainnya. Misalnya, Anda dapat melihat kapan dimulai, siapa yang dimulai, dan apakah masih berjalan.

Topik

Memulai alur kerja berjalan secara manual

Gunakan prosedur berikut untuk memulai alur kerja yang dijalankan secara manual.

> **Note**
>
> Anda juga dapat memulai alur kerja yang dijalankan secara otomatis dengan mengonfigurasi pemicu.

Untuk memulai alur kerja, jalankan secara manual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja yang ingin Anda jalankan. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.

5. Pilih Jalankan.

Memulai alur kerja berjalan secara otomatis dengan pemicu

Pemicu alur kerja, atau hanya pemicu, memungkinkan Anda memulai alur kerja yang berjalan secara otomatis ketika peristiwa tertentu terjadi, seperti push kode. Anda mungkin ingin mengonfigurasi pemicu untuk membebaskan pengembang perangkat lunak Anda dari keharusan memulai alur kerja yang berjalan secara manual melalui konsol. CodeCatalyst

Anda dapat menggunakan tiga jenis pemicu:

- Push - Pemicu push kode menyebabkan alur kerja dijalankan setiap kali komit didorong.
- Permintaan tarik - Pemicu permintaan tarik menyebabkan alur kerja dijalankan setiap kali permintaan tarik dibuat, direvisi, atau ditutup.
- Jadwal - Pemicu jadwal menyebabkan alur kerja berjalan dimulai pada jadwal yang Anda tentukan. Pertimbangkan untuk menggunakan pemicu jadwal untuk menjalankan build malam perangkat lunak Anda sehingga build terbaru siap untuk pengembang perangkat lunak Anda untuk bekerja keesokan paginya.

Anda dapat menggunakan push, pull request, dan schedule trigger sendiri atau dalam kombinasi dalam alur kerja yang sama.

> **Tip**
>
> Untuk melihat pemicu beraksi, luncurkan proyek dengan cetak biru. Sebagian besar cetak biru berisi alur kerja dengan pemicu. Cari `Trigger` properti di file definisi alur kerja cetak biru. Untuk informasi selengkapnya tentang cetak biru, lihat Membuat proyek dengan cetak biru.

Topik

Konfigurasi pemicu umum

Bagian ini menjelaskan cara mengatur pemicu untuk rilis perangkat lunak umum dan strategi percabangan.

Pelepasan perangkat lunak dan strategi percabangan:

- Anda memiliki kode aplikasi di repositori sumber.
- `main`Cabang Anda berisi kode finalisasi yang selalu siap rilis.
- Pengembang perangkat lunak Anda membuat perubahan mereka di cabang fitur di luar `main` cabang.
- Pengembang perangkat lunak Anda membuat permintaan tarik yang meminta untuk menggabungkan cabang fitur mereka `main` saat fitur mereka siap.

  Anda ingin permintaan tarik ini memulai alur kerja secara otomatis yang membangun dan menguji — tetapi tidak menyebarkan — aplikasi menggunakan file di cabang fitur pengembang perangkat lunak.
- Pengembang perangkat lunak Anda memeriksa build dan tes untuk memastikan semuanya terlihat bagus. Mereka kemudian menggabungkan permintaan tarik ke `main` cabang.

  Anda ingin penggabungan secara otomatis memulai alur kerja yang membangun dan menyebarkan kode aplikasi Anda secara otomatis.

Konfigurasi alur kerja/pemicu yang diusulkan:

Mengingat strategi percabangan perangkat lunak yang diuraikan sebelumnya, Anda mungkin ingin menggunakan dua alur kerja:

- Workflow 1 membangun dan menguji aplikasi Anda saat permintaan tarik dibuat atau direvisi.
- Workflow 2 membangun dan menerapkan aplikasi Anda saat permintaan tarik digabungkan.

Alur kerja 1 akan terlihat seperti ini:

```
Triggers:
  - Type: PULLREQUEST
    Branches:
      - main
    Events:
```

```
      - OPEN
      - REVISION
Actions:
  BuildAction:
    instructions-for-building-the-app
  TestAction:
    instructions-for-test-the-app
```

Kode pemicu sebelumnya secara otomatis memulai alur kerja yang dijalankan setiap kali pengembang perangkat lunak membuat permintaan tarik (atau memodifikasinya) yang meminta untuk menggabungkan cabang fitur mereka ke cabang. `main` CodeCatalyst memulai alur kerja yang dijalankan menggunakan kode di cabang sumber (yaitu, cabang fitur pengembang). Alur kerja membangun dan menyebarkan aplikasi.

Alur kerja 2 akan terlihat seperti ini:

```
Triggers:
  - Type: PUSH
    Branches:
      - main
Actions:
  BuildAction:
    instructions-for-building-the-app
  DeployAction:
    instructions-for-deploying-the-app
```

Dalam kode pemicu sebelumnya, ketika penggabungan `main` terjadi, `PUSH` pemicu diaktifkan. CodeCatalyst memulai alur kerja yang dijalankan menggunakan kode di `main` cabang (yang sekarang menyertakan kode dari permintaan tarik). Alur kerja membangun dan menyebarkan aplikasi.

Untuk petunjuk tentang menambahkan pemicu ke file definisi alur kerja, lihat. Menambahkan pemicu push, pull, atau schedule

Untuk lebih banyak contoh pemicu dan penjelasan tambahan, lihat. Contoh pemicu

Memicu pertimbangan saat bercabang

Bagian ini menjelaskan beberapa pertimbangan utama saat menyiapkan pemicu yang mencakup cabang.

- Pertimbangan 1: Untuk pemicu permintaan push dan pull, jika Anda akan menentukan cabang, Anda harus menentukan cabang tujuan (atau 'to') dalam konfigurasi pemicu. Jangan pernah menentukan sumber (atau 'dari') cabang.

  Dalam contoh berikut, push dari cabang manapun untuk `main` mengaktifkan alur kerja.

  ```
  Triggers:
    - Type: PUSH
      Branches:
        - main
  ```

  Dalam contoh berikut, permintaan tarik dari cabang mana pun ke `main` mengaktifkan alur kerja.

  ```
  Triggers:
    - Type: PULLREQUEST
      Branches:
        - main
      Events:
        - OPEN
        - REVISION
  ```

- Pertimbangan 2: Untuk pemicu push, setelah alur kerja diaktifkan, alur kerja akan berjalan menggunakan file definisi alur kerja dan file sumber di cabang tujuan.
- Pertimbangan 3: Untuk pemicu permintaan tarik, setelah alur kerja diaktifkan, alur kerja akan berjalan menggunakan file definisi alur kerja dan file sumber di cabang sumber (meskipun Anda menentukan cabang tujuan dalam konfigurasi pemicu).
- Pertimbangan 4: Pemicu yang sama persis di satu cabang mungkin tidak berjalan di cabang lain.

  Pertimbangkan pemicu dorong berikut:

  ```
  Triggers:
    - Type: PUSH
      Branches:
        - main
  ```

  Jika file definisi alur kerja yang berisi pemicu ini ada `main` dan dikloning`test`, alur kerja tidak akan pernah mulai secara otomatis menggunakan file di `test` (meskipun Anda dapat memulai alur kerja secara manual untuk memiliki jika menggunakan file di). `test` Tinjau Pertimbangan 1 dan 2 untuk

memahami mengapa alur kerja tidak akan pernah berjalan secara otomatis menggunakan file di. `test`

Pertimbangkan juga pemicu permintaan tarik berikut:

```
Triggers:
  - Type: PULLREQUEST
    Branches:
      - main
    Events:
      - OPEN
      - REVISION
```

Jika file definisi alur kerja yang berisi pemicu ini ada di`main`, alur kerja tidak akan pernah berjalan menggunakan file di. `main` (Namun, jika Anda membuat `test` cabang dari`main`, alur kerja akan berjalan menggunakan file di`test`.) Tinjau Pertimbangan 1 dan 3 untuk memahami alasannya.

Menambahkan pemicu push, pull, atau schedule

Gunakan petunjuk berikut untuk menambahkan pemicu push, pull, atau schedule ke alur kerja Anda.

Visual

Untuk menambahkan pemicu (editor visual)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Dalam diagram alur kerja, pilih kotak Sumber dan Pemicu.
8. Di panel konfigurasi, Pilih Tambah pemicu.
9. Dalam Tambahkan pemicu kotak dialog, berikan informasi di bidang, sebagai berikut.

   Jenis pemicu

Tentukan jenis pemicu. Anda dapat menggunakan salah satu nilai berikut:

- Dorong (editor visual) atau `PUSH` (editor YAMAL)

  Pemicu push memulai alur kerja saat perubahan didorong ke repositori sumber Anda. Workflow run akan menggunakan file di cabang yang Anda dorong (yaitu, cabang tujuan).

- Permintaan tarik (editor visual) atau `PULLREQUEST` (editor YAMG)

  Pemicu permintaan tarik memulai alur kerja saat permintaan tarik dibuka, diperbarui, atau ditutup di repositori sumber Anda. Workflow run akan menggunakan file di cabang yang Anda tarik dari (yaitu, cabang sumber).

- Jadwal (editor visual) atau `SCHEDULE` (editor YAMAL)

  Pemicu jadwal memulai alur kerja berjalan pada jadwal yang ditentukan oleh ekspresi cron yang Anda tentukan. Jalankan alur kerja terpisah akan dimulai untuk setiap cabang di repositori sumber Anda menggunakan file cabang. (Untuk membatasi cabang tempat pemicu diaktifkan, gunakan bidang Cabang (editor visual) atau `Branches` properti (editor YAMG).)

  Saat mengonfigurasi pemicu jadwal, ikuti panduan ini:

  - Hanya gunakan satu pemicu jadwal per alur kerja.
  - Jika Anda telah menentukan beberapa alur kerja di CodeCatalyst ruang Anda, sebaiknya Anda menjadwalkan tidak lebih dari 10 alur kerja untuk memulai secara bersamaan.
  - Pastikan Anda mengonfigurasi ekspresi cron pemicu dengan waktu yang cukup di antara proses. Untuk informasi selengkapnya, lihat Expression.

Sebagai contoh, silakan lihat Contoh pemicu.

Acara untuk permintaan tarik

Bidang ini hanya muncul jika Anda memilih jenis pemicu permintaan Tarik.

Tentukan jenis peristiwa permintaan tarik yang akan memulai alur kerja. Berikut ini adalah nilai yang valid:

- Permintaan tarik dibuat (editor visual) atau `OPEN` (editor YAMG)

  Proses alur kerja dimulai saat permintaan tarik dibuat.

- Permintaan tarik ditutup (editor visual) atau `CLOSED` (editor YAMG)

  Proses alur kerja dimulai saat permintaan tarik ditutup. Perilaku `CLOSED` acara itu rumit, dan paling baik dipahami melalui sebuah contoh. Untuk informasi selengkapnya, lihat Contoh: Pemicu dengan tarikan, cabang, dan acara 'CLOSED'.

- Revisi baru dibuat untuk menarik permintaan (editor visual) atau `REVISION` (editor YAMG)

  Alur kerja dijalankan ketika revisi ke permintaan tarik dibuat. Revisi pertama dibuat saat permintaan tarik dibuat. Setelah itu, revisi baru dibuat setiap kali seseorang mendorong komit baru ke cabang sumber yang ditentukan dalam permintaan tarik. Jika Anda menyertakan `REVISION` peristiwa dalam pemicu permintaan tarik Anda, Anda dapat menghilangkan `OPEN` acara tersebut, karena `REVISION` merupakan superset dari. `OPEN`

Anda dapat menentukan beberapa peristiwa dalam pemicu permintaan tarik yang sama.

Sebagai contoh, lihat Contoh pemicu.

Jadwal

Bidang ini hanya muncul jika Anda memilih jenis pemicu Jadwal.

Tentukan ekspresi cron yang menjelaskan kapan Anda ingin alur kerja terjadwal Anda berjalan terjadi.

Ekspresi cron CodeCatalyst menggunakan sintaks enam bidang berikut, di mana setiap bidang dipisahkan oleh spasi:

```
menit jam days-of-monthbulan days-of-weektahun
```

Contoh ekspresi cron

Menit	Jam	Hari dalam sebulan	Bulan	Hari dalam seminggu	Tahun	Arti
0	0	?	*	MON-FRI	*	Menjalankan alur kerja pada tengah malam (UTC+0) setiap Senin sampai Jumat.
0	2	*	*	?	*	Menjalankan alur kerja pada pukul 2:00 pagi (UTC+0) setiap hari.
15	22	*	*	?	*	Menjalankan alur kerja pada pukul 10:15 malam (UTC+0) setiap hari.

Menit	Jam	Hari dalam sebulan	Bulan	Hari dalam seminggu	Tahun	Arti
0/30	22-2	?	*	SAT-MATAHARI	*	Menjalankan alur kerja setiap 30 menit Sabtu hingga Minggu antara pukul 10:00 malam pada hari mulai dan 2:00 pagi pada hari berikutnya (UTC+0).

Menit	Jam	Hari dalam sebulan	Bulan	Hari dalam seminggu	Tahun	Arti
45	13	L	*	?	2023-2027	Menjalankan alur kerja pada pukul 1:45 siang (UTC+0) pada hari terakhir bulan antara tahun 2023 dan 2027 inklusif.

Saat menentukan ekspresi cron di CodeCatalyst, pastikan Anda mengikuti panduan ini:

- Tentukan ekspresi cron tunggal per `SCHEDULE` pemicu.
- Lampirkan ekspresi cron dalam tanda kutip ganda (") di editor YAMG.
- Tentukan waktu di Coordinated Universal Time (UTC). Zona waktu lain tidak didukung.
- Konfigurasikan setidaknya 30 menit di antara proses. Irama yang lebih cepat tidak didukung.
- Tentukan *`days-of-week`*bidang *`days-of-month`*atau, tetapi tidak keduanya. Jika Anda menentukan nilai atau tanda bintang (*) di salah satu bidang, Anda harus menggunakan tanda tanya (?) di bidang lainnya. Tanda bintang berarti 'semua' dan tanda tanya berarti 'apa saja'.

Untuk lebih banyak contoh ekspresi cron dan informasi tentang wildcard seperti?,*, danL, lihat referensi ekspresi Cron di Panduan Pengguna Amazon EventBridge . Ekspresi cron masuk EventBridge dan CodeCatalyst bekerja dengan cara yang persis sama.

Untuk contoh pemicu jadwal, lihatContoh pemicu.

Cabang dan pola Cabang

(Opsional)

Tentukan cabang di repositori sumber Anda yang dipantau pemicu untuk mengetahui kapan harus memulai alur kerja. Anda dapat menggunakan pola regex untuk menentukan nama cabang Anda. Misalnya, gunakan `main.*` untuk mencocokkan semua cabang yang dimulai dengan`main`.

Cabang yang akan ditentukan berbeda tergantung pada jenis pemicu:

- Untuk pemicu push, tentukan cabang yang Anda dorong, yaitu cabang tujuan. Satu alur kerja akan dimulai per cabang yang cocok, menggunakan file di cabang yang cocok.

  Contoh:`main.*`, `mainline`
- Untuk pemicu permintaan tarik, tentukan cabang yang Anda dorong, yaitu cabang tujuan. Satu alur kerja akan dimulai per cabang yang cocok, menggunakan file definisi alur kerja dan file sumber di cabang sumber (bukan cabang yang cocok).

  Contoh:`main.*`,`mainline`, `v1\-.*` (cocok dengan cabang yang dimulai dengan`v1-`)
- Untuk pemicu jadwal, tentukan cabang yang berisi file yang ingin Anda jalankan terjadwal untuk digunakan. Satu alur kerja akan dimulai per cabang yang cocok, menggunakan file definisi alur kerja dan file sumber di cabang yang cocok.

  Contoh:`main.*`, `version\-1\.0`

> **Note**
>
> Jika Anda tidak menentukan cabang, pemicu memantau semua cabang di repositori sumber Anda, dan akan memulai alur kerja menggunakan file definisi alur kerja dan file sumber di:
>
> - Cabang yang Anda dorong (untuk pemicu push). Untuk informasi selengkapnya, lihat Contoh: Pemicu push kode sederhana.
> - Cabang yang Anda tarik (untuk pemicu permintaan tarik). Untuk informasi selengkapnya, lihat Contoh: Pemicu permintaan tarik sederhana.

- Semua cabang (untuk pemicu jadwal). Satu alur kerja akan dimulai per cabang di repositori sumber Anda. Untuk informasi selengkapnya, lihat Contoh: Pemicu jadwal sederhana.

Untuk informasi lebih lanjut tentang cabang dan pemicu, lihatMemicu pertimbangan saat bercabang.

Untuk contoh lainnya, lihat Contoh pemicu.

File diubah

Bidang ini hanya muncul jika Anda memilih jenis pemicu permintaan Push atau Pull.

Tentukan file atau folder di repositori sumber Anda yang dipantau pemicu untuk mengetahui kapan harus memulai alur kerja. Anda dapat menggunakan ekspresi reguler untuk mencocokkan nama file atau jalur.

Sebagai contoh, lihat Contoh pemicu.

10. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
11. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menambahkan pemicu (editor YAMAL)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Tambahkan `Triggers` bagian dan properti yang mendasari menggunakan contoh berikut sebagai panduan. Untuk informasi lebih lanjut, lihat Triggers diAlur kerja definisi YAMAL.

Pemicu push kode mungkin terlihat seperti ini:

```
Triggers:
  - Type: PUSH
    Branches:
      - main
```

Pemicu permintaan tarik mungkin terlihat seperti ini:

```
Triggers:
  - Type: PULLREQUEST
    Branches:
      - main.*
    Events:
      - OPEN
      - REVISION
      - CLOSED
```

Pemicu jadwal mungkin terlihat seperti ini:

```
Triggers:
  - Type: SCHEDULE
    Branches:
      - main.*
    # Run the workflow at 1:15 am (UTC+0) every Friday until the end of 2023
    Expression: "15 1 ? * FRI 2022-2023"
```

Untuk lebih banyak contoh ekspresi cron yang dapat Anda gunakan di `Expression` properti, lihatExpression.

Untuk lebih banyak contoh push, pull request, dan schedule trigger, lihatContoh pemicu.

8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Contoh pemicu

Contoh berikut menunjukkan cara menambahkan berbagai jenis pemicu dalam file definisi alur kerja.

Topik

- Contoh: Pemicu push kode sederhana

- Contoh: Pemicu 'push to main' sederhana
- Contoh: Pemicu permintaan tarik sederhana
- Contoh: Pemicu jadwal sederhana
- Contoh: Pemicu dengan jadwal dan cabang
- Contoh: Pemicu dengan jadwal, dorongan, dan cabang
- Contoh: Pemicu dengan tarikan dan cabang
- Contoh: Pemicu dengan tarikan, cabang, dan acara 'CLOSED'
- Contoh: Pemicu dengan dorongan, cabang, dan file

Contoh: Pemicu push kode sederhana

Contoh berikut menunjukkan pemicu yang memulai alur kerja yang dijalankan setiap kali kode didorong ke cabang mana pun di repositori sumber Anda.

Ketika pemicu ini diaktifkan, CodeCatalyst mulai menjalankan alur kerja menggunakan file di cabang yang Anda dorong (yaitu, cabang tujuan).

Misalnya, jika Anda mendorong komit ke`main`, CodeCatalyst mulai menjalankan alur kerja menggunakan file definisi workfow dan file sumber lainnya. `main`

Sebagai contoh lain, jika Anda mendorong komit ke`feature-branch-123`, CodeCatalyst mulai menjalankan alur kerja menggunakan file definisi workfow dan file sumber lainnya. `feature-branch-123`

```
Triggers:
  - Type: PUSH
```

> **Note**
>
> Jika Anda ingin alur kerja dijalankan hanya ketika Anda mendorong ke`main`, lihatContoh: Pemicu 'push to main' sederhana.

Contoh: Pemicu 'push to main' sederhana

Contoh berikut menunjukkan pemicu yang memulai alur kerja yang dijalankan setiap kali kode didorong ke `main` cabang—dan hanya cabang—di repositori sumber Anda. `main`

```
Triggers:
  - Type: PUSH
    Branches:
      - main
```

Contoh: Pemicu permintaan tarik sederhana

Contoh berikut menunjukkan pemicu yang memulai alur kerja yang dijalankan setiap kali permintaan tarik dibuat atau direvisi di repositori sumber Anda.

Ketika pemicu ini diaktifkan, CodeCatalyst mulai menjalankan alur kerja menggunakan file definisi alur kerja dan file sumber lain di cabang yang Anda tarik (yaitu, cabang sumber).

Misalnya, jika Anda membuat permintaan tarik dengan cabang sumber yang dipanggil `feature-123` dan cabang tujuan dipanggil`main`, CodeCatalyst mulai menjalankan alur kerja menggunakan file definisi workfow dan file sumber lainnya. `feature-123`

```
Triggers:
  - Type: PULLREQUEST
    Events:
      - OPEN
      - REVISION
```

Contoh: Pemicu jadwal sederhana

Contoh berikut menunjukkan pemicu yang memulai alur kerja berjalan pada tengah malam (UTC+0) setiap Senin sampai Jumat.

Ketika pemicu ini diaktifkan, CodeCatalyst mulai menjalankan alur kerja tunggal untuk setiap cabang di repositori sumber Anda yang berisi file definisi alur kerja dengan pemicu ini.

Misalnya, jika Anda memiliki tiga cabang di repositori sumber Anda,,`main`, `release-v1``feature-123`, dan masing-masing cabang ini berisi file definisi alur kerja dengan pemicu berikut, CodeCatalyst mulai tiga alur kerja berjalan: satu menggunakan file di, yang lain menggunakan file di `main``release-v1`, dan yang lain menggunakan file di. `feature-123`

```
Triggers:
  - Type: SCHEDULE
    Expression: "0 0 ? * MON-FRI *"
```

Untuk lebih banyak contoh ekspresi cron yang dapat Anda gunakan di `Expression` properti, lihatExpression.

Contoh: Pemicu dengan jadwal dan cabang

Contoh berikut menunjukkan pemicu yang memulai alur kerja berjalan pada pukul 18:15 (UTC+0) setiap hari.

Ketika pemicu ini diaktifkan, CodeCatalyst mulai menjalankan alur kerja menggunakan file di `main` cabang, dan memulai proses tambahan untuk setiap cabang yang dimulai dengan`release-`.

Misalnya, jika Anda memiliki cabang bernama`main`,, `release-v1``bugfix-1`, dan `bugfix-2` di repositori sumber Anda, CodeCatalyst mulai dua alur kerja berjalan: satu menggunakan file di`main`, dan yang lain menggunakan file di. `release-v1` Itu tidak memulai alur kerja berjalan untuk `bugfix-1` cabang `bugfix-1` dan.

```
Triggers:
  - Type: SCHEDULE
    Expression: "15 18 * * ? *"
    Branches:
      - main
      - release\-.*
```

Untuk lebih banyak contoh ekspresi cron yang dapat Anda gunakan di `Expression` properti, lihatExpression.

Contoh: Pemicu dengan jadwal, dorongan, dan cabang

Contoh berikut menunjukkan pemicu yang memulai alur kerja berjalan pada tengah malam (UTC+0) setiap hari, dan setiap kali kode didorong ke cabang. `main`

Dalam contoh ini:

- Alur kerja dimulai pada tengah malam setiap hari. Jalankan alur kerja menggunakan file definisi alur kerja dan file sumber lainnya di cabang. `main`
- Jalankan alur kerja juga dimulai setiap kali Anda mendorong komit ke `main` cabang. Jalankan alur kerja menggunakan file definisi alur kerja dan file sumber lainnya di cabang tujuan ()`main`.

```
Triggers:
  - Type: SCHEDULE
```

```
    Expression: "0 0 * * ? *"
    Branches:
      - main
  - Type: PUSH
    Branches:
      - main
```

Untuk lebih banyak contoh ekspresi cron yang dapat Anda gunakan di `Expression` properti, lihatExpression.

Contoh: Pemicu dengan tarikan dan cabang

Contoh berikut menunjukkan pemicu yang memulai alur kerja yang dijalankan setiap kali seseorang membuka atau memodifikasi permintaan tarik dengan cabang tujuan yang dipanggil. `main` Meskipun cabang yang ditentukan dalam *`Triggers`* konfigurasi adalah*`main`*, alur kerja yang dijalankan akan menggunakan file definisi alur kerja dan file sumber lainnya di cabang sumber (yang merupakan cabang yang Anda tarik).

```
Triggers:
  - Type: PULLREQUEST
    Branches:
      - main
    Events:
      - OPEN
      - REVISION
```

Contoh: Pemicu dengan tarikan, cabang, dan acara 'CLOSED'

Contoh berikut menunjukkan pemicu yang memulai alur kerja yang dijalankan setiap kali permintaan tarik ditutup pada cabang yang dimulai dengan`main`.

Dalam contoh ini:

- Saat Anda menutup permintaan tarik dengan cabang tujuan yang dimulai`main`, alur kerja dimulai secara otomatis menggunakan file definisi alur kerja dan file sumber lainnya di cabang sumber (sekarang ditutup).
- Jika Anda telah mengonfigurasi repositori sumber Anda untuk menghapus cabang secara otomatis setelah permintaan tarik digabungkan, cabang ini tidak akan pernah memiliki kesempatan untuk memasuki status. `CLOSED` Ini berarti bahwa cabang yang digabungkan tidak akan mengaktifkan `CLOSED` pemicu permintaan tarik. Satu-satunya cara untuk mengaktifkan `CLOSED` pemicu dalam skenario ini adalah dengan menutup permintaan tarik tanpa menggabungkannya.

```
Triggers:
  - Type: PULLREQUEST
    Branches:
      - main.*
    Events:
      - CLOSED
```

Contoh: Pemicu dengan dorongan, cabang, dan file

Contoh berikut menunjukkan pemicu yang memulai alur kerja yang dijalankan setiap kali perubahan dilakukan pada `filename.txt` file, atau file apa pun di `src` direktori, di `main` cabang.

Ketika pemicu ini diaktifkan, CodeCatalyst mulai menjalankan alur kerja menggunakan file definisi alur kerja dan file sumber lainnya di `main` cabang.

```
Triggers:
  - Type: PUSH
    Branches:
      - main
    FilesChanged:
      - filename.txt
      - src\/.*
```

Menghentikan alur kerja

Gunakan prosedur berikut untuk menghentikan alur kerja yang sedang berlangsung. Anda mungkin ingin berhenti berlari jika dimulai secara tidak sengaja.

Saat Anda menghentikan proses alur kerja, CodeCatalyst tunggu tindakan yang sedang berlangsung selesai sebelum menandai proses sebagai Berhenti di konsol. CodeCatalyst Setiap tindakan yang tidak memiliki kesempatan untuk memulai tidak akan dimulai, dan akan ditandai sebagai Ditinggalkan.

> **Note**
>
> Jika run antri (yaitu, tidak ada tindakan yang sedang berlangsung), maka run segera dihentikan.

Untuk menghentikan alur kerja

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Di bawah Alur kerja, pilih Runs dan pilih run dalam proses dari daftar.
5. Pilih Berhenti.

Mengembangkan alur kerja

Gate adalah komponen alur kerja yang dapat Anda gunakan untuk mencegah alur kerja berjalan kecuali kondisi tertentu terpenuhi. Contoh gerbang adalah gerbang Persetujuan tempat pengguna harus mengirimkan persetujuan di CodeCatalyst konsol sebelum proses alur kerja diizinkan untuk dilanjutkan.

Anda dapat menambahkan gerbang di antara urutan tindakan dalam alur kerja, atau sebelum tindakan pertama (yang berjalan segera setelah Sumber diunduh). Anda juga dapat menambahkan gerbang setelah tindakan terakhir, jika Anda memiliki kebutuhan untuk melakukannya.

Jenis gerbang

Saat ini, Amazon CodeCatalyst mendukung satu jenis gerbang: gerbang Persetujuan. Untuk informasi selengkapnya, lihat Memerlukan persetujuan pada alur kerja berjalan.

Bisakah saya mengatur gerbang untuk berjalan secara paralel dengan tindakan lain?

Tidak. Gates hanya dapat berjalan sebelum atau sesudah suatu tindakan. Untuk informasi selengkapnya, lihat Menyiapkan dependensi antara gerbang dan tindakan.

Dapatkah saya menggunakan gerbang untuk mencegah alur kerja berjalan dari awal?

Ya, dengan kualifikasi.

Anda dapat mencegah alur kerja menjalankan tugas, yang sedikit berbeda dari mencegahnya memulai.

Untuk mencegah alur kerja melakukan tugas, tambahkan gerbang sebelum tindakan pertama dalam alur kerja. Dalam skenario ini, alur kerja akan dimulai —artinya akan mengunduh file repositori sumber Anda—tetapi akan dicegah melakukan tugas sampai gerbang dibuka kuncinya.

> **Note**
>
> Alur kerja yang dimulai dan kemudian diblokir oleh gerbang masih dihitung terhadap jumlah maksimum alur kerja bersamaan berjalan per kuota ruang dan kuota lainnya. Untuk memastikan bahwa Anda tidak melebihi kuota alur kerja, pertimbangkan untuk menggunakan pemicu alur kerja untuk memulai alur kerja secara kondisional alih-alih menggunakan gerbang. Pertimbangkan juga untuk menggunakan aturan persetujuan permintaan tarik alih-alih gerbang. Untuk informasi selengkapnya tentang kuota, pemicu, dan aturan persetujuan permintaan tarik, lihat Kuota untuk alur kerjaMemulai alur kerja berjalan secara otomatis dengan pemicu, dan. Mengelola persyaratan untuk menggabungkan permintaan tarik dengan aturan persetujuan

Keterbatasan gerbang

Gates memiliki batasan sebagai berikut:

- Gates tidak dapat digunakan bersama dengan fitur berbagi komputasi. Untuk informasi selengkapnya tentang fitur ini, lihat Berbagi komputasi di seluruh tindakan.
- Gates tidak dapat digunakan dalam kelompok aksi. Untuk informasi selengkapnya tentang grup aksi, lihatMengelompokkan tindakan ke dalam kelompok aksi.

Topik

- Menambahkan gerbang ke alur kerja
- Menyiapkan dependensi antara gerbang dan tindakan
- Menentukan versi mayor, minor, atau patch dari gerbang

Menambahkan gerbang ke alur kerja

Gunakan petunjuk berikut untuk menambahkan gerbang ke alur kerja dan kemudian mengkonfigurasinya.

Untuk menambah dan mengkonfigurasi gerbang

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.

3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Di sebelah kiri, pilih Gates.
8. Di katalog gerbang, cari gerbang, lalu pilih tanda plus (+) untuk menambahkan gerbang ke alur kerja Anda.
9. Konfigurasikan gerbang. Pilih Visual untuk menggunakan editor visual, atau YAMAL untuk menggunakan editor YAMAL. Untuk petunjuk terperinci, lihat:
   - Menambahkan gerbang “Persetujuan” ke alur kerja
10. (Opsional) Pilih Validasi untuk memastikan kode YAMAL valid.
11. Pilih Komit untuk melakukan perubahan Anda.

Menyiapkan dependensi antara gerbang dan tindakan

Anda dapat mengatur gerbang untuk dijalankan sebelum atau sesudah tindakan, grup aksi, atau gerbang. Misalnya, Anda dapat mengatur `Approval` gerbang untuk dijalankan sebelum `Deploy` tindakan. Dalam hal ini, `Deploy` tindakan dikatakan tergantung pada `Approval` gerbang.

Untuk mengatur dependensi antara gerbang dan tindakan, konfigurasikan properti Depens on gate atau action. Untuk petunjuk, lihat Menyiapkan dependensi antar tindakan. Instruksi yang direferensikan mengacu pada tindakan alur kerja tetapi berlaku sama untuk gerbang.

Untuk contoh cara mengatur Tergantung pada properti dengan gerbang, lihatContoh: Mengkonfigurasi gerbang “Persetujuan”.

Menentukan versi mayor, minor, atau patch dari gerbang

Secara default, saat Anda menambahkan gerbang ke alur kerja, CodeCatalyst tambahkan versi lengkap ke file definisi alur kerja menggunakan format:

`v`*`major`*`.`*`minor`*`.`*`patch`*

Sebagai contoh:

```
My-Gate:
  Identifier: aws/approval@v1
```

Anda dapat memperpanjang versi sehingga alur kerja menggunakan versi mayor atau minor tertentu dari gerbang. Untuk petunjuk, lihat Menentukan versi mayor, minor, atau patch dari suatu tindakan. Topik yang direferensikan mengacu pada tindakan alur kerja tetapi berlaku sama untuk gerbang.

Memerlukan persetujuan pada alur kerja berjalan

Anda dapat mengonfigurasi alur kerja untuk meminta persetujuan sebelum dapat dilanjutkan. Untuk mencapai ini, Anda harus menambahkan gerbang Persetujuan ke alur kerja. Gerbang Persetujuan mencegah alur kerja berjalan hingga pengguna atau kumpulan pengguna mengirimkan satu atau beberapa persetujuan di konsol. CodeCatalyst Setelah semua persetujuan diberikan, gerbang 'tidak terkunci' dan alur kerja dijalankan diizinkan untuk dilanjutkan.

Gunakan gerbang Persetujuan dalam alur kerja Anda untuk memberi kesempatan kepada tim pengembangan, operasi, dan kepemimpinan Anda untuk meninjau perubahan Anda sebelum diterapkan ke khalayak yang lebih luas.

Bagaimana cara membuka gerbang persetujuan?

Untuk membuka gerbang Persetujuan, semua ketentuan berikut harus dipenuhi:

- Kondisi 1: Jumlah persetujuan yang diperlukan harus diserahkan. Jumlah persetujuan yang diperlukan dapat dikonfigurasi, dan setiap pengguna diizinkan untuk mengirimkan satu persetujuan.
- Kondisi 2: Semua persetujuan harus diserahkan sebelum waktu gerbang habis. Waktu gerbang habis 14 hari setelah diaktifkan. Periode ini tidak dapat dikonfigurasi.
- Kondisi 3: Tidak ada yang harus menolak alur kerja yang dijalankan. Penolakan tunggal akan menyebabkan alur kerja berjalan gagal.
- Kondisi 4: (Hanya berlaku jika Anda menggunakan mode lari yang digantikan.) Lari tidak boleh digantikan oleh lari nanti. Untuk informasi selengkapnya, lihat Bagaimana cara kerja persetujuan alur kerja dengan mode lari antrian, digantikan, dan paralel?.

Jika salah satu kondisi tidak terpenuhi, CodeCatalyst hentikan alur kerja dan atur status run ke Gagal (dalam kasus Kondisi 1 hingga 3) atau Digantikan (dalam kasus Kondisi 4).

Kapan harus menggunakan gerbang “Persetujuan”

Biasanya, Anda akan menggunakan gerbang Persetujuan dalam alur kerja yang menyebarkan aplikasi dan sumber daya lainnya ke server produksi atau lingkungan di mana standar kualitas harus divalidasi. Dengan menempatkan gerbang sebelum penyebaran ke produksi, Anda memberi pengulas kesempatan untuk memvalidasi revisi perangkat lunak baru Anda sebelum tersedia untuk umum.

Siapa yang bisa memberikan persetujuan?

Setiap pengguna yang merupakan anggota proyek Anda dan yang memiliki peran Kontributor atau administrator Proyek dapat memberikan persetujuan. Pengguna dengan peran administrator Space yang termasuk dalam ruang proyek Anda juga dapat memberikan persetujuan.

> **Note**
>
> Pengguna dengan peran Peninjau tidak dapat memberikan persetujuan.

Bagaimana cara memberi tahu pengguna bahwa persetujuan diperlukan?

Untuk memberi tahu pengguna bahwa persetujuan diperlukan, Anda harus:

- CodeCatalyst Kirimi mereka pemberitahuan Slack. Untuk informasi selengkapnya, lihat Mengkonfigurasi pemberitahuan persetujuan.
- Buka halaman di CodeCatalyst konsol tempat tombol Setujui dan Tolak berada, dan tempel URL halaman itu ke aplikasi email atau pesan yang ditujukan kepada pemberi persetujuan. Untuk informasi selengkapnya tentang cara menavigasi ke halaman ini, lihatMenyetujui atau menolak alur kerja.

Dapatkah saya menggunakan gerbang “Persetujuan” untuk mencegah alur kerja dimulai?

Ya, dengan kualifikasi. Untuk informasi selengkapnya, lihat Dapatkah saya menggunakan gerbang untuk mencegah alur kerja berjalan dari awal?.

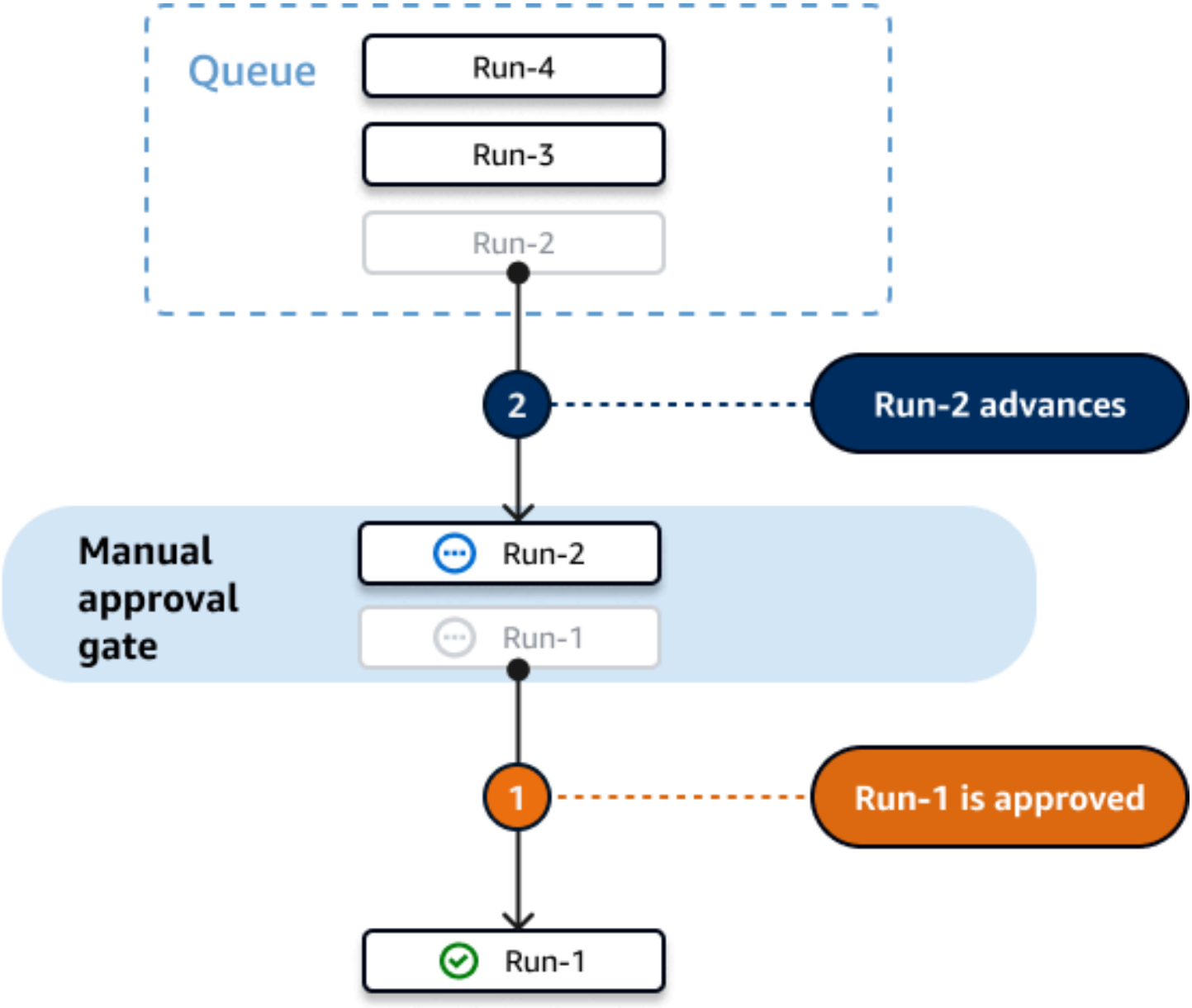
Bagaimana cara kerja persetujuan alur kerja dengan mode lari antrian, digantikan, dan paralel?

Saat menggunakan mode lari antrian, digantikan, atau paralel, gerbang Persetujuan bekerja dengan cara yang mirip dengan tindakan. Kami menyarankan membacaTentang mode lari antrian,Tentang mode lari yang digantikan, Tentang mode parallel run bagian untuk membiasakan diri dengan mode run ini. Setelah Anda memiliki pemahaman dasar tentang mereka, kembali ke bagian ini untuk mencari tahu bagaimana mode run ini bekerja ketika gerbang Persetujuan hadir.

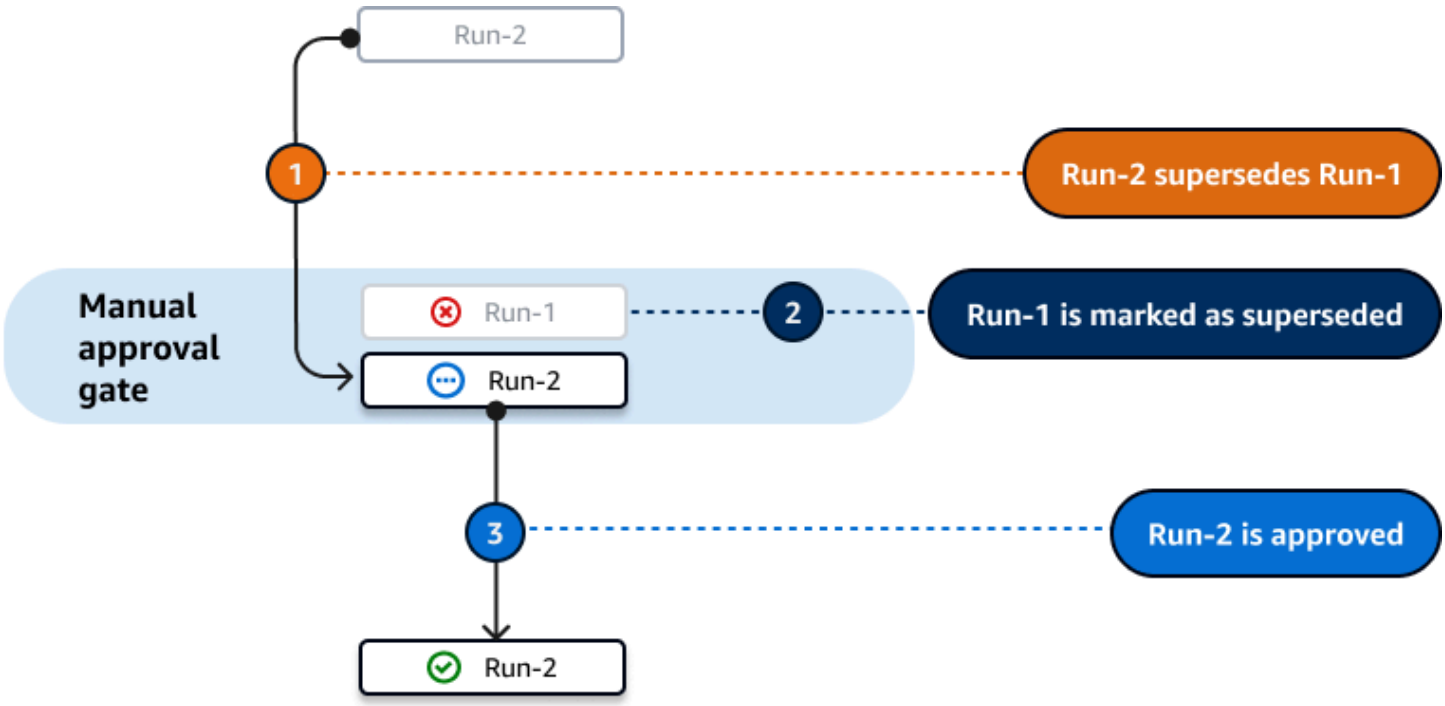
Ketika gerbang Persetujuan hadir, proses diproses sebagai berikut:

- Jika Anda menggunakan mode lari antrian, run akan mengantri di belakang run yang saat ini menunggu persetujuan di gerbang. Ketika gerbang itu menjadi tidak terkunci (yaitu, semua persetujuan telah diberikan), proses berikutnya dalam antrian maju ke gerbang, dan menunggu persetujuan. Proses ini berlanjut dengan proses antrian yang diproses melalui gerbang. one-by-one Figure 1menggambarkan proses ini.
- Jika Anda menggunakan mode run yang digantikan, perilakunya sama dengan mode lari antrian, kecuali bahwa alih-alih menjalankan menumpuk dalam antrian di gerbang, proses yang lebih baru menggantikan (mengambil alih dari) proses sebelumnya. Tidak ada antrian, dan lari apa pun yang saat ini menunggu di gerbang untuk persetujuan akan dibatalkan dan digantikan oleh proses yang lebih baru. Figure 2menggambarkan proses ini.
- Jika Anda menggunakan mode parallel run, run start secara paralel dan tidak ada bentuk antrian. Setiap proses diproses oleh gerbang segera karena tidak ada jalan di depannya. Figure 3menggambarkan proses ini.

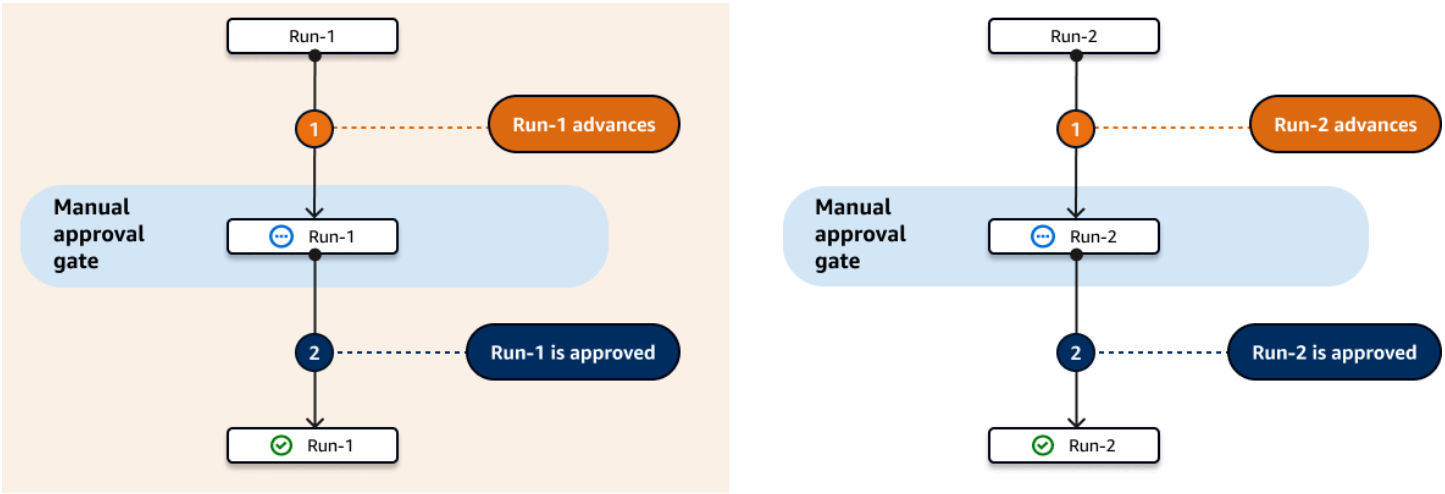
Gambar 1: 'Mode lari antrian' dan gerbang Persetujuan



Gambar 2: 'Mode lari yang digantikan' dan gerbang Persetujuan



Gambar 3: 'Mode lari paralel' dan gerbang Persetujuan



Topik

- Contoh: Mengkonfigurasi gerbang “Persetujuan”
- Menambahkan gerbang “Persetujuan” ke alur kerja
- Mengkonfigurasi pemberitahuan persetujuan
- Menyetujui atau menolak alur kerja
- “Persetujuan” gerbang definisi YAMAL

Contoh: Mengkonfigurasi gerbang “Persetujuan”

Contoh berikut menunjukkan cara menambahkan gerbang Persetujuan yang disebut `Approval_01` antara dua tindakan yang disebut`Staging`, dan`Production`. `Staging`Aksi berjalan pertama, `Approval_01` gerbang kedua, dan `Production` aksi terakhir. `Production`Tindakan hanya berjalan jika `Approval_01` gerbang tidak terkunci. `DependsOn`Properti memastikan bahwa`Staging`,`Approval_01`, dan `Production` fase berjalan dalam urutan berurutan.

Untuk informasi selengkapnya tentang gerbang Persetujuan, lihatMemerlukan persetujuan pada alur kerja berjalan.

```
Actions:
  Staging: # Deploy to a staging server
    Identifier: aws/ecs-deploy@v1
    Configuration:
    ...
  Approval_01:
    Identifier: aws/approval@v1
    DependsOn:
      - Staging
    Configuration:
      ApprovalsRequired: 2
  Production: # Deploy to a production server
    Identifier: aws/ecs-deploy@v1
    DependsOn:
      - Approval_01
    Configuration:
    ...
```

Menambahkan gerbang “Persetujuan” ke alur kerja

Untuk mengonfigurasi alur kerja agar memerlukan persetujuan, Anda harus menambahkan gerbang Persetujuan ke alur kerja. Gunakan petunjuk berikut untuk menambahkan gerbang Persetujuan ke alur kerja Anda.

Untuk informasi lebih lanjut tentang gerbang ini, lihatMemerlukan persetujuan pada alur kerja berjalan.

Visual

Untuk menambahkan gerbang “Persetujuan” ke alur kerja (editor visual)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Di kiri atas, pilih Gates.
7. Di katalog Gates, di Approval, pilih tanda plus (+).
8. Pilih Input, dan di bidang Tergantung pada, lakukan hal berikut.

   Tentukan tindakan, grup tindakan, atau gerbang yang harus berjalan dengan sukses agar gerbang ini berjalan. Secara default, saat Anda menambahkan gerbang ke alur kerja, gerbang diatur untuk bergantung pada tindakan terakhir dalam alur kerja Anda. Jika Anda menghapus properti ini, gerbang tidak akan tergantung pada apa pun, dan akan berjalan terlebih dahulu, sebelum tindakan lain.

   > **Note**
   >
   > Gerbang harus dikonfigurasi untuk dijalankan sebelum atau sesudah tindakan, grup tindakan, atau gerbang. Itu tidak dapat diatur untuk berjalan secara paralel dengan tindakan lain, kelompok tindakan, dan gerbang.

   Untuk informasi selengkapnya tentang Tergantung pada fungsionalitas, lihatMenyiapkan dependensi antara gerbang dan tindakan.
9. Pilih tab Konfigurasi.
10. Di bidang nama Gerbang, lakukan hal berikut.

    Tentukan nama yang ingin Anda berikan gerbang. Semua nama gerbang harus unik dalam alur kerja. Nama gerbang terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama gerbang.

11. (Opsional) Di bidang Jumlah persetujuan, lakukan hal berikut.

    Tentukan jumlah minimum persetujuan yang diperlukan untuk membuka gerbang Persetujuan. Minimal adalah1. Maksimal adalah2. Jika dihilangkan, default-nya adalah 1.

    > **Note**
    >
    > Jika Anda ingin menghilangkan `ApprovalsRequired` properti, hapus `Configuration` bagian gerbang dari file definisi alur kerja.

12. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
13. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menambahkan gerbang “Persetujuan” ke alur kerja (editor YAMG)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMG.
7. Tambahkan `Approval` bagian dan properti yang mendasari menggunakan contoh berikut sebagai panduan. Untuk informasi lebih lanjut, lihat “Persetujuan” gerbang definisi YAMAL diAlur kerja definisi YAMAL.

```
Actions:
  MyApproval_01:
    Identifier: aws/approval@v1
    DependsOn:
      - PreviousAction
    Configuration:
      ApprovalsRequired: 2
```

Untuk contoh lain, lihatContoh: Mengkonfigurasi gerbang “Persetujuan”.

8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Mengkonfigurasi pemberitahuan persetujuan

Anda dapat CodeCatalyst mengirim pemberitahuan ke saluran Slack yang memberi tahu pengguna bahwa alur kerja berjalan memerlukan persetujuan. Pengguna melihat notifikasi dan mengklik tautan di dalamnya. Tautan membawa mereka ke halaman CodeCatalyst persetujuan di mana mereka dapat menyetujui atau menolak alur kerja.

Anda juga dapat mengonfigurasi notifikasi untuk memberi tahu pengguna bahwa alur kerja telah disetujui, ditolak, atau bahwa permintaan persetujuan telah kedaluwarsa.

Gunakan petunjuk berikut untuk mengatur notifikasi Slack.

Sebelum Anda mulai

Pastikan Anda telah menambahkan gerbang Persetujuan ke alur kerja Anda. Untuk informasi selengkapnya, lihat Menambahkan gerbang “Persetujuan” ke alur kerja.

Untuk mengirim pemberitahuan persetujuan alur kerja ke saluran Slack

1. Konfigurasikan CodeCatalyst dengan Slack. Untuk informasi selengkapnya, lihat Memulai dengan notifikasi Slack.
2. Dalam CodeCatalyst proyek yang berisi alur kerja yang memerlukan persetujuan, aktifkan notifikasi, jika belum diaktifkan. Untuk mengaktifkan notifikasi:
    a. Arahkan ke proyek Anda dan di panel navigasi, pilih Pengaturan proyek.
    b. Di bagian atas, pilih Notifikasi.
    c. Di acara Pemberitahuan, pilih Edit notifikasi.
    d. Aktifkan persetujuan alur kerja tertunda dan pilih saluran Slack tempat CodeCatalyst akan mengirim notifikasi.
    e. (Opsional) Aktifkan notifikasi tambahan untuk memberi tahu orang tentang persetujuan yang disetujui, ditolak, dan kedaluwarsa. Anda dapat mengaktifkan Workflow run approved, Workflow run ditolak, Persetujuan alur kerja digantikan, dan persetujuan alur kerja habis. Di sebelah setiap notifikasi, pilih saluran Slack tempat CodeCatalyst akan mengirim notifikasi.
    f. Pilih Simpan.

Menyetujui atau menolak alur kerja

Alur kerja yang menyertakan gerbang Persetujuan harus disetujui atau ditolak. Pengguna dapat memberikan persetujuan atau penolakan mereka mulai dari:

- CodeCatalyst konsol
- tautan yang disediakan oleh anggota tim
- pemberitahuan Slack otomatis

Setelah pengguna memberikan persetujuan atau penolakan mereka, keputusan ini tidak dapat dibatalkan.

> **Note**
>
> Hanya pengguna tertentu yang dapat menyetujui atau menolak alur kerja yang dijalankan. Untuk informasi selengkapnya, lihat Siapa yang bisa memberikan persetujuan?.

Sebelum kamu memulai

Pastikan Anda telah menambahkan gerbang Persetujuan ke alur kerja Anda. Untuk informasi selengkapnya, lihat Menambahkan gerbang “Persetujuan” ke alur kerja.

Untuk menyetujui atau menolak alur kerja yang dijalankan mulai dari konsol CodeCatalyst

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Dalam diagram alur kerja, pilih kotak yang mewakili gerbang Persetujuan.

   Panel samping muncul.

> **Note**
>
> Pada titik ini, Anda dapat mengirim URL halaman ini ke pemberi persetujuan lain jika Anda mau.

6. Di bawah keputusan Tinjau, pilih Menyetujui atau Tolak.
7. (Opsional) Di Komentar - opsional, masukkan komentar yang menunjukkan mengapa Anda menyetujui atau menolak alur kerja berjalan.
8. Pilih Kirim.

Untuk menyetujui atau menolak alur kerja yang dijalankan mulai dari tautan yang disediakan oleh anggota tim

1. Pilih tautan yang dikirimkan kepada Anda oleh anggota tim Anda. (Anda dapat meminta anggota tim Anda membaca prosedur sebelumnya untuk mendapatkan tautan.)
2. Masuk ke CodeCatalyst, jika ditanya.

   Anda diarahkan ke halaman persetujuan jalankan alur kerja.
3. Di bawah keputusan Tinjau, pilih Menyetujui atau Tolak.
4. (Opsional) Di Komentar - opsional, masukkan komentar yang menunjukkan mengapa Anda menyetujui atau menolak alur kerja berjalan.
5. Pilih Kirim.

Untuk menyetujui atau menolak alur kerja yang dimulai dari pemberitahuan Slack otomatis

1. Pastikan notifikasi Slack sudah diatur. Lihat Mengkonfigurasi pemberitahuan persetujuan.
2. Di Slack, di saluran tempat pemberitahuan persetujuan dikirim, pilih tautan dalam pemberitahuan persetujuan.
3. Masuk ke CodeCatalyst, jika ditanya.

   Anda diarahkan ke halaman jalankan alur kerja.
4. Dalam diagram alur kerja, pilih gerbang persetujuan.
5. Di bawah keputusan Tinjau, pilih Menyetujui atau Tolak.

6. (Opsional) Di Komentar - opsional, masukkan komentar yang menunjukkan mengapa Anda menyetujui atau menolak alur kerja berjalan.
7. Pilih Kirim.

“Persetujuan” gerbang definisi YAMAL

Berikut ini adalah definisi YAMAL dari gerbang Persetujuan. Untuk mempelajari cara menggunakan gerbang ini, lihatMemerlukan persetujuan pada alur kerja berjalan.

Definisi tindakan ini ada sebagai bagian dalam file definisi alur kerja yang lebih luas. Untuk informasi lebih lanjut tentang file ini, lihatAlur kerja definisi YAMAL.

> **Note**
>
> Sebagian besar properti YAMAL yang mengikuti memiliki elemen UI yang sesuai di editor visual. Untuk mencari elemen UI, gunakan Ctrl+F. Elemen akan terdaftar dengan properti YAMLnya yang terkait.

```
# The workflow definition starts here.
# See Properti tingkat atas for details.

Name: MyWorkflow
SchemaVersion: 1.0
Actions:

# The "Approval" gate definition starts here.
  Approval:
    Identifier: aws/approval@v1
    DependsOn:
      - another-action
    Configuration:
      ApprovalsRequired: number
```

Approval

(Diperlukan)

Tentukan nama yang ingin Anda berikan gerbang. Semua nama gerbang harus unik dalam alur kerja. Nama gerbang terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah

(_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama gerbang.

Default: `Approval_nn`.

UI yang sesuai: Tab konfigurasi/nama Gerbang

Identifier

(*`Approval`*/Identifier)

(Diperlukan)

Mengidentifikasi gerbang. Gerbang Persetujuan mendukung versi`1.0.0`. Jangan mengubah properti ini kecuali Anda ingin mempersingkat versinya. Untuk informasi selengkapnya, lihat Menentukan versi mayor, minor, atau patch dari suatu tindakan.

Default: `aws/approval@v1`.

UI yang sesuai: Diagram alur Approval kerja/_nn/ aws/persetujuan @v1 label

DependsOn

(*`Approval`*/DependsOn)

(Opsional)

Tentukan tindakan, grup tindakan, atau gerbang yang harus berjalan dengan sukses agar gerbang ini berjalan. Secara default, saat Anda menambahkan gerbang ke alur kerja, gerbang diatur untuk bergantung pada tindakan terakhir dalam alur kerja Anda. Jika Anda menghapus properti ini, gerbang tidak akan tergantung pada apa pun, dan akan berjalan terlebih dahulu, sebelum tindakan lain.

> **Note**
>
> Gerbang harus dikonfigurasi untuk dijalankan sebelum atau sesudah tindakan, grup tindakan, atau gerbang. Itu tidak dapat diatur untuk berjalan secara paralel dengan tindakan lain, kelompok tindakan, dan gerbang.

Untuk informasi selengkapnya tentang Tergantung pada fungsionalitas, lihatMenyiapkan dependensi antara gerbang dan tindakan.

UI yang sesuai: Tab masukan/Tergantung pada

Configuration

(*Approval*/Configuration)

(Opsional)

Bagian di mana Anda dapat menentukan properti konfigurasi gerbang.

UI yang sesuai: Tab konfigurasi

ApprovalsRequired

(*Approval*/Configuration/ApprovalsRequired)

(Opsional)

Tentukan jumlah minimum persetujuan yang diperlukan untuk membuka gerbang Persetujuan. Minimal adalah1. Maksimal adalah2. Jika dihilangkan, default-nya adalah 1.

> **Note**
>
> Jika Anda ingin menghilangkan `ApprovalsRequired` properti, hapus `Configuration` bagian gerbang dari file definisi alur kerja.

UI yang sesuai: Tab konfigurasi/Jumlah persetujuan

Mengonfigurasi perilaku antrian run

Secara default, ketika beberapa alur kerja berjalan terjadi pada saat yang sama, CodeCatalyst mengantri mereka, dan memprosesnya satu per satu, dalam urutan bahwa mereka dimulai. Anda dapat mengubah perilaku default ini dengan menentukan mode run. Ada beberapa mode lari:

- (Default) Mode lari antrian — CodeCatalyst proses berjalan satu per satu
- Mode lari yang digantikan - CodeCatalyst proses berjalan satu per satu, dengan proses yang lebih baru menyalip yang lebih lama
- Parallel run mode — CodeCatalyst proses berjalan secara paralel

Topik

Tentang mode lari antrian

Dalam mode lari antrian, proses berjalan terjadi secara seri, dengan menunggu berjalan membentuk antrian.

Antrian terbentuk di titik masuk ke tindakan dan grup tindakan, sehingga Anda dapat memiliki beberapa antrian dalam alur kerja yang sama (lihat). Figure 1 Ketika lari antrian memasuki suatu tindakan, tindakan terkunci dan tidak ada jalan lain yang bisa masuk. Ketika run selesai dan keluar dari aksi, aksi menjadi tidak terkunci dan siap untuk menjalankan berikutnya.

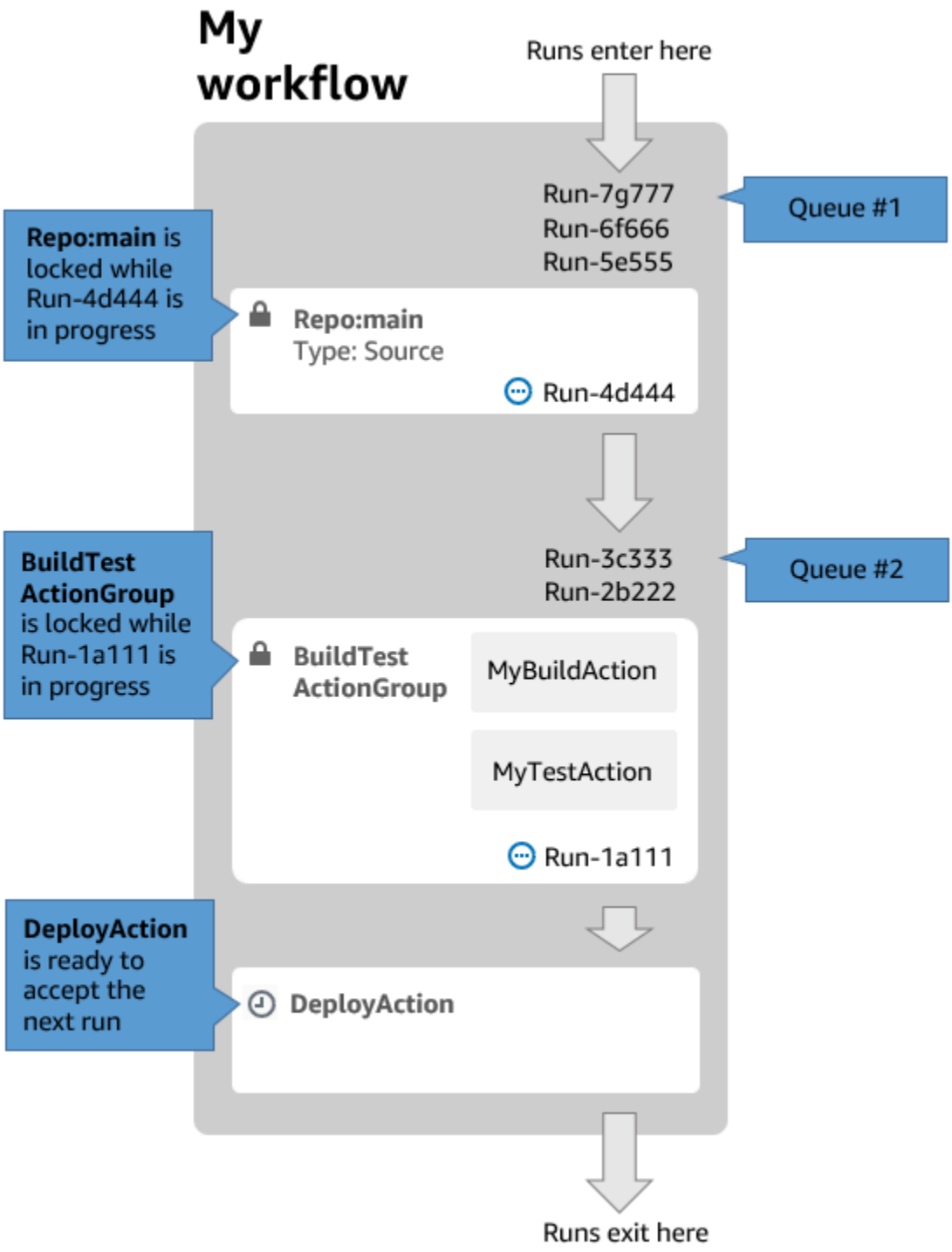
Figure 1mengilustrasikan alur kerja yang dikonfigurasi dalam mode lari antrian. Ini menunjukkan:

- Tujuh berjalan bekerja dengan cara mereka melalui alur kerja.
- Dua antrian: satu di luar entri ke sumber input (Repo: main), dan satu di luar entri ke tindakan. BuildTestActionGroup
- Dua blok terkunci: sumber input (Repo: Main) dan. BuildTestActionGroup

Berikut adalah bagaimana hal-hal akan terjadi saat alur kerja menjalankan pemrosesan selesai:

- Ketika Run-4D444 selesai mengkloning repositori sumber, itu akan keluar dari sumber input dan bergabung dengan antrian di belakang Run-3C333. Kemudian, Run-5E555 akan masuk ke sumber input.
- Ketika Run-1A111 selesai membangun dan menguji, itu akan keluar dari tindakan dan memasuki tindakan. BuildTestActionGroupDeployAction Kemudian, Run-2B222 akan memasuki aksi. BuildTestActionGroup

Gambar 1: Alur kerja yang dikonfigurasi dalam 'mode lari antrean'



Gunakan mode lari antrian jika:

- Anda ingin menjaga one-to-one hubungan antara fitur dan proses — fitur ini dapat dikelompokkan saat menggunakan mode yang diganti. Misalnya, saat Anda menggabungkan fitur 1 di komit 1, jalankan 1 dimulai, dan saat Anda menggabungkan fitur 2 di komit 2, jalankan 2 dimulai, dan

seterusnya. Jika Anda menggunakan mode yang digantikan alih-alih mode antrian, fitur Anda (dan komit) akan dikelompokkan bersama dalam proses yang menggantikan yang lain.

- Anda ingin menghindari kondisi balapan dan masalah tak terduga yang mungkin terjadi saat menggunakan mode paralel. Misalnya, jika dua pengembang perangkat lunak, Wang dan Saanvi, memulai alur kerja berjalan pada waktu yang kira-kira bersamaan untuk menyebarkan ke cluster Amazon ECS, proses Wang mungkin memulai tes integrasi pada cluster sementara menjalankan Saanvi menyebarkan kode aplikasi baru ke cluster, menyebabkan tes Wang gagal atau menguji kode yang salah. Sebagai contoh lain, Anda mungkin memiliki target yang tidak memiliki mekanisme penguncian, dalam hal ini kedua proses dapat menimpa perubahan satu sama lain dengan cara yang tidak terduga.
- Anda ingin membatasi beban pada sumber daya komputasi yang CodeCatalyst digunakan untuk memproses proses Anda. Misalnya, jika Anda memiliki tiga tindakan dalam alur kerja Anda, Anda dapat memiliki maksimum tiga proses yang terjadi pada saat yang sama. Memaksakan batasan pada jumlah run yang dapat terjadi sekaligus membuat throughput run lebih dapat diprediksi.
- Anda ingin membatasi jumlah permintaan yang dibuat ke layanan pihak ketiga berdasarkan alur kerja. Misalnya, alur kerja Anda mungkin memiliki tindakan build yang menyertakan instruksi untuk menarik gambar dari Docker Hub. Docker Hub membatasi jumlah permintaan tarik yang dapat Anda buat hingga jumlah tertentu per jam per akun, dan Anda akan dikunci jika melampaui batas. Menggunakan mode lari antrian untuk memperlambat throughput run Anda akan memiliki efek menghasilkan lebih sedikit permintaan ke Docker Hub per jam, sehingga membatasi potensi penguncian dan mengakibatkan kegagalan build dan run.

Ukuran antrian maksimal: 50

Catatan tentang ukuran antrian maksimum:

- Ukuran antrian maksimum mengacu pada jumlah maksimum proses yang diizinkan di semua antrian dalam alur kerja.
- Jika antrian menjadi lebih panjang dari 50 run, maka CodeCatalyst turunkan lari ke-51 dan selanjutnya.

Perilaku kegagalan:

Jika proses menjadi tidak responsif saat sedang diproses oleh suatu tindakan, maka proses di belakangnya ditahan dalam antrian hingga waktu tindakan habis. Waktu tindakan habis setelah satu jam.

Jika run gagal di dalam suatu tindakan, maka lari antrian pertama di belakangnya diizinkan untuk melanjutkan.

Tentang mode lari yang digantikan

Mode lari yang diganti sama dengan mode lari antrian kecuali bahwa:

- Jika proses antrian mengejar lari lain dalam antrian, proses selanjutnya menggantikan (mengambil alih dari) proses sebelumnya, dan proses sebelumnya dibatalkan dan ditandai sebagai 'digantikan'.
- Sebagai hasil dari perilaku yang dijelaskan dalam bullet pertama, antrian hanya dapat menyertakan satu run ketika mode run digantikan digunakan.

Menggunakan alur kerja Figure 1 sebagai panduan, menerapkan mode run yang digantikan ke alur kerja ini akan menghasilkan hal berikut:

- Run-7G777 akan menggantikan dua run lainnya dalam antreannya, dan akan menjadi satu-satunya run yang tersisa di Antrian #1. Run-6F666 dan Run-5E555 akan dibatalkan.
- Run-3C333 akan menggantikan Run-2B222 dan menjadi satu-satunya run yang tersisa di Antrian #2. Run-2b222 akan dibatalkan.

Gunakan mode lari yang digantikan jika Anda ingin:

- throughput yang lebih baik daripada dengan mode antrian
- bahkan lebih sedikit permintaan ke layanan pihak ketiga dibandingkan dengan mode antrian; ini menguntungkan jika layanan pihak ketiga memiliki batas tarif, seperti Docker Hub

Tentang mode parallel run

Dalam mode parallel run, run tidak tergantung satu sama lain dan jangan menunggu proses lain selesai sebelum memulai. Tidak ada antrian, dan run throughput hanya dibatasi oleh seberapa cepat tindakan di dalam alur kerja selesai.

Gunakan mode parallel run di lingkungan pengembangan di mana setiap pengguna memiliki cabang fitur mereka sendiri dan menyebarkan ke target yang tidak dibagikan oleh pengguna lain.

> **Important**
>
> Jika Anda memiliki target bersama yang dapat diterapkan oleh beberapa pengguna, seperti fungsi Lambda di lingkungan produksi, jangan gunakan mode paralel, karena kondisi balapan dapat terjadi. Kondisi balapan terjadi ketika alur kerja paralel berjalan mencoba mengubah sumber daya bersama pada saat yang sama, yang mengarah ke hasil yang tidak dapat diprediksi.

Jumlah maksimum parallel run: 1000 per CodeCatalyst spasi

Mengkonfigurasi mode lari

Anda dapat mengatur mode lari ke antrian, digantikan, atau paralel. Defaultnya antri.

Saat Anda mengubah mode lari dari antrian atau digantikan menjadi paralel CodeCatalyst , membatalkan proses yang diantrian, dan memungkinkan proses yang saat ini sedang diproses oleh tindakan selesai sebelum membatalkannya.

Ketika Anda mengubah mode run dari parallel ke antrian atau digantikan, memungkinkan CodeCatalyst semua berjalan paralel yang berjalan saat ini selesai. Setiap proses yang Anda mulai setelah mengubah mode run menjadi antrian atau digantikan menggunakan mode baru.

Visual

Untuk mengubah mode lari menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Di kanan atas, pilih properti Workflow.
7. Perluas Lanjutan, dan di bawah mode Jalankan, pilih salah satu dari berikut ini:
   a. Antrian — lihat Tentang mode lari antrian
   b. Digantikan — lihat Tentang mode lari yang digantikan

c. Paralel - lihat Tentang mode parallel run

8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk mengubah mode lari menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Tambahkan `RunMode` properti, seperti ini:

```
Name: Workflow_6d39
SchemaVersion: "1.0"
RunMode: QUEUED|SUPERSEDED|PARALLEL
```

Untuk informasi lebih lanjut, lihat deskripsi `RunMode` properti di Properti tingkat atas bagianAlur kerja definisi YAMAL.

8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Caching file di antara alur kerja berjalan

Saat caching file diaktifkan, tindakan build dan test menyimpan file on-disk ke cache dan mengembalikannya dari cache tersebut dalam alur kerja berikutnya. Caching mengurangi latensi yang disebabkan oleh membangun atau mengunduh dependensi yang tidak berubah di antara proses. CodeCatalyst juga mendukung cache fallback, yang dapat digunakan untuk mengembalikan cache sebagian yang berisi beberapa dependensi yang diperlukan. Ini membantu mengurangi dampak latensi dari kehilangan cache.

> **Note**
>
> Caching file hanya tersedia dengan tindakan CodeCatalyst pembuatan dan pengujian Amazon, dan hanya jika file tersebut dikonfigurasi untuk menggunakan jenis komputasi EC2.

Topik

Tentang file caching

File caching memungkinkan Anda untuk mengatur data Anda ke dalam beberapa cache, yang masing-masing direferensikan di bawah properti. `FileCaching` Setiap cache menyimpan direktori yang ditentukan oleh jalur yang diberikan. Direktori yang ditentukan akan dipulihkan dalam alur kerja masa depan berjalan. Berikut ini adalah contoh cuplikan YAMAL untuk caching dengan beberapa cache bernama dan. `cacheKey1` `cacheKey2`

```
Actions:
  BuildMyNpmApp:
    Identifier: aws/build@v1
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      Steps:
        - Run: npm install
        - Run: npm run test
    Caching:
      FileCaching:
        cacheKey1:
          Path: file1.txt
          RestoreKeys:
             - restoreKey1
        cacheKey2:
          Path: /root/repository
          RestoreKeys:
             - restoreKey2
```

```
            - restoreKey3
```

> **Note**
>
> CodeCatalyst menggunakan caching berlapis-lapis, yang terdiri dari cache lokal dan cache jarak jauh. Ketika armada yang disediakan atau mesin sesuai permintaan mengalami kehilangan cache pada cache lokal, dependensi akan dipulihkan dari cache jarak jauh. Akibatnya, beberapa tindakan berjalan mungkin mengalami latensi dari mengunduh cache jarak jauh.

CodeCatalyst menerapkan pembatasan akses cache untuk memastikan bahwa tindakan dalam satu alur kerja tidak dapat mengubah cache dari alur kerja yang berbeda. Ini melindungi setiap alur kerja dari orang lain yang mungkin mendorong data yang salah yang memengaruhi build atau penerapan. Pembatasan diberlakukan dengan cakupan cache yang mengisolasi cache ke setiap alur kerja dan pasangan cabang. Misalnya, `workflow-A` di cabang `feature-A` memiliki cache file yang berbeda dari `workflow-A` di cabang `feature-B` saudara.

Kesalahan cache terjadi ketika alur kerja mencari cache file tertentu dan tidak dapat menemukannya. Hal ini dapat terjadi karena beberapa alasan, seperti ketika cabang baru dibuat atau ketika cache baru direferensikan dan belum dibuat. Ini juga dapat terjadi ketika cache kedaluwarsa, yang secara default terjadi 14 hari setelah terakhir digunakan. Untuk mengurangi kesalahan cache dan meningkatkan tingkat klik cache, CodeCatalyst mendukung cache fallback. Cache fallback adalah cache alternatif dan memberikan kesempatan untuk memulihkan cache sebagian, yang bisa menjadi versi cache yang lebih lama. Cache dipulihkan dengan terlebih dahulu mencari kecocokan di bawah `FileCaching` untuk nama properti, dan jika tidak ditemukan, evaluasi`RestoreKeys`. Jika ada cache yang hilang untuk nama properti dan semua`RestoreKeys`, alur kerja akan terus berjalan, karena caching adalah upaya terbaik dan tidak dijamin.

Membuat cache

Anda dapat menggunakan petunjuk berikut untuk menambahkan cache ke alur kerja Anda.

Visual

Untuk menambahkan cache menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.

3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Dalam diagram alur kerja, pilih tindakan di mana Anda ingin menambahkan cache Anda.
8. Pilih Konfigurasi.
9. Di bawah File caching - opsional, pilih Tambahkan cache dan masukkan informasi ke dalam bidang, sebagai berikut:

   Kunci

   Tentukan nama nama properti cache utama Anda. Nama properti cache harus unik dalam alur kerja Anda. Setiap tindakan dapat memiliki hingga lima entri. `FileCaching`

   Jalan

   Tentukan jalur terkait untuk cache Anda.

   Kembalikan kunci - opsional

   Tentukan kunci pemulihan yang akan digunakan sebagai fallback ketika properti cache utama tidak dapat ditemukan. Kembalikan nama kunci harus unik dalam alur kerja Anda. Setiap cache dapat memiliki hingga lima entri. `RestoreKeys`
10. (Opsional) Pilih Validasi untuk memvalidasi kode YAMM alur kerja sebelum melakukan.
11. Pilih Komit, masukkan pesan komit, lalu pilih Komit lagi.

YAML

Untuk menambahkan cache menggunakan editor YAMM

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.

5. Pilih Edit.
6. Pilih YAMAL.
7. Dalam tindakan alur kerja, tambahkan kode yang mirip dengan berikut ini:

```
action-name:
  Configuration:
    Steps: ...
  Caching:
    FileCaching:
      key-name:
        Path: file-path
        # # Specify any additional fallback caches
        # RestoreKeys:
        #  - restore-key
```

8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMM alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Kendala caching file

Berikut ini adalah kendala untuk nama properti dan: `RestoreKeys`

- Nama harus unik dalam alur kerja.
- Nama terbatas pada karakter alfanumerik (A-Z, a-z, 0-9), tanda hubung (-), dan garis bawah (_).
- Nama dapat memiliki hingga 180 karakter.
- Setiap tindakan dapat memiliki hingga lima cache. `FileCaching`
- Setiap cache dapat memiliki hingga lima entri. `RestoreKeys`

Berikut ini adalah kendala untuk jalur:

- Tanda bintang (*) tidak diperbolehkan.
- Jalur dapat memiliki hingga 255 karakter.

Melihat alur kerja menjalankan status dan detail

Anda dapat melihat status dan detail dari satu alur kerja yang dijalankan, atau beberapa proses secara bersamaan.

Untuk daftar kemungkinan status jalankan lihatAlur kerja menjalankan status.

> **Note**
>
> Anda juga dapat melihat status alur kerja, yang berbeda dari status menjalankan alur kerja. Untuk informasi selengkapnya, lihat Melihat status alur kerja.

Topik

Melihat status dan detail dari satu run

Anda mungkin ingin melihat status dan detail alur kerja tunggal untuk memeriksa apakah itu berhasil, untuk melihat pada jam berapa selesai, atau untuk melihat siapa atau apa yang memulainya.

Untuk melihat status dan detail dari satu run

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Di bawah nama alur kerja, pilih Runs.
6. Di Riwayat Jalankan, di kolom Run ID, pilih run. Misalnya, `Run-95a4d`.
7. Di bawah nama run, lakukan salah satu hal berikut:

    - Visual untuk melihat diagram alur kerja yang menunjukkan tindakan alur kerja dan statusnya (lihatAlur kerja menjalankan status). Tampilan ini juga menunjukkan repositori sumber dan cabang yang digunakan selama menjalankan.

      Dalam diagram alur kerja, pilih tindakan untuk melihat detail seperti log, laporan, dan output yang dihasilkan oleh tindakan selama menjalankan. Informasi yang ditampilkan tergantung

pada jenis tindakan yang dipilih. Untuk informasi selengkapnya tentang melihat log build atau deploy, lihat Melihat hasil tindakan build atauMelihat log penerapan.

- YAMM untuk melihat file definisi alur kerja yang digunakan untuk menjalankan.
- Artefak untuk melihat artefak yang dihasilkan oleh alur kerja dijalankan. Untuk informasi lebih lanjut tentang artifact, lihat Berbagi data antar tindakan dalam alur kerja menggunakan artefak.
- Laporan untuk melihat laporan pengujian dan jenis laporan lain yang dihasilkan oleh alur kerja yang dijalankan. Untuk informasi lebih lanjut tentang laporan, lihatJenis laporan kualitas.
- Variabel untuk melihat variabel output yang dihasilkan oleh alur kerja yang dijalankan. Untuk informasi lebih lanjut tentang variabel, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

> **Note**
>
> Jika alur kerja induk run dihapus, pesan yang menunjukkan fakta ini akan muncul di bagian atas halaman rincian proses.

Melihat status dan detail semua proses dalam proyek Anda

Anda mungkin ingin melihat status dan detail dari semua alur kerja yang berjalan dalam proyek Anda memahami berapa banyak aktivitas alur kerja yang terjadi dalam proyek Anda, dan mempelajari tentang kesehatan keseluruhan alur kerja Anda.

Untuk melihat status dan detail semua proses dalam proyek Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Di bawah Alur kerja, pilih Runs.

   Semua proses, untuk semua alur kerja, di semua cabang, di semua repositori dalam proyek Anda, ditampilkan.

   Halaman ini mencakup kolom berikut:

   - Run ID — Pengidentifikasi unik dari proses. Pilih tautan run ID untuk melihat informasi terperinci tentang proses.

- Status - Status pemrosesan alur kerja yang dijalankan. Untuk informasi selengkapnya tentang status run, lihatAlur kerja menjalankan status.
- Trigger — Orang, commit, pull request (PR), atau jadwal yang memulai alur kerja berjalan. Untuk informasi selengkapnya, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.
- Alur kerja - Nama alur kerja tempat proses dimulai, dan repositori sumber dan cabang tempat file definisi alur kerja berada. Anda mungkin perlu memperluas lebar kolom untuk melihat informasi ini.

  > **Note**
  >
  > Jika kolom ini disetel ke Tidak tersedia, biasanya karena alur kerja terkait telah dihapus atau dipindahkan.

- Waktu mulai — Waktu ketika alur kerja berjalan dimulai.
- Durasi — Berapa lama alur kerja berjalan untuk diproses. Durasi yang sangat panjang atau sangat singkat mungkin mengindikasikan masalah.
- Waktu akhir - Waktu ketika alur kerja berjalan berakhir.

Melihat status dan detail semua proses alur kerja tertentu

Anda mungkin ingin melihat status dan detail semua proses yang terkait dengan alur kerja tertentu untuk melihat apakah ada proses yang membuat hambatan dalam alur kerja, atau untuk melihat proses mana yang sedang berlangsung atau telah selesai.

Untuk melihat status dan detail semua proses alur kerja tertentu

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Di bawah nama alur kerja, pilih Runs.

   Proses yang terkait dengan alur kerja yang dipilih muncul.

Halaman ini dibagi menjadi dua bagian:

- Aktif berjalan - Menampilkan berjalan yang sedang berlangsung. Lari ini akan berada di salah satu status berikut: Sedang berlangsung.
- Run history - Menampilkan run yang telah selesai (yaitu, tidak sedang berlangsung).

  Untuk informasi selengkapnya tentang status run, lihatAlur kerja menjalankan status.

Melihat alur kerja dalam diagram alur kerja

Anda dapat melihat status semua proses alur kerja saat mereka maju bersama melalui alur kerja. Jalankan ditampilkan dalam diagram alur kerja (sebagai lawan dari dalam tampilan daftar). Ini memberi Anda representasi visual dari proses mana yang sedang diproses oleh tindakan mana, dan lari mana yang menunggu dalam antrian.

Untuk melihat status beberapa proses saat mereka maju bersama melalui alur kerja

> **Note**
>
> Prosedur ini hanya berlaku jika alur kerja Anda menggunakan mode lari antrian atau digantikan. Untuk informasi selengkapnya, lihat Mengonfigurasi perilaku antrian run.

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja yang berisi proses yang ingin Anda lihat. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.

   > **Note**
   >
   > Pastikan Anda melihat halaman alur kerja dan bukan halaman run.

5. Pilih tab Status terbaru di kiri atas.

   Diagram alur kerja muncul.

6. Tinjau diagram alur kerja. Diagram menunjukkan semua proses yang sedang berlangsung dalam alur kerja, dan proses terbaru yang telah selesai. Lebih khusus lagi:

   - Lari yang muncul di bagian atas, sebelum Sumber, antri dan menunggu untuk memulai.
   - Jalankan yang muncul di antara tindakan diantrian dan menunggu untuk diproses oleh tindakan berikutnya.
   - Runs yang muncul dalam suatu tindakan adalah 1. saat ini sedang diproses oleh tindakan, 2. telah selesai diproses oleh tindakan, atau 3. tidak diproses oleh tindakan (biasanya karena tindakan sebelumnya gagal).

Mengkonfigurasi tindakan yang dilakukan alur kerja

Tindakan adalah blok bangunan utama alur kerja, dan mendefinisikan unit logis kerja, atau tugas, untuk dilakukan selama alur kerja dijalankan. Biasanya, alur kerja mencakup beberapa tindakan yang berjalan secara berurutan atau paralel tergantung pada cara Anda mengonfigurasinya.

Topik

Jenis tindakan

Dalam CodeCatalyst alur kerja Amazon, Anda dapat menggunakan jenis tindakan berikut.

Jenis tindakan

- CodeCatalyst tindakan
- CodeCatalyst Tindakan Lab
- GitHub Tindakan
- Tindakan pihak ketiga

CodeCatalyst tindakan

CodeCatalyst Tindakan adalah tindakan yang ditulis, dipelihara, dan didukung penuh oleh tim CodeCatalyst pengembangan.

Ada CodeCatalyst tindakan untuk membangun, menguji, dan menyebarkan aplikasi, serta untuk melakukan tugas lain-lain, seperti memanggil fungsi. AWS Lambda

CodeCatalyst Tindakan berikut tersedia:

- Membangun

  Tindakan ini membangun artefak Anda dan menjalankan pengujian unit Anda dalam wadah Docker. Untuk informasi selengkapnya, lihat Menambahkan aksi build.
- Uji

  Tindakan ini menjalankan integrasi dan pengujian sistem terhadap aplikasi atau artefak Anda. Untuk informasi selengkapnya, lihat Menambahkan tindakan pengujian.
- Amazon S3 mempublikasikan

  Tindakan ini menyalin artefak aplikasi Anda ke bucket Amazon S3. Untuk informasi selengkapnya, lihat Menerbitkan file ke Amazon S3 dengan alur kerja.
- AWS CDK bootstrap

  Tindakan ini menyediakan sumber daya yang AWS CDK dibutuhkan untuk menerapkan aplikasi CDK Anda. Untuk informasi selengkapnya, lihat Bootstrapping AWS CDK aplikasi dengan alur kerja.
- AWS CDK menyebarkan

  Tindakan ini mensintesis dan menyebarkan aplikasi AWS Cloud Development Kit (AWS CDK) . Untuk informasi selengkapnya, lihat Menerapkan AWS Cloud Development Kit (AWS CDK) aplikasi dengan alur kerja.
- AWS Lambda memohon

Tindakan ini memanggil AWS Lambda fungsi. Untuk informasi selengkapnya, lihat Memanggil AWS Lambda fungsi menggunakan alur kerja.

- GitHub Tindakan

  Tindakan ini adalah CodeCatalysttindakan yang memungkinkan Anda menjalankan GitHub Tindakan dalam CodeCatalyst alur kerja. Untuk informasi selengkapnya, lihat Memanggil AWS Lambda fungsi menggunakan alur kerja.

- Menyebarkan tumpukan AWS CloudFormation

  Tindakan ini menyebarkan AWS CloudFormation tumpukan. Untuk informasi selengkapnya, lihat Menerapkan AWS CloudFormation tumpukan dengan alur kerja.

- Terapkan ke Amazon ECS

  Tindakan ini mendaftarkan definisi tugas Amazon ECS dan menerapkannya ke layanan Amazon ECS. Untuk informasi selengkapnya, lihat Menyebarkan aplikasi ke Amazon Elastic Container Service (ECS) dengan alur kerja.

- Terapkan ke klaster Kubernetes

  Tindakan ini menyebarkan aplikasi ke klaster Kubernetes. Untuk informasi selengkapnya, lihat Menerapkan aplikasi ke Amazon Elastic Kubernetes Service dengan alur kerja.

- Render definisi tugas Amazon ECS

  Tindakan ini menyisipkan URI gambar kontainer ke dalam file JSON definisi tugas Amazon ECS, membuat file definisi tugas baru. Untuk informasi selengkapnya, lihat Memodifikasi file definisi tugas Amazon ECS menggunakan alur kerja.

Dokumentasi untuk CodeCatalyst tindakan tersedia dalam panduan ini, dan di readme setiap tindakan.

Untuk informasi tentang CodeCatalyst tindakan yang tersedia, dan cara menambahkannya ke alur kerja, lihatMenambahkan tindakan ke CodeCatalyst alur kerja.

CodeCatalyst Tindakan Lab

Tindakan CodeCatalyst Labs adalah tindakan yang merupakan bagian dari Amazon CodeCatalyst Labs, tempat pembuktian untuk aplikasi eksperimental. CodeCatalyst Tindakan Labs telah dikembangkan untuk menampilkan integrasi dengan AWS layanan.

Tindakan CodeCatalyst Labs berikut tersedia:

- Menyebarkan ke Hosting AWS Amplify

  Tindakan ini menyebarkan aplikasi ke Amplify Hosting.
- Menyebarkan ke AWS App Runner

  Tindakan ini menyebarkan gambar terbaru dalam repositori gambar sumber ke App Runner.
- Terapkan ke Amazon CloudFront dan Amazon S3

  Tindakan ini menyebarkan aplikasi ke CloudFront dan Amazon S3.
- Menyebarkan dengan AWS SAM

  Tindakan ini menerapkan aplikasi tanpa server Anda dengan AWS Serverless Application Model ().AWS SAM
- Membatalkan Cache Amazon CloudFront

  Tindakan ini membatalkan CloudFront cache untuk kumpulan jalur tertentu.
- Webhook Keluar

  Tindakan ini memungkinkan pengguna untuk mengirim pesan dalam alur kerja ke server web arbitrer menggunakan permintaan HTTPS.
- Publikasikan ke AWS CodeArtifact

  Tindakan ini menerbitkan paket ke CodeArtifact repositori.
- Publikasikan ke Amazon SNS

  Tindakan ini memungkinkan pengguna untuk berintegrasi dengan Amazon SNS dengan membuat topik, menerbitkan topik, atau berlangganan topik.
- Dorong ke Amazon ECR

  Tindakan ini membangun dan menerbitkan image Docker ke repositori Amazon Elastic Container Registry (Amazon ECR).
- Pindai dengan Amazon CodeGuru Security

  Tindakan ini membuat arsip zip dari jalur kode yang dikonfigurasi dan menggunakan CodeGuru Keamanan untuk menjalankan pemindaian kode.
- Edisi Komunitas Terraform

Tindakan ini menjalankan Terraform Community Edition `plan` dan `apply` operasi.

Dokumentasi untuk tindakan CodeCatalyst Labs tersedia di readme setiap tindakan.

Untuk informasi tentang menambahkan tindakan CodeCatalyst Labs ke alur kerja dan melihat readme-nya, lihat. Menambahkan tindakan ke CodeCatalyst alur kerja

GitHub Tindakan

GitHub Tindakan sangat mirip dengan CodeCatalyst tindakan, kecuali bahwa itu dikembangkan untuk digunakan dengan GitHub alur kerja. Untuk detail tentang GitHub Tindakan, lihat dokumentasi GitHub Tindakan.

Anda dapat menggunakan GitHub Tindakan bersama CodeCatalyst tindakan asli dalam CodeCatalyst alur kerja.

Untuk kenyamanan Anda, CodeCatalyst konsol menyediakan akses ke beberapa GitHub Tindakan populer. Anda juga dapat menggunakan GitHub Tindakan apa pun yang tercantum di GitHub Marketplace (tunduk pada beberapa batasan).

Dokumentasi untuk GitHub Tindakan tersedia di readme setiap tindakan.

Untuk informasi selengkapnya, lihat Mengintegrasikan GitHub Tindakan ke dalam alur kerja.

Tindakan pihak ketiga

Tindakan pihak ketiga adalah tindakan yang ditulis oleh vendor pihak ketiga, dan tersedia di CodeCatalyst konsol. Contoh tindakan pihak ketiga termasuk tindakan Mend SCA dan SonarCloud Scan, masing-masing ditulis oleh Mend dan Sonar.

Dokumentasi untuk tindakan pihak ketiga tersedia di readme setiap tindakan. Dokumentasi tambahan mungkin juga disediakan oleh vendor pihak ketiga.

Untuk informasi tentang menambahkan tindakan pihak ketiga ke alur kerja dan melihat readme-nya, lihat. Menambahkan tindakan ke CodeCatalyst alur kerja

Menambahkan tindakan ke CodeCatalyst alur kerja

Gunakan petunjuk berikut untuk menambahkan tindakan ke alur kerja dan kemudian mengkonfigurasinya.

Untuk menambah dan mengkonfigurasi tindakan

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Di kiri atas, pilih + Tindakan, Katalog Tindakan muncul.
7. Dalam daftar drop-down, lakukan salah satu hal berikut:
   - Pilih Amazon CodeCatalyst untuk melihat CodeCatalyst, CodeCatalyst Lab, atau tindakan pihak ketiga.
     - CodeCatalyst tindakan memiliki AWS label berdasarkan.
     - CodeCatalyst Tindakan Labs memiliki label by CodeCatalyst Labs.
     - Tindakan pihak ketiga memiliki label by ***vendor***, di mana *vendor* adalah nama vendor pihak ketiga.
   - Pilih GitHubuntuk melihat daftar GitHub Tindakan yang dikuratori.
8. Di katalog tindakan, cari tindakan, lalu lakukan salah satu hal berikut:
   - Pilih tanda plus (+) untuk menambahkan tindakan ke alur kerja Anda.
   - Pilih nama tindakan untuk melihat readme nya.
9. Konfigurasikan tindakan. Pilih Visual untuk menggunakan editor visual, atau YAMAL untuk menggunakan editor YAMAL. Untuk petunjuk terperinci, lihat tautan berikut.

   Untuk petunjuk tentang menambahkan CodeCatalysttindakan, lihat:

   - Menambahkan aksi build
   - Menambahkan tindakan pengujian
   - Menambahkan tindakan “Terapkan ke Amazon ECS”
   - Menambahkan aksi “Deploy to Kubernetes cluster”
   - Menambahkan tindakan “Deploy AWS CloudFormation stack”
   - Menambahkan tindakan "AWS CDK menyebarkan”
   - Menambahkan tindakan "AWS CDK bootstrap”

- Menambahkan tindakan “Amazon S3 publish”
- Menambahkan tindakan "AWS Lambda memanggil”
- Menambahkan tindakan “Render definisi tugas Amazon ECS”

Untuk petunjuk tentang menambahkan tindakan CodeCatalyst Labs, lihat:

- Readme aksi. Anda dapat menemukan readme dengan memilih nama tindakan di katalog tindakan.

Untuk petunjuk tentang menambahkan GitHub Tindakan, lihat:

- Mengintegrasikan GitHub Tindakan ke dalam alur kerja

Untuk petunjuk tentang menambahkan tindakan pihak ketiga, lihat:

- Readme aksi. Anda dapat menemukan readme dengan memilih nama tindakan di katalog tindakan.

10. (Opsional) Pilih Validasi untuk memastikan kode YAMAL valid.
11. Pilih Komit untuk melakukan perubahan Anda.

Menghapus tindakan dari alur kerja

Gunakan petunjuk berikut untuk menghapus tindakan dari alur kerja.

Visual

Untuk menghapus tindakan menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.

7. Dalam diagram alur kerja, dalam tindakan yang ingin Anda hapus, pilih ikon elipsis vertikal, dan pilih Hapus.
8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menghapus tindakan menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Temukan bagian YAMAL yang berisi tindakan yang ingin Anda hapus.

   Pilih bagian dan tekan tombol hapus pada keyboard Anda.
8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Mengembangkan tindakan khusus

Anda dapat mengembangkan tindakan kustom untuk digunakan dalam alur kerja Anda menggunakan CodeCatalyst Action Development Kit (ADK). Anda kemudian dapat mempublikasikan tindakan ke katalog CodeCatalyst tindakan, sehingga CodeCatalyst pengguna lain dapat melihat dan menggunakannya dalam alur kerja mereka.

Untuk mengembangkan, menguji, dan mempublikasikan tindakan (tugas tingkat tinggi)

1. Instal alat dan paket yang diperlukan untuk mengembangkan suatu tindakan.
2. Buat CodeCatalyst repositori untuk menyimpan kode tindakan Anda.
3. Inisialisasi tindakan. Ini meletakkan file sumber yang diperlukan oleh tindakan, termasuk file definisi tindakan (`action.yml`) yang dapat Anda perbarui dengan kode Anda sendiri.

4. Bootstrap kode tindakan untuk mendapatkan alat dan pustaka yang diperlukan untuk membangun, menguji, dan merilis proyek tindakan.
5. Bangun tindakan di komputer lokal Anda, dan dorong perubahan ke CodeCatalyst repositori Anda.
6. Uji tindakan dengan pengujian unit secara lokal, dan jalankan alur kerja yang dihasilkan ADK. CodeCatalyst
7. Publikasikan tindakan ke katalog CodeCatalyst tindakan dengan memilih tombol Publikasikan di CodeCatalyst konsol.

Untuk langkah-langkah mendetail, lihat Panduan Pengembang Kit Pengembangan CodeCatalyst Aksi Amazon.

Mengelompokkan tindakan ke dalam kelompok aksi

Grup aksi berisi satu atau lebih tindakan. Mengelompokkan tindakan ke dalam grup tindakan membantu Anda menjaga alur kerja tetap teratur, dan juga memungkinkan Anda mengonfigurasi dependensi di antara grup yang berbeda.

> **Note**
>
> Anda tidak dapat menyarangkan grup aksi dalam kelompok tindakan atau tindakan lain.

Topik

Gunakan instruksi berikut untuk menentukan grup tindakan.

Visual

Tidak tersedia. Pilih YAMAL untuk melihat instruksi YAMAL.

YAML

Untuk mendefinisikan grup

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.

2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Dalam`Actions`, tambahkan kode yang mirip dengan berikut ini:

```
Actions:
  action-group-name:
    Actions:
      action-1:
        Identifier: aws/build@v1
        Configuration:
          ...
      action-2:
        Identifier: aws/build@v1
        Configuration:
          ...
```

   Untuk contoh lain, lihatContoh: Mendefinisikan dua kelompok aksi. Untuk informasi lebih lanjut, lihat deskripsi `action-group-name` properti di Tindakan halamanAlur kerja definisi YAMAL.
8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Contoh: Mendefinisikan dua kelompok aksi

Contoh berikut menunjukkan bagaimana mendefinisikan dua kelompok tindakan: `BuildAndTest` dan`Deploy`. `BuildAndTest`Kelompok ini mencakup dua tindakan (`Build`dan`Test`), dan `Deploy` kelompok juga mencakup dua tindakan (`DeployCloudFormationStack`dan`DeployToECS`).

```
Actions:
  BuildAndTest: # Action group 1
    Actions:
      Build:
        Identifier: aws/build@v1
        Configuration:
```

```
        ...
    Test:
      Identifier: aws/managed-test@v1
      Configuration:
Deploy: #Action group 2
  Actions:
    DeployCloudFormationStack:
      Identifier: aws/cfn-deploy@v1
      Configuration:
        ...
    DeployToECS:
      Identifier: aws/ecs-deploy@v1
      Configuration:
        ...
```

Mengkonfigurasi tindakan untuk bergantung pada tindakan lain

Secara default, saat Anda menambahkan tindakan ke alur kerja, tindakan tersebut ditambahkan berdampingan di editor visual. Ini berarti bahwa tindakan akan berjalan secara paralel ketika Anda memulai menjalankan alur kerja. Jika Anda ingin tindakan berjalan dalam urutan berurutan (dan muncul secara vertikal di editor visual), Anda harus mengatur dependensi di antara mereka. Misalnya, Anda dapat menyiapkan `Test` tindakan agar bergantung pada `Build` tindakan sehingga tindakan pengujian berjalan setelah tindakan build.

Anda dapat mengatur dependensi antara tindakan dan grup tindakan. Anda juga dapat mengonfigurasi one-to-many dependensi sehingga satu tindakan bergantung pada beberapa tindakan lain untuk memulai. Konsultasikan Pedoman untuk menyiapkan dependensi untuk memastikan pengaturan dependensi Anda sesuai dengan sintaks YAMAL alur kerja.

Topik

- Menyiapkan dependensi antar tindakan
- Pedoman untuk menyiapkan dependensi
- Contoh cara mengonfigurasi dependensi antar tindakan

Menyiapkan dependensi antar tindakan

Gunakan petunjuk berikut untuk mengatur dependensi antar tindakan dalam alur kerja.

Visual

Untuk mengatur dependensi menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Dalam diagram alur kerja, pilih tindakan yang akan bergantung pada tindakan lain.
8. Pilih tab Input.
9. Dalam Tergantung pada - opsional, lakukan hal berikut:

   Tentukan tindakan, grup tindakan, atau gerbang yang harus berjalan dengan sukses agar tindakan ini berjalan.

   Untuk informasi selengkapnya tentang fungsionalitas 'tergantung pada', lihat. Mengkonfigurasi tindakan untuk bergantung pada tindakan lain
10. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
11. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk mengatur dependensi menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.

7. Dalam tindakan yang akan bergantung pada yang lain, tambahkan kode yang mirip dengan yang berikut ini:

```
action-name:
  DependsOn:
    - action-1
```

   Untuk contoh lainnya, lihat Contoh cara mengonfigurasi dependensi antar tindakan. Untuk pedoman umum, lihatPedoman untuk menyiapkan dependensi. Untuk informasi selengkapnya, lihat deskripsi `DependsOn` properti di Alur kerja definisi YAMAL for your action.

8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Pedoman untuk menyiapkan dependensi

Saat mengonfigurasi dependensi, ikuti panduan ini:

- Jika suatu tindakan berada di dalam grup, tindakan itu hanya dapat bergantung pada tindakan lain dalam grup yang sama.
- Tindakan dan grup tindakan dapat bergantung pada tindakan dan kelompok tindakan lain pada tingkat yang sama dalam hierarki YAMAL, tetapi tidak pada tingkat yang berbeda.

Contoh cara mengonfigurasi dependensi antar tindakan

Contoh berikut menunjukkan cara mengkonfigurasi dependensi antara tindakan dan grup dalam file definisi alur kerja.

Topik

- Contoh: Mengkonfigurasi ketergantungan sederhana
- Contoh: Mengonfigurasi grup tindakan untuk bergantung pada tindakan
- Contoh: Mengkonfigurasi grup tindakan untuk bergantung pada grup tindakan lain
- Contoh: Mengonfigurasi grup tindakan agar bergantung pada beberapa tindakan

Contoh: Mengkonfigurasi ketergantungan sederhana

Contoh berikut menunjukkan cara mengkonfigurasi `Test` tindakan untuk bergantung pada `Build` tindakan menggunakan `DependsOn` properti.

```
Actions:
  Build:
    Identifier: aws/build@v1
    Configuration:
      ...
  Test:
    DependsOn:
      - Build
    Identifier: aws/managed-test@v1
     Configuration:
       ...
```

Contoh: Mengonfigurasi grup tindakan untuk bergantung pada tindakan

Contoh berikut menunjukkan cara mengkonfigurasi grup `DeployGroup` tindakan untuk bergantung pada `FirstAction` tindakan. Perhatikan bahwa action dan action group berada pada level yang sama.

```
Actions:
  FirstAction: #An action outside an action group
    Identifier: aws/github-actions-runner@v1
    Configuration:
      ...
  DeployGroup: #An action group containing two actions
    DependsOn:
      - FirstAction
    Actions:
      DeployAction1:
      ...
      DeployAction2:
      ...
```

Contoh: Mengkonfigurasi grup tindakan untuk bergantung pada grup tindakan lain

Contoh berikut menunjukkan cara mengkonfigurasi grup `DeployGroup` tindakan untuk bergantung pada grup `BuildAndTestGroup` tindakan. Perhatikan bahwa kelompok aksi berada pada level yang sama.

```
Actions:
  BuildAndTestGroup: # Action group 1
    Actions:
      BuildAction:
      ...
      TestAction:
      ...
  DeployGroup: #Action group 2
    DependsOn:
      - BuildAndTestGroup
    Actions:
      DeployAction1:
      ...
      DeployAction2:
      ...
```

Contoh: Mengonfigurasi grup tindakan agar bergantung pada beberapa tindakan

Contoh berikut menunjukkan cara mengonfigurasi grup `DeployGroup` tindakan agar bergantung pada `FirstAction` tindakan, `SecondAction` tindakan, serta grup `BuildAndTestGroup` tindakan. Perhatikan bahwa `DeployGroup` berada pada tingkat yang sama dengan`FirstAction`,`SecondAction`, dan`BuildAndTestGroup`.

```
Actions:
  FirstAction: #An action outside an action group
    ...
  SecondAction: #Another action
    ...
  BuildAndTestGroup: #Action group 1
    Actions:
      Build:
      ...
      Test:
      ...
  DeployGroup: #Action group 2
    DependsOn:
      - FirstAction
      - SecondAction
      - BuildAndTestGroup
    Actions:
      DeployAction1:
      ...
```

```
      DeployAction2:
      ...
```

Berbagi data antar tindakan dalam alur kerja menggunakan artefak

Artefak adalah output dari tindakan alur kerja, dan biasanya terdiri dari folder atau arsip file. Artefak penting karena memungkinkan Anda berbagi file dan informasi antar tindakan.

Misalnya, Anda mungkin memiliki tindakan build yang menghasilkan `sam-template.yml` file, tetapi Anda ingin tindakan penerapan menggunakannya. Dalam skenario ini, Anda akan menggunakan artefak untuk memungkinkan tindakan build membagikan `sam-template.yml` file dengan tindakan penerapan. Kode mungkin terlihat seperti ini:

```
Actions:
  BuildAction:
    Identifier: aws/build@v1
    Steps:
      - Run: sam package --output-template-file sam-template.yml
    Outputs:
      Artifacts:
        - Name: MYARTIFACT
          Files:
            - sam-template.yml
  DeployAction:
    Identifier: aws/cfn-deploy@v1
    Inputs:
      Artifacts:
        - MYARTIFACT
    Configuration:
      template: sam-template.yml
```

Dalam kode sebelumnya, build action (`BuildAction`) menghasilkan `sam-template.yml` file, dan kemudian menambahkannya ke artefak keluaran yang disebut`MYARTIFACT`. Tindakan penerapan berikutnya (`DeployAction`) menentukan `MYARTIFACT` sebagai input, memberikannya akses ke file. `sam-template.yml`

Dapatkah saya membagikan artefak tanpa menentukannya sebagai output dan input?

Ya, Anda dapat berbagi artefak antar tindakan tanpa menentukannya di `Outputs` dan `Inputs` bagian kode YAMM tindakan Anda. Untuk melakukan ini, Anda harus mengaktifkan berbagi

komputasi. Untuk informasi selengkapnya tentang berbagi komputasi dan cara menentukan artefak saat dihidupkan, lihat. Berbagi komputasi di seluruh tindakan

> **Note**
>
> Meskipun fitur berbagi komputasi memungkinkan Anda menyederhanakan kode YAMM alur kerja Anda dengan menghilangkan kebutuhan untuk `Inputs` bagian `Outputs` dan, fitur ini memiliki batasan yang harus Anda ketahui sebelum Anda menyalakannya. Untuk informasi tentang batasan ini, lihatPertimbangan untuk berbagi komputasi.

Bisakah saya berbagi artefak antar alur kerja?

Tidak, Anda tidak dapat berbagi artefak di antara alur kerja yang berbeda; namun, Anda dapat berbagi artefak antar tindakan dalam alur kerja yang sama.

Topik

- Mendefinisikan artefak keluaran
- Mendefinisikan artefak input
- Merujuk file dalam artefak
- Mengunduh artefak
- Contoh artefak

Mendefinisikan artefak keluaran

Gunakan petunjuk berikut untuk menentukan artefak yang Anda inginkan tindakan untuk dihasilkan. Artefak ini kemudian tersedia untuk tindakan lain untuk digunakan.

> **Note**
>
> Tidak semua tindakan mendukung artefak keluaran. Untuk menentukan apakah tindakan Anda mendukungnya, jalankan melalui instruksi editor visual yang mengikuti, dan lihat apakah tindakan tersebut menyertakan tombol artefak Output pada tab Output. Jika ya, artefak keluaran didukung.

Visual

Untuk menentukan artefak keluaran menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Dalam diagram alur kerja, pilih tindakan yang akan menghasilkan artefak.
8. Pilih tab Output.
9. Di bawah Artefak, pilih Tambahkan artefak.
10. Pilih Tambahkan artefak, dan masukkan informasi ke dalam bidang, sebagai berikut.

    Bangun nama artefak

    Tentukan nama artefak yang dihasilkan oleh tindakan. Nama artifak harus unik dalam alur kerja, dan terbatas pada karakter alfanumerik (a-z, A-Z, 0-9) dan garis bawah (_). Spasi, tanda hubung (-), dan karakter khusus lainnya tidak diperbolehkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan spasi, tanda hubung, dan karakter khusus lainnya dalam nama artefak keluaran.

    Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

    File yang dihasilkan oleh build

    Tentukan file yang CodeCatalyst termasuk dalam artefak yang dihasilkan oleh tindakan. File-file ini dihasilkan oleh tindakan alur kerja saat dijalankan, dan juga tersedia di repositori sumber Anda. Jalur file dapat berada di repositori sumber atau artefak dari tindakan sebelumnya, dan relatif terhadap repositori sumber atau root artefak. Anda dapat menggunakan pola glob untuk menentukan jalur. Contoh:

    - Untuk menentukan satu file yang ada di root lokasi build atau lokasi repositori sumber, gunakan. `my-file.jar`

- Untuk menentukan satu file dalam subdirektori, gunakan `directory/my-file.jar` atau`directory/subdirectory/my-file.jar`.
- Untuk menentukan semua file, gunakan`"**/*"`. Pola `**` glob menunjukkan untuk mencocokkan sejumlah subdirektori.
- Untuk menentukan semua file dan direktori dalam direktori bernama`directory`, gunakan`"directory/**/*"`. Pola `**` glob menunjukkan untuk mencocokkan sejumlah subdirektori.
- Untuk menentukan semua file dalam direktori bernama`directory`, tetapi tidak salah satu subdirektorinya, gunakan. `"directory/*"`

> **Note**
>
> Jika jalur file Anda menyertakan satu atau beberapa tanda bintang (*) atau karakter khusus lainnya, lampirkan jalur dengan tanda kutip ganda (). "" Untuk informasi selengkapnya tentang karakter khusus, lihatPedoman dan konvensi sintaks.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

> **Note**
>
> Anda mungkin perlu menambahkan awalan ke jalur file untuk menunjukkan artefak atau sumber mana yang akan menemukannya. Untuk informasi selengkapnya, lihat Mereferensikan file dalam repositori sumber dan Merujuk file dalam artefak.

11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMM alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menentukan artefak keluaran menggunakan editor YAMM

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.

3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Dalam tindakan alur kerja, tambahkan kode yang mirip dengan berikut ini:

```
action-name:
  Outputs:
    Artifacts:
      - Name: artifact-name
        Files:
          - file-path-1
          - file-path-2
```

   Untuk contoh lainnya, lihat Contoh artefak. Untuk informasi selengkapnya, lihat Alur kerja definisi YAMAL untuk tindakan Anda.
8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMM alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Mendefinisikan artefak input

Jika Anda ingin menggunakan artefak yang dihasilkan oleh tindakan lain, Anda harus menentukannya sebagai masukan untuk tindakan saat ini. Anda mungkin dapat menentukan beberapa artefak sebagai input—itu tergantung pada tindakan. Untuk informasi selengkapnya, lihat Alur kerja definisi YAMAL untuk tindakan Anda.

> **Note**
>
> Anda tidak dapat mereferensikan artefak dari alur kerja lain.

Gunakan petunjuk berikut untuk menentukan artefak dari tindakan lain sebagai masukan untuk tindakan saat ini.

Prasyarat

Sebelum Anda mulai, pastikan Anda memiliki output artefak dari tindakan lain. Untuk informasi selengkapnya, lihat Mendefinisikan artefak keluaran. Mengeluarkan artefak membuatnya tersedia untuk tindakan lain untuk digunakan.

Visual

Untuk menentukan artefak sebagai masukan untuk tindakan (editor visual)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Dalam diagram alur kerja, pilih tindakan di mana Anda ingin menentukan artefak sebagai input.
8. Pilih Input.
9. Dalam Artefak - opsional, lakukan hal berikut:

   Tentukan artefak dari tindakan sebelumnya yang ingin Anda berikan sebagai masukan untuk tindakan ini. Artefak ini harus sudah didefinisikan sebagai artefak keluaran dalam tindakan sebelumnya.

   Jika Anda tidak menentukan artefak input apa pun, maka Anda harus menentukan setidaknya satu repositori sumber di bawah. `action-name/Inputs/Sources`

   Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

   > **Note**
   >
   > Jika Artefacts - daftar drop-down opsional tidak tersedia (editor visual), atau jika Anda mendapatkan kesalahan saat memvalidasi YAMG (editor YAMM), itu mungkin karena tindakan hanya mendukung satu input. Dalam hal ini, coba hapus input sumber.

10. (Opsional) Pilih Validasi untuk memvalidasi kode YAMM alur kerja sebelum melakukan.

11. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menentukan artefak sebagai masukan ke tindakan (editor YAMB)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Dalam tindakan di mana Anda ingin menentukan artefak sebagai input, tambahkan kode yang mirip dengan berikut ini:

```
action-name:
  Inputs:
    Artifacts:
      - artifact-name
```

   Untuk contoh lainnya, lihat Contoh artefak.
8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMM alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Merujuk file dalam artefak

Jika Anda memiliki file yang berada dalam artefak, dan Anda perlu merujuk ke file ini di salah satu tindakan alur kerja Anda, selesaikan prosedur berikut.

> **Note**
>
> Lihat juga Mereferensikan file dalam repositori sumber.

Untuk referensi file dalam artefak

- Dalam tindakan di mana Anda ingin mereferensikan file, tambahkan kode yang mirip dengan yang berikut ini:

```
Actions:
  My-action:
    Inputs:
      Sources:
        - WorkflowSource
      Artifacts:
        - artifact-name
    Configuration:
      Steps:
        - run: cd $CATALYST_SOURCE_DIR_artifact-name/build-output && cat file.txt
```

Dalam kode sebelumnya, tindakan terlihat di `build-output` direktori dalam artefak *`nama artefak`* untuk menemukan dan menampilkan file. `file.txt`

Untuk contoh lainnya, lihat Contoh artefak.

> **Note**
>
> Anda mungkin dapat menghilangkan `$CATALYST_SOURCE_DIR_`*`artifact-name`*`/` awalan tergantung pada cara Anda mengonfigurasi tindakan Anda. Untuk informasi lebih lanjut, lihat panduan berikut.

Panduan tentang cara merujuk ke variabel:

- Jika tindakan Anda hanya menyertakan satu item di bawah `Inputs` (misalnya, itu termasuk satu artefak input dan tidak ada sumber), maka Anda dapat menghilangkan awalan dan menentukan hanya jalur file relatif terhadap root artefak.
- Anda juga dapat menghilangkan awalan jika file berada di input utama. Input utama adalah`WorkflowSource`, atau artefak input pertama yang terdaftar, jika tidak `WorkflowSource` ada.
- Awalan bisa berbeda tergantung pada tindakan yang Anda gunakan. Untuk informasi lebih lanjut, lihat tabel di bawah ini.

Tipe tindakan	Awalan jalur file yang akan digunakan	Contoh
Bangun tindakan, uji tindakan	`$CATALYST_SOURCE_DIR_` *artifact-name* `/`	`$CATALYST_SOURCE_DIR_MyArtifact/folder1/file.txt`

Semua tindakan lainnya	`$CATALYST_SOURCE_DIR_` *artifact-name* `/`
	
	atau
	
	`/artifacts/` *current-action-name* */artifact-name* */(jalur ini adalah tautan simbolis ke $CATALYST_SOURCE_DIR_ artefact-name/)*
		`$CATALYST_SOURCE_DIR_MyArtifact/folder1/file.txt`
		
		atau
		
		`/artifacts/MyCurrentAction/MyArtifact/folder1/file.txt`

Mengunduh artefak

Anda dapat mengunduh dan memeriksa artefak yang dihasilkan oleh tindakan alur kerja Anda untuk tujuan pemecahan masalah. Ada dua jenis artefak yang dapat Anda unduh:

- Artefak sumber — Artefak yang berisi snapshot konten repositori sumber seperti yang ada saat proses dijalankan.
- Artefak alur kerja - Artefak yang didefinisikan dalam `Outputs` properti file konfigurasi alur kerja Anda.

Untuk mengunduh keluaran artefak berdasarkan alur kerja

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.

5. aDi bawah nama alur kerja, pilih Runs.
6. Di Riwayat Jalankan, di kolom Run ID, pilih run. Misalnya, `Run-95a4d`.
7. Di bawah nama run, pilih Artefak.
8. Di samping artefak, pilih Unduh. File arsip diunduh. Nama filenya terdiri dari tujuh karakter acak.
9. Ekstrak arsip menggunakan utilitas ekstraksi arsip pilihan Anda.

Contoh artefak

Contoh berikut menunjukkan cara mengeluarkan, memasukkan, dan referensi artefak dalam file definisi alur kerja.

Topik

- Contoh: Mengeluarkan artefak
- Contoh: Memasukkan artefak yang dihasilkan oleh tindakan lain
- Contoh: Merujuk file dalam beberapa artefak
- Contoh: Mereferensikan file dalam satu artefak
- Contoh: Mereferensikan file dalam artefak saat ada WorkflowSource

Contoh: Mengeluarkan artefak

Contoh berikut menunjukkan bagaimana untuk output artefak yang mencakup dua file.jar.

```
Actions:
  Build:
    Identifier: aws/build@v1
    Outputs:
      Artifacts:
        - Name: ARTIFACT1
          Files:
            - build-output/file1.jar
            - build-output/file2.jar
```

Contoh: Memasukkan artefak yang dihasilkan oleh tindakan lain

Contoh berikut menunjukkan kepada Anda cara menampilkan artefak yang dipanggil `ARTIFACT4BuildActionA`, dan memasukkannya ke dalam`BuildActionB`.

```
Actions:
  BuildActionA:
    Identifier: aws/build@v1
    Outputs:
      Artifacts:
        - Name: ARTIFACT4
          Files:
            - build-output/file1.jar
            - build-output/file2.jar
  BuildActionB:
    Identifier: aws/build@v1
    Inputs:
      Artifacts:
        - ARTIFACT4
    Configuration:
```

Contoh: Merujuk file dalam beberapa artefak

Contoh berikut menunjukkan kepada Anda cara menampilkan dua artefak bernama ART5 dan ART6 masuk`BuildActionC`, dan kemudian referensi dua file bernama `file5.txt` (dalam artefak`ART5`) dan `file6.txt` (dalam artefak`ART6`) di `BuildActionD` (bawah). `Steps`

> **Note**
>
> Untuk informasi selengkapnya tentang referensi file, lihatMerujuk file dalam artefak.

> **Note**
>
> Meskipun contoh menunjukkan `$CATALYST_SOURCE_DIR_ART5` awalan yang digunakan, Anda bisa menghilangkannya. Ini karena `ART5` merupakan masukan utama. Untuk mempelajari lebih lanjut tentang input utama, lihatMerujuk file dalam artefak.

```
Actions:
  BuildActionC:
    Identifier: aws/build@v1
    Outputs:
      Artifacts:
        - Name: ART5
```

```
          Files:
            - build-output/file5.txt
        - Name: ART6
          Files:
            - build-output/file6.txt
  BuildActionD:
    Identifier: aws/build@v1
    Inputs:
      Artifacts:
        - ART5
        - ART6
    Configuration:
      Steps:
        - run: cd $CATALYST_SOURCE_DIR_ART5/build-output && cat file5.txt
        - run: cd $CATALYST_SOURCE_DIR_ART6/build-output && cat file6.txt
```

Contoh: Mereferensikan file dalam satu artefak

Contoh berikut menunjukkan kepada Anda cara menampilkan satu artefak bernama `ART7`BuildActionE, dan kemudian referensi `file7.txt` (dalam artefakART7) di `BuildActionF` (bawah`Steps`).

Perhatikan bagaimana referensi tidak memerlukan awalan `$CATALYST_SOURCE_DIR_` *`nama artefak`* di depan `build-output` direktori seperti yang terjadi di. Contoh: Merujuk file dalam beberapa artefak Ini karena hanya ada satu item yang ditentukan di bawah`Inputs`.

> **Note**
>
> Untuk informasi selengkapnya tentang referensi file, lihatMerujuk file dalam artefak.

```
Actions:
  BuildActionE:
    Identifier: aws/build@v1
    Outputs:
      Artifacts:
        - Name: ART7
          Files:
            - build-output/file7.txt
  BuildActionF:
    Identifier: aws/build@v1
```

```
    Inputs:
      Artifacts:
        - ART7
    Configuration:
      Steps:
        - run: cd build-output && cat file7.txt
```

Contoh: Mereferensikan file dalam artefak saat ada WorkflowSource

Contoh berikut menunjukkan kepada Anda cara menampilkan satu artefak bernama `ART8BuildActionG`, dan kemudian referensi `file8.txt` (dalam artefak`ART8`) di `BuildActionH` (bawah`Steps`).

Perhatikan bagaimana referensi memerlukan awalan `$CATALYST_SOURCE_DIR_` *`nama artefak`*, seperti yang terjadi di. Contoh: Merujuk file dalam beberapa artefak Ini karena ada beberapa item yang ditentukan di bawah `Inputs` (sumber dan artefak), jadi Anda memerlukan awalan untuk menunjukkan di mana harus mencari file.

> **Note**
>
> Untuk informasi selengkapnya tentang referensi file, lihatMerujuk file dalam artefak.

```
Actions:
  BuildActionG:
    Identifier: aws/build@v1
    Outputs:
      Artifacts:
        - Name: ART8
          Files:
            - build-output/file8.txt
  BuildActionH:
    Identifier: aws/build@v1
    Inputs:
      Sources:
        - WorkflowSource
      Artifacts:
        - ART8
    Configuration:
      Steps:
        - run: cd $CATALYST_SOURCE_DIR_ART8/build-output && cat file8.txt
```

Menentukan versi mayor, minor, atau patch dari suatu tindakan

Secara default, saat Anda menambahkan tindakan ke alur kerja, Amazon CodeCatalyst menambahkan versi lengkap ke file definisi alur kerja menggunakan format:

v*major*.*minor*.*patch*

Sebagai contoh:

```
My-Build-Action:
  Identifier: aws/build@v1.0.0
```

Anda dapat mempersingkat versi lengkap di `Identifier` properti sehingga alur kerja selalu menggunakan versi minor atau patch terbaru dari tindakan tersebut.

Misalnya, jika Anda menentukan:

```
My-CloudFormation-Action:
  Identifier: aws/cfn-deploy@v1.0
```

... dan versi patch terbaru adalah`1.0.4`, maka tindakan akan digunakan`1.0.4`. Jika versi yang lebih baru dirilis, katakanlah`1.0.5`, maka tindakan akan digunakan`1.0.5`. Jika versi minor dirilis, katakanlah`1.1.0`, maka tindakan akan terus digunakan`1.0.5`.

Untuk petunjuk terperinci tentang menentukan versi, lihat salah satu topik berikut.

Gunakan petunjuk berikut untuk menunjukkan versi tindakan yang ingin digunakan alur kerja Anda. Anda dapat menentukan versi mayor atau minor terbaru, atau versi patch tertentu.

Sebaiknya gunakan versi minor atau patch terbaru dari suatu tindakan.

Visual

Tidak tersedia. Pilih YAMG untuk melihat instruksi YAMG.

YAML

Untuk mengonfigurasi alur kerja agar menggunakan versi terbaru dari suatu tindakan, atau versi tambalan tertentu

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.

2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Temukan tindakan yang versinya ingin Anda edit.
8. Temukan `Identifier` properti tindakan, dan atur versinya ke salah satu dari berikut ini:

   - action-identifier @v *`major`* — Gunakan sintaks ini agar alur kerja menggunakan versi mayor tertentu, dan izinkan versi minor dan patch terbaru dipilih secara otomatis.
   - *`pengenal tindakan @v mayor.minor`* — Gunakan sintaks ini agar alur kerja menggunakan versi minor tertentu, dan izinkan versi patch terbaru dipilih secara otomatis.
   - *`pengenal tindakan @v mayor.kecil.patch`* — Gunakan sintaks ini agar alur kerja menggunakan versi patch tertentu.

   > **Note**
   >
   > Jika Anda tidak yakin versi mana yang tersedia, lihatMenentukan versi tindakan mana yang tersedia.

   > **Note**
   >
   > Anda tidak dapat menghilangkan versi utama.

9. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
10. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Menentukan versi tindakan mana yang tersedia

Gunakan petunjuk berikut untuk menentukan versi tindakan yang tersedia untuk Anda gunakan dalam alur kerja.

Visual

Untuk menentukan versi tindakan mana yang tersedia

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Temukan tindakan yang versinya ingin Anda lihat:
   a. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
   b. Pilih nama alur kerja apa pun, atau buat satu. Untuk informasi tentang membuat alur kerja, lihatMembuat alur kerja.
   c. Pilih Edit.
   d. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
   e. Di daftar drop-down, pilih Amazon CodeCatalyst untuk melihat CodeCatalyst, CodeCatalyst Lab, dan tindakan pihak ketiga, atau pilih GitHubuntuk melihat Tindakan yang dikurasi GitHub.
   f. Cari tindakan, dan pilih namanya. Jangan memilih tanda plus (+).

      Detail tentang tindakan muncul.
4. Di kotak dialog detail tindakan, di dekat kanan atas, pilih daftar drop-down Versi untuk melihat daftar versi tindakan yang tersedia.

YAML

Tidak tersedia. Pilih 'visual' untuk melihat instruksi editor visual.

Melihat kode sumber tindakan

Anda dapat melihat kode sumber tindakan untuk memastikannya tidak mengandung kode berisiko, kerentanan keamanan, atau cacat lainnya.

Gunakan petunjuk berikut untuk melihat kode sumber tindakan CodeCatalyst, CodeCatalyst Lab, atau pihak ketiga.

> **Note**
>
> Untuk melihat kode sumber GitHubAction, buka halaman tindakan di GitHub Marketplace. Halaman ini menyertakan tautan ke repositori tindakan, tempat Anda dapat menemukan kode sumber tindakan.

> **Note**
>
> Anda tidak dapat melihat kode sumber dari CodeCatalyst tindakan berikut: build, test, GitHub Actions.

> **Note**
>
> AWS tidak mendukung atau menjamin kode tindakan Tindakan atau GitHub tindakan pihak ketiga.

Untuk melihat kode sumber tindakan

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Temukan tindakan yang kodenya ingin Anda lihat:
   a. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
   b. Pilih nama alur kerja apa pun, atau buat satu. Untuk informasi tentang membuat alur kerja, lihatMembuat alur kerja.
   c. Pilih Edit.
   d. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
   e. Di daftar drop-down, pilih Amazon CodeCatalyst untuk melihat CodeCatalyst, CodeCatalyst Labs, dan tindakan pihak ketiga.
   f. Cari tindakan, dan pilih namanya. Jangan memilih tanda plus (+).

      Detail tentang tindakan muncul.
4. Di kotak dialog detail tindakan, di dekat bagian bawah, pilih Unduh.

Sebuah halaman muncul, menunjukkan bucket Amazon S3 tempat kode sumber tindakan berada. Untuk informasi tentang Amazon S3, lihat Apa itu Amazon S3? di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

5. Periksa kode untuk memastikannya memenuhi harapan Anda akan kualitas dan keamanan.

Mengintegrasikan GitHub Tindakan ke dalam alur kerja

GitHub Aksi sangat mirip dengan CodeCatalyst tindakan, kecuali bahwa itu dikembangkan untuk digunakan dengan GitHub alur kerja. Untuk detail tentang GitHub Tindakan, lihat dokumentasi GitHub Tindakan.

Anda dapat menggunakan GitHub Tindakan bersama CodeCatalyst tindakan asli dalam CodeCatalyst alur kerja.

Ada dua cara untuk menambahkan GitHub Action ke CodeCatalyst alur kerja:

- Anda dapat memilih GitHub Tindakan dari daftar yang dikuratori di CodeCatalyst konsol. Beberapa GitHub Tindakan populer tersedia. Untuk informasi selengkapnya, lihat Menambahkan Action yang dikuratori GitHub .
- Jika GitHub Tindakan yang ingin Anda gunakan tidak tersedia di CodeCatalyst konsol, Anda dapat menambahkannya menggunakan GitHub tindakan Tindakan.

  GitHub Tindakan Tindakan adalah CodeCatalyst tindakan yang membungkus GitHub Action dan membuatnya kompatibel dengan CodeCatalyst alur kerja.

  Berikut adalah contoh GitHub tindakan Actions yang membungkus GitHubSuper-Linter Action:

```
Actions:
  GitHubAction:
    Identifier: aws/github-actions-runner@v1
    Configuration:
      Steps:
        - name: Lint Code Base
          uses: github/super-linter@v4
          env:
            VALIDATE_ALL_CODEBASE: "true"
            DEFAULT_BRANCH: main
```

Dalam kode sebelumnya, CodeCatalyst GitHub tindakan Tindakan (diidentifikasi oleh`aws/github-actions-runner@v1`) membungkus tindakan Super-Linter (diidentifikasi oleh`github/super-linter@v4`), membuatnya bekerja dalam alur kerja. CodeCatalyst

Untuk informasi selengkapnya, lihat Menambahkan GitHub tindakan "Tindakan”.

Semua GitHub tindakan—baik yang dikuratori maupun tidak—harus dibungkus di dalam GitHub Actions action (`aws/github-actions-runner@v1`), seperti yang ditunjukkan pada contoh sebelumnya. Pembungkus diperlukan agar tindakan berfungsi dengan baik.

Bagaimana GitHub tindakan berbeda dari CodeCatalyst tindakan?

GitHub Tindakan yang digunakan di dalam CodeCatalyst alur kerja tidak memiliki tingkat akses dan integrasi yang sama dengan AWS dan CodeCatalyst fitur (seperti lingkungan dan masalah) yang dilakukan CodeCatalyst tindakan.

Bisakah GitHub Tindakan berinteraksi dengan CodeCatalyst tindakan lain dalam alur kerja?

Ya. Misalnya, GitHub Actions dapat menggunakan variabel yang dihasilkan oleh CodeCatalyst tindakan lain sebagai input, dan juga dapat berbagi parameter output dan artefak dengan CodeCatalyst tindakan. Untuk informasi selengkapnya, lihat Mengekspor parameter GitHub output sehingga tindakan lain dapat menggunakannya dan Mereferensikan parameter GitHub keluaran.

GitHub Tindakan apa yang bisa saya gunakan?

Anda dapat menggunakan GitHub Action apa pun yang tersedia melalui CodeCatalyst konsol, dan GitHub Action apa pun yang tersedia di GitHubMarketplace. Jika Anda memutuskan untuk menggunakan GitHub Action dari Marketplace, ingatlah batasan berikut.

Keterbatasan GitHub Tindakan di CodeCatalyst

- GitHub Tindakan tidak dapat digunakan dengan tipe komputasi CodeCatalyst Lambda.

- GitHub Tindakan berjalan pada image Docker lingkungan runtime November 2022, yang mencakup perkakas yang lebih lama. Untuk informasi selengkapnya tentang gambar dan perkakas, lihatMenentukan gambar Docker lingkungan runtime.

- GitHub Tindakan yang secara internal bergantung pada `github` konteks atau sumber daya GitHub spesifik referensi itu tidak akan berfungsi. CodeCatalyst Misalnya, tindakan berikut tidak akan berfungsi di CodeCatalyst:
  - Tindakan yang mencoba menambah, mengubah, atau memperbarui GitHub sumber daya. Contohnya termasuk tindakan yang memperbarui permintaan tarik, atau membuat masalah di GitHub.
  - Hampir semua tindakan tercantum di https://github.com/actions.
- GitHub Tindakan yang merupakan tindakan kontainer Docker akan berfungsi, tetapi harus dijalankan oleh pengguna Docker default (root). Jangan menjalankan tindakan sebagai pengguna 1001. (Pada saat penulisan, pengguna 1001 bekerja di GitHub, tetapi tidak di CodeCatalyst.) Untuk informasi selengkapnya, lihat topik USER di dukungan Dockerfile untuk GitHub Tindakan.

Untuk daftar GitHub Tindakan yang tersedia melalui CodeCatalyst konsol, lihatMenambahkan Action yang dikuratori GitHub .

Bagaimana cara menambahkan GitHub Action (langkah-langkah tingkat tinggi)?

Langkah-langkah tingkat tinggi untuk menambahkan GitHub Action ke CodeCatalyst alur kerja adalah sebagai berikut:

1. Dalam CodeCatalyst proyek Anda, Anda membuat alur kerja. Alur kerja adalah tempat Anda menentukan cara membuat, menguji, dan menerapkan aplikasi Anda. Untuk informasi selengkapnya, lihat Memulai dengan alur kerja.
2. Di alur kerja, Anda menambahkan GitHub Tindakan yang dikuratori atau menambahkan GitHub tindakan Tindakan.
3. Anda melakukan salah satu hal berikut:
   - Jika Anda memilih untuk menambahkan tindakan yang dikuratori, konfigurasikan. Untuk informasi selengkapnya, lihat Menambahkan Action yang dikuratori GitHub .
   - Jika Anda memilih untuk menambahkan tindakan yang tidak dikurasi, dalam tindakan GitHubTindakan, Anda menempelkan kode YAMAL GitHub Action. Anda dapat menemukan kode ini di halaman detail GitHub Tindakan yang Anda pilih di GitHubMarketplace. Anda mungkin perlu memodifikasi kode sedikit agar berfungsi CodeCatalyst. Untuk informasi selengkapnya, lihat Menambahkan GitHub tindakan "Tindakan”.
4. (Opsional) Dalam alur kerja, Anda menambahkan tindakan lain seperti tindakan build dan test. Untuk informasi selengkapnya, lihat Bangun, uji, dan terapkan dengan alur kerja di CodeCatalyst.

5. Anda memulai alur kerja baik secara manual atau otomatis melalui pemicu. Alur kerja menjalankan GitHub Tindakan dan tindakan lainnya dalam alur kerja. Untuk informasi selengkapnya, lihat Memulai alur kerja berjalan secara manual.

Untuk langkah-langkah rinci, lihat:

- Menambahkan Action yang dikuratori GitHub .
- Menambahkan GitHub tindakan "Tindakan”.

Apakah GitHub Aksi berjalan GitHub?

Tidak. GitHub Action berjalan di CodeCatalyst, menggunakan CodeCatalyst mesin build.

Bisakah saya menggunakan GitHub alur kerja juga?

Tidak.

Topik

- Tutorial: Kode lint menggunakan GitHub Action dalam alur kerja
- Menambahkan GitHub tindakan "Tindakan”
- Menambahkan Action yang dikuratori GitHub
- Mengekspor parameter GitHub output sehingga tindakan lain dapat menggunakannya
- Mereferensikan parameter GitHub keluaran
- GitHub Tindakan “Tindakan” definisi YAMAL

Tutorial: Kode lint menggunakan GitHub Action dalam alur kerja

Dalam tutorial ini, Anda menambahkan Super-Linter GitHub Action ke alur kerja Amazon CodeCatalyst . Tindakan Super-Linter memeriksa kode, menemukan area di mana kode memiliki kesalahan, masalah pemformatan, dan konstruksi yang mencurigakan, dan kemudian mengeluarkan hasilnya ke konsol). CodeCatalyst Setelah menambahkan linter ke alur kerja Anda, Anda menjalankan alur kerja untuk lint contoh aplikasi Node.js (). `app.js` Anda kemudian memperbaiki masalah yang dilaporkan dan menjalankan alur kerja lagi untuk melihat apakah perbaikan berhasil.

> **Tip**
>
> Pertimbangkan untuk menggunakan Super-Linter untuk lint file YAMAL, seperti template.AWS CloudFormation

Topik

Prasyarat

Sebelum Anda mulai, Anda akan membutuhkan:

- CodeCatalyst Ruang dengan yang terhubung Akun AWS. Untuk informasi selengkapnya, lihat Menciptakan ruang.
- Proyek kosong di CodeCatalyst ruang Anda disebut`codecatalyst-linter-project`. Pilih opsi Mulai dari awal untuk membuat proyek ini.

  Untuk informasi selengkapnya, lihat Membuat proyek kosong di Amazon CodeCatalyst.

Langkah 1: Buat repositori sumber

Pada langkah ini, Anda membuat repositori sumber di. CodeCatalyst Anda akan menggunakan repositori ini untuk menyimpan contoh file sumber aplikasi,`app.js`, untuk tutorial ini.

Untuk informasi selengkapnya tentang repositori sumber, lihat. Membuat repositori sumber

Untuk membuat repositori sumber

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek Anda,`codecatalyst-linter-project`.

3. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
4. Pilih Tambahkan repositori, lalu pilih Buat repositori.
5. Dalam nama Repositori, masukkan:

```
codecatalyst-linter-source-repository
```

6. Pilih Buat.

Langkah 2: Tambahkan file app.js

Pada langkah ini, Anda menambahkan `app.js` file ke repositori sumber Anda. Kode `app.js` berisi fungsi yang memiliki beberapa kesalahan yang akan ditemukan linter.

Untuk menambahkan file app.js

1. Di CodeCatalyst konsol, pilih proyek Anda,`codecatalyst-linter-project`.
2. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
3. Dari daftar repositori sumber, pilih repositori Anda,. `codecatalyst-linter-source-repository`
4. Di File, pilih Buat file.
5. Di kotak teks, masukkan kode berikut:

```
// const axios = require('axios')
// const url = 'http://checkip.amazonaws.com/';
let response;
/**
 *
 * Event doc: https://docs.aws.amazon.com/apigateway/latest/developerguide/set-up-
lambda-proxy-integrations.html#api-gateway-simple-proxy-for-lambda-input-format
 * @param {Object} event - API Gateway Lambda Proxy Input Format
 *
 * Context doc: https://docs.aws.amazon.com/lambda/latest/dg/nodejs-prog-model-
context.html
 * @param {Object} context
 *
 * Return doc: https://docs.aws.amazon.com/apigateway/latest/developerguide/set-up-
lambda-proxy-integrations.html
 * @returns {Object} object - API Gateway Lambda Proxy Output Format
 *
 */
```

```
exports.lambdaHandler = async (event, context) => {
  try {
    // const ret = await axios(url);
    response = {
      statusCode: 200,
      'body': JSON.stringify({
        message: 'hello world'
        // location: ret.data.trim()
      })
    }
  } catch (err) {
    console.log(err)
    return err
  }

    return response
}
```

6. Untuk nama File, masukkan`app.js`. Simpan opsi default lainnya.
7. Pilih Terapkan.

   Anda sekarang telah membuat file bernama`app.js`.

Langkah 3: Buat alur kerja yang menjalankan tindakan Super-Linter

Pada langkah ini, Anda membuat alur kerja yang menjalankan tindakan Super-Linter saat Anda mendorong kode ke repositori sumber Anda. Alur kerja terdiri dari blok bangunan berikut, yang Anda tentukan dalam file YAMAL:

- Pemicu - Pemicu ini memulai alur kerja yang dijalankan secara otomatis saat Anda mendorong perubahan ke repositori sumber Anda. Untuk informasi lebih lanjut tentang menggunakan pemicu, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.
- GitHub Tindakan "Tindakan" — Pada pemicu, tindakan Tindakan menjalankan GitHub tindakan Super-Linter, yang pada gilirannya memeriksa semua file di repositori sumber Anda. Jika linter menemukan masalah, tindakan alur kerja gagal.

Untuk membuat alur kerja yang menjalankan tindakan Super-Linter

1. Di CodeCatalyst konsol, pilih proyek Anda,`codecatalyst-linter-project`.
2. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.

3. Pilih Buat alur kerja.
4. Untuk repositori Sumber, pilih. `codecatalyst-linter-source-repository`
5. Untuk Cabang, pilih`main`.
6. Pilih Buat.
7. Hapus kode sampel YAMAL.
8. Tambahkan YAML berikut:

```
Name: codecatalyst-linter-workflow
SchemaVersion: "1.0"
Triggers:
  - Type: PUSH
    Branches:
      - main
Actions:
  SuperLinterAction:
    Identifier: aws/github-actions-runner@v1
    Configuration:
      Steps:
        github-action-code
```

   Dalam kode sebelumnya, ganti *`github-action-code`*dengan kode tindakan Super-Linter, seperti yang diinstruksikan dalam langkah-langkah berikut dari prosedur ini.
9. Buka halaman Super-Linter di Marketplace GitHub .
10. Di bawah `steps:` (huruf kecil), temukan kode dan tempelkan ke CodeCatalyst alur kerja di bawah `Steps:` (huruf besar).

    Sesuaikan kode GitHub Action agar sesuai dengan CodeCatalyst standar, seperti yang ditunjukkan pada kode berikut.

    CodeCatalyst Alur kerja Anda sekarang terlihat seperti ini:

```
Name: codecatalyst-linter-workflow
SchemaVersion: "1.0"
Triggers:
  - Type: PUSH
    Branches:
      - main
Actions:
  SuperLinterAction:
```

```
    Identifier: aws/github-actions-runner@v1
    Configuration:
      Steps:
        - name: Lint Code Base
          uses: github/super-linter@v4
          env:
            VALIDATE_ALL_CODEBASE: "true"
            DEFAULT_BRANCH: main
```

11. (Opsional) Pilih Validasi untuk memastikan kode YAMAL valid sebelum melakukan.
12. Pilih Komit, masukkan pesan Komit, pilih `codecatalyst-linter-source-repository` Repositori Anda, dan pilih Komit lagi.

    Anda sekarang telah membuat alur kerja. Jalankan alur kerja dimulai secara otomatis karena pemicu yang ditentukan di bagian atas alur kerja.

Untuk melihat alur kerja yang sedang berjalan

1. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih alur kerja yang baru saja Anda buat:`codecatalyst-linter-workflow`.
3. Dalam diagram alur kerja, pilih SuperLinterAction.
4. Tunggu sampai tindakan gagal. Kegagalan ini diharapkan karena linter menemukan masalah dalam kode.
5. Biarkan CodeCatalyst konsol terbuka dan pergi keLangkah 4: Perbaiki masalah yang ditemukan Super-Linter.

Langkah 4: Perbaiki masalah yang ditemukan Super-Linter

Super-Linter seharusnya menemukan masalah dalam `app.js` kode, serta `README.md` file yang disertakan dalam repositori sumber Anda.

Untuk memperbaiki masalah linter ditemukan

1. Di CodeCatalyst konsol, pilih tab Log, lalu pilih Lint Code Base.

   Log yang dihasilkan tindakan Super-Linter ditampilkan.
2. Di log Super-Linter, gulir ke bawah ke sekitar baris 90, di mana Anda menemukan awal masalah. Mereka terlihat mirip dengan yang berikut:

```
/github/workspace/hello-world/app.js:3:13: Extra semicolon.
/github/workspace/hello-world/app.js:9:92: Trailing spaces not allowed.
/github/workspace/hello-world/app.js:21:7: Unnecessarily quoted property 'body'
 found.
/github/workspace/hello-world/app.js:31:1: Expected indentation of 2 spaces but
 found 4.
/github/workspace/hello-world/app.js:32:2: Newline required at end of file but not
 found.
```

3. Perbaiki `app.js` dan `README.md` di repositori sumber Anda dan komit perubahan Anda.

> **Tip**
>
> Untuk memperbaikinya`README.md`, tambahkan `markdown` ke blok kode, seperti ini:
>
> ```
> ```markdown
> Setup examples:
> ...
> ```
> ```

Perubahan Anda memulai alur kerja lain berjalan secara otomatis. Tunggu alur kerja selesai. Jika Anda memperbaiki semua masalah, alur kerja harus berhasil.

Bersihkan

Bersihkan CodeCatalyst untuk menghapus jejak tutorial ini dari lingkungan Anda.

Untuk membersihkan CodeCatalyst

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Hapus`codecatalyst-linter-source-repository`.
3. Hapus`codecatalyst-linter-workflow`.

Dalam tutorial ini, Anda belajar cara menambahkan Super-Linter GitHub Action ke CodeCatalyst alur kerja untuk lint beberapa kode.

Menambahkan GitHub tindakan "Tindakan”

GitHub Tindakan Tindakan adalah CodeCatalyst tindakan yang membungkus GitHub Action dan membuatnya kompatibel dengan CodeCatalyst alur kerja.

Untuk informasi selengkapnya, lihat Mengintegrasikan GitHub Tindakan ke dalam alur kerja.

Untuk menambahkan GitHub tindakan Tindakan ke alur kerja, ikuti langkah-langkah ini.

> **Tip**
>
> Untuk tutorial yang menunjukkan cara menggunakan GitHub tindakan Actions, lihatTutorial: Kode lint menggunakan GitHub Action dalam alur kerja.

Visual

Untuk menambahkan GitHub tindakan "Tindakan” menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih GitHub.
9. Cari GitHub tindakan Tindakan, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih GitHub Tindakan. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Lihat sumber untuk melihat kode sumber tindakan.

- Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

10. Pada tab Input dan Konfigurasi, lengkapi bidang sesuai dengan kebutuhan Anda. Untuk deskripsi setiap bidang, lihatGitHub Tindakan “Tindakan” definisi YAMAL. Referensi ini memberikan informasi rinci tentang setiap bidang (dan nilai properti YAMAL yang sesuai) seperti yang muncul di YAMAL dan editor visual.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menambahkan GitHub tindakan "Tindakan” menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih GitHub.
9. Cari GitHub tindakan Tindakan, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih GitHub Tindakan. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Lihat sumber untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Ubah properti dalam kode YAMAL sesuai dengan kebutuhan Anda. Penjelasan tentang setiap properti yang tersedia disediakan diGitHub Tindakan “Tindakan” definisi YAMAL.

11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Definisi GitHub tindakan “Tindakan”

Tindakan GitHub Tindakan didefinisikan sebagai sekumpulan properti YAMAL di dalam file definisi alur kerja Anda. Untuk informasi tentang properti ini, lihat GitHub Tindakan “Tindakan” definisi YAMAL diAlur kerja definisi YAMAL.

Menambahkan Action yang dikuratori GitHub

GitHub Action curated adalah GitHub Action yang tersedia di CodeCatalyst konsol, dan berfungsi sebagai contoh cara menggunakan GitHub Action di dalam CodeCatalyst alur kerja.

GitHub Tindakan yang Dikurasi dibungkus dalam tindakan Tindakan CodeCatalyst yang ditulisGitHub , diidentifikasi oleh pengenal. `aws/github-actions-runner@v1` Misalnya, inilah tampilan GitHub Action yang dikuratori, TruffleHog OSS,:

```
Actions:
  TruffleHogOSS_e8:
    Identifier: aws/github-actions-runner@v1
    Inputs:
      Sources:
        - WorkflowSource # This specifies that the action requires this Workflow as a
 source
    Configuration:
      Steps:
        - uses: trufflesecurity/trufflehog@v3.16.0
          with:
            path: ' ' # Required; description: Repository path
            base: ' ' # Required; description: Start scanning from here (usually main
 branch).
            head: ' ' # Optional; description: Scan commits until here (usually dev
 branch).
            extra_args: ' ' # Optional; description: Extra args to be passed to the
 trufflehog cli.
```

Dalam kode sebelumnya, CodeCatalyst GitHub tindakan Tindakan (diidentifikasi oleh`aws/github-actions-runner@v1`) membungkus tindakan TruffleHog OSS (diidentifikasi oleh`trufflesecurity/trufflehog@v3.16.0`), membuatnya bekerja dalam alur kerja. CodeCatalyst

Untuk mengonfigurasi tindakan ini, Anda akan mengganti string kosong di bawah `with:` dengan nilai Anda sendiri. Misalnya:

```
Actions:
  TruffleHogOSS_e8:
    Identifier: aws/github-actions-runner@v1
    Inputs:
      Sources:
        - WorkflowSource # This specifies that the action requires this Workflow as a
 source
    Configuration:
      Steps:
        - uses: trufflesecurity/trufflehog@v3.16.0
          with:
            path: ./
            base: main # Required; description: Start scanning from here (usually main
 branch).
            head: HEAD # Optional; description: Scan commits until here (usually dev
 branch).
            extra_args: '--debug --only-verified' # Optional; description: Extra args
 to be passed to the trufflehog cli.
```

Untuk menambahkan GitHub Tindakan yang dikurasi ke alur kerja, gunakan prosedur berikut. Untuk informasi umum tentang menggunakan GitHub Tindakan dalam CodeCatalyst alur kerja, lihatMengintegrasikan GitHub Tindakan ke dalam alur kerja.

> **Note**
>
> Jika Anda tidak melihat GitHub Tindakan di antara daftar tindakan yang dikurasi, Anda masih dapat menambahkannya ke alur kerja menggunakan GitHub tindakan Tindakan. Untuk informasi selengkapnya, lihat Menambahkan GitHub tindakan "Tindakan”.

Visual

Untuk menambahkan GitHub tindakan yang dikuratori menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.

4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih GitHub.
9. Jelajahi atau cari GitHub Action, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih nama GitHub Action. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Lihat sumber untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Pada tab Input, Konfigurasi, dan Output, lengkapi bidang sesuai dengan kebutuhan Anda. Untuk deskripsi setiap bidang, lihatGitHub Tindakan “Tindakan” definisi YAMAL. Referensi ini memberikan informasi rinci tentang setiap bidang (dan nilai properti YAMAL terkait) yang tersedia untuk GitHubtindakan Tindakan, seperti yang muncul di editor YAMAL dan visual.

    Untuk informasi tentang opsi konfigurasi yang tersedia untuk GitHub Tindakan yang dikurasi, lihat dokumentasinya.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menambahkan GitHub tindakan yang dikuratori menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.

5. Pilih Edit.
6. Pilih YAMAL.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih GitHub.
9. Jelajahi atau cari GitHub Action, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih nama GitHub Action. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Lihat sumber untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Ubah properti dalam kode YAMAL sesuai dengan kebutuhan Anda. Penjelasan tentang setiap properti yang tersedia untuk GitHub tindakan Tindakan disediakan diGitHub Tindakan "Tindakan" definisi YAMAL.

    Untuk informasi tentang opsi konfigurasi yang tersedia untuk GitHub Tindakan yang dikurasi, lihat dokumentasinya.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Mengekspor parameter GitHub output sehingga tindakan lain dapat menggunakannya

Anda dapat menggunakan parameter GitHub output dalam CodeCatalyst alur kerja Anda.

> **Note**
>
> Kata lain untuk parameter output adalah variabel. Karena GitHub menggunakan parameter keluaran istilah dalam dokumentasinya, kita akan menggunakan istilah ini juga.

Gunakan petunjuk berikut untuk mengekspor parameter GitHub keluaran dari GitHub Action sehingga tersedia untuk digunakan oleh tindakan CodeCatalyst alur kerja lainnya.

Untuk mengekspor parameter GitHub output

1. Buka alur kerja dan pilih Edit. Untuk informasi selengkapnya, lihat Membuat alur kerja.
2. Dalam GitHub tindakan Tindakan yang menghasilkan parameter keluaran yang ingin Anda ekspor, tambahkan `Outputs` bagian dengan `Variables` properti dasar yang terlihat seperti ini:

```
Actions:
  MyGitHubAction:
    Identifier: aws/github-actions-runner@v1
    Outputs:
      Variables:
        - 'step-id_output-name'
```

Ganti:

- *step-id* dengan nilai `id:` properti di `steps` bagian GitHub tindakan.
- *output-name* dengan nama parameter output. GitHub

Contoh

Contoh berikut menunjukkan cara mengekspor parameter GitHub output yang disebut`SELECTEDCOLOR`.

```
Actions:
  MyGitHubAction:
    Identifier: aws/github-actions-runner@v1
    Outputs:
      Variables:
        - 'random-color-generator_SELECTEDCOLOR'
    Configuration:
      Steps:
        - name: Set selected color
          run: echo "SELECTEDCOLOR=green" >> $GITHUB_OUTPUT
          id: random-color-generator
```

Mereferensikan parameter GitHub keluaran

Gunakan petunjuk berikut untuk referensi parameter GitHub output.

Untuk mereferensikan parameter GitHub output

1. Selesaikan langkah-langkah dalam Mengekspor parameter GitHub output sehingga tindakan lain dapat menggunakannya.

   Parameter GitHub output sekarang tersedia untuk digunakan dalam tindakan lain.
2. Perhatikan `Variables` nilai parameter output. Ini termasuk garis bawah (_).
3. Lihat parameter output menggunakan sintaks berikut:

   ```
   ${action-name.output-name}
   ```

   Ganti:

   - *action-name* dengan nama CodeCatalyst GitHub Action yang menghasilkan parameter output (jangan gunakan GitHub action `name` atau`uid`).
   - *output-name* dengan `Variables` nilai parameter output yang Anda catat sebelumnya.

   Contoh

   ```
   BuildActionB:
     Identifier: aws/build@v1
     Configuration:
       Steps:
         - Run: echo ${MyGitHubAction.random-color-generator_SELECTEDCOLOR}
   ```

   Contoh dengan konteks

   Contoh berikut menunjukkan kepada Anda bagaimana mengatur `SELECTEDCOLOR` variabel dalam`GitHubActionA`, output itu, dan kemudian merujuk ke dalam`BuildActionB`.

   ```
   Actions:
     GitHubActionA:
       Identifier: aws/github-actions-runner@v1
       Configuration:
         Steps:
           - name: Set selected color
             run: echo "SELECTEDCOLOR=green" >> $GITHUB_OUTPUT
             id: random-color-generator
   ```

```
    Outputs:
      Variables:
      - 'random-color-generator_SELECTEDCOLOR'

   BuildActionB:
    Identifier: aws/build@v1
    Configuration:
      Steps:
        - Run: echo ${GitHubActionA.random-color-generator_SELECTEDCOLOR}
```

GitHub Tindakan "Tindakan" definisi YAMAL

Berikut ini adalah definisi YAMB dari GitHubtindakan Tindakan.

Definisi tindakan ini ada sebagai bagian dalam file definisi alur kerja yang lebih luas. Untuk informasi selengkapnya tentang file ini, lihatAlur kerja definisi YAMAL.

Pilih properti YAMB dalam kode berikut untuk melihat deskripsi jika itu.

> **Note**
>
> Sebagian besar properti YAMB yang mengikuti memiliki elemen UI yang sesuai di editor visual. Untuk mencari elemen UI, gunakan Ctrl+F. Elemen akan terdaftar dengan properti YAMLnya yang terkait.

```
# The workflow definition starts here.
# See Properti tingkat atas for details.

Name: MyWorkflow
SchemaVersion: 1.0
Actions:

# The action definition starts here.
  action-name:
    Identifier:  aws/github-actions-runner@v1
    DependsOn:
      - dependent-action-name-1
    Compute:
      Fleet: fleet-name
    Timeout: timeout-minutes
```

```
    Environment:
      Name: environment-name
      Connections:
        - Name: account-connection-name
          Role: iam-role-name
    Inputs:
      Sources:
        - source-name-1
        - source-name-2
      Artifacts:
        - artifact-name
      Variables:
        - Name: variable-name-1
          Value: variable-value-1
        - Name: variable-name-2
          Value: variable-value-2
    Outputs:
      Artifacts:
        - Name: output-artifact-1
          Files:
            - github-output/artifact-1.jar
            - "github-output/build*"
        - Name: output-artifact-2
          Files:
            - github-output/artifact-2.1.jar
            - github-output/artifact-2.2.jar
      Variables:
        - variable-name-1
        - variable-name-2
      AutoDiscoverReports:
        Enabled: true | false
        ReportNamePrefix: AutoDiscovered
        IncludePaths:
          - "**/*"
        ExcludePaths:
          - node_modules/cdk/junit.xml
        SuccessCriteria:
          PassRate: percent
          LineCoverage: percent
          BranchCoverage: percent
          Vulnerabilities:
            Severity: CRITICAL|HIGH|MEDIUM|LOW|INFORMATIONAL
            Number: whole-number
      Reports:
```

```
      report-name-1:
        Format: format
        IncludePaths:
          - "*.xml"
        ExcludePaths:
          - report2.xml
          - report3.xml
        SuccessCriteria:
          PassRate: percent
          LineCoverage: percent
          BranchCoverage: percent
          Vulnerabilities:
            Severity: CRITICAL|HIGH|MEDIUM|LOW|INFORMATIONAL
            Number: whole-number
  Configuration
    Steps:
      - github-actions-code
```

nama-tindakan

(Diperlukan)

Tentukan nama tindakan. Semua nama tindakan harus unik dalam alur kerja. Nama aksi terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama tindakan.

UI yang sesuai: Tab konfigurasi/nama tindakan

Identifier

(*nama-tindakan/*) Identifier

Mengidentifikasi tindakan. Jangan mengubah properti ini kecuali Anda ingin mengubah versi. Untuk informasi selengkapnya, lihat Menentukan versi mayor, minor, atau patch dari suatu tindakan.

Gunakan `aws/github-actions-runner@v1` untuk GitHubtindakan tindakan.

UI yang sesuai: Diagram alur kerja/nama ***tindakan/label*** aws/ @v1 github-actions-runner

DependsOn

(*nama-tindakan/*) DependsOn

(Opsional)

Tentukan tindakan, grup tindakan, atau gerbang yang harus berjalan dengan sukses agar tindakan ini berjalan.

Untuk informasi selengkapnya tentang fungsionalitas 'tergantung pada', lihat. Mengkonfigurasi tindakan untuk bergantung pada tindakan lain

UI yang sesuai: Tab masukan/Tergantung pada - opsional

Compute

(*nama-tindakan/*) Compute

(Opsional)

Mesin komputasi yang digunakan untuk menjalankan tindakan alur kerja Anda. Anda dapat menentukan komputasi baik di tingkat alur kerja atau di tingkat tindakan, tetapi tidak keduanya. Ketika ditentukan pada tingkat alur kerja, konfigurasi komputasi berlaku untuk semua tindakan yang ditentukan dalam alur kerja. Pada tingkat alur kerja, Anda juga dapat menjalankan beberapa tindakan pada instance yang sama. Untuk informasi selengkapnya, lihat Berbagi komputasi di seluruh tindakan.

UI yang sesuai: tidak ada

Fleet

(*nama-tindakan) /Compute/* ***Fleet***

(Opsional)

Tentukan mesin atau armada yang akan menjalankan alur kerja atau tindakan alur kerja Anda. Dengan armada sesuai permintaan, ketika suatu tindakan dimulai, alur kerja menyediakan sumber daya yang dibutuhkan, dan mesin dihancurkan ketika tindakan selesai. Contoh armada sesuai permintaan:`Linux.x86-64.Large`,. `Linux.x86-64.XLarge` Untuk informasi lebih lanjut tentang armada sesuai permintaan, lihat. Properti armada sesuai permintaan

Dengan armada yang disediakan, Anda mengonfigurasi satu set mesin khusus untuk menjalankan tindakan alur kerja Anda. Mesin-mesin ini tetap menganggur, siap untuk memproses tindakan segera. Untuk informasi lebih lanjut tentang armada yang disediakan, lihat. Properti armada yang disediakan

Jika `Fleet` dihilangkan, defaultnya adalah. `Linux.x86-64.Large`

UI yang sesuai: Tab konfigurasi/Armada komputasi - opsional

Timeout

(*nama-tindakan/*) Timeout

(Opsional)

Tentukan jumlah waktu dalam menit (editor YAMAL), atau jam dan menit (editor visual), bahwa tindakan dapat berjalan sebelum CodeCatalyst mengakhiri tindakan. Minimal adalah 5 menit dan maksimum dijelaskan dalamKuota untuk alur kerja. Batas waktu default sama dengan batas waktu maksimum.

UI yang sesuai: Tab konfigurasi/Timeout - opsional

Environment

(*nama-tindakan/*) Environment

(Opsional)

Tentukan CodeCatalyst lingkungan yang akan digunakan dengan tindakan.

Untuk informasi lebih lanjut tentang lingkungan, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst danPembuatan lingkungan.

UI yang sesuai: Tab konfigurasi/lingkungan/akun/peran

Name

(*nama-tindakan) /Environment/* ***Name***

(Diperlukan Environment jika disertakan)

Tentukan nama lingkungan yang ada yang ingin Anda kaitkan dengan tindakan.

UI yang sesuai: Tab Konfigurasi/'Lingkungan/Akun/Peran'/Lingkungan

Connections

(*nama-tindakan) /Environment/* ***Connections***

(Diperlukan Environment jika disertakan)

Tentukan koneksi akun untuk dikaitkan dengan tindakan. Anda dapat menentukan maksimum satu koneksi akun di bawah`Environment`.

Untuk informasi selengkapnya tentang koneksi akun, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS. Untuk informasi tentang cara mengaitkan koneksi akun dengan lingkungan Anda, lihatPembuatan lingkungan.

UI yang sesuai: tidak ada

Name

(nama-tindakan) /Environment/Connections/ ***Name***

(Opsional)

Tentukan nama koneksi akun.

UI yang sesuai: Tab konfigurasi/'lingkungan/akun/peran'/koneksi akun AWS

Role

(nama-tindakan) /Environment/Connections/ ***Role***

(Opsional)

Tentukan nama peran IAM yang digunakan tindakan ini untuk mengakses dan beroperasi di AWS layanan seperti Amazon S3 dan Amazon ECR. Pastikan peran ini ditambahkan ke koneksi akun Anda. Untuk menambahkan peran IAM ke koneksi akun, lihatMenambahkan peran IAM ke koneksi akun.

> **Note**
>
> Anda mungkin dapat menentukan nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran di sini, asalkan memiliki izin yang cukup. Untuk informasi selengkapnya tentang peran ini, silakan lihat Membuat CodeCatalystWorkflowDevelopmentRole-***`spaceName`***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami menyarankan Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian.

> **Warning**
>
> Batasi izin untuk yang diperlukan oleh GitHub tindakan Tindakan. Menggunakan peran dengan izin yang lebih luas dapat menimbulkan risiko keamanan.

UI yang sesuai: Tab Konfigurasi/'lingkungan/akun/peran'/ Peran

Inputs

(*nama-tindakan/*) Inputs

(Opsional)

`Inputs`Bagian ini mendefinisikan data yang dibutuhkan tindakan selama menjalankan alur kerja.

> **Note**
>
> Maksimal empat input (satu sumber dan tiga artefak) diperbolehkan per GitHub tindakan Tindakan. Variabel tidak dihitung terhadap total ini.

Jika Anda perlu merujuk ke file yang berada di input yang berbeda (katakanlah sumber dan artefak), input sumber adalah input utama, dan artefak adalah input sekunder. Referensi ke file dalam input sekunder mengambil awalan khusus untuk menyisihkannya dari primer. Lihat perinciannya di Contoh: Merujuk file dalam beberapa artefak.

UI yang sesuai: Tab input

Sources

(*nama-tindakan) /Inputs/* ***Sources***

(Opsional)

Tentukan label yang mewakili repositori sumber yang akan dibutuhkan oleh tindakan. Saat ini, satu-satunya label yang didukung adalah`WorkflowSource`, yang mewakili repositori sumber tempat file definisi alur kerja Anda disimpan.

Jika Anda menghilangkan sumber, maka Anda harus menentukan setidaknya satu artefak input di bawah. *action-name*`/Inputs/Artifacts`

Untuk informasi selengkapnya tentang sumber, lihat Menghubungkan alur kerja ke repositori sumber.

UI yang sesuai: Tab/Sumber Input - opsional

Artifacts - input

(nama-tindakan) /Inputs/ ***Artifacts***

(Opsional)

Tentukan artefak dari tindakan sebelumnya yang ingin Anda berikan sebagai masukan untuk tindakan ini. Artefak ini harus sudah didefinisikan sebagai artefak keluaran dalam tindakan sebelumnya.

Jika Anda tidak menentukan artefak input apa pun, maka Anda harus menentukan setidaknya satu repositori sumber di bawah. *action-name*`/Inputs/Sources`

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

> **Note**
>
> Jika Artefacts - daftar drop-down opsional tidak tersedia (editor visual), atau jika Anda mendapatkan kesalahan saat memvalidasi YAMAL (editor YAMAL), itu mungkin karena tindakan hanya mendukung satu input. Dalam hal ini, coba hapus input sumber.

UI yang sesuai: Input tab/Artefak - opsional

Variables - input

(nama-tindakan) /Inputs/ ***Variables***

(Opsional)

Tentukan urutan pasangan nama/nilai yang menentukan variabel input yang ingin Anda sediakan untuk tindakan. Nama variabel terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama variabel.

Untuk informasi selengkapnya tentang variabel, termasuk contoh, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

UI yang sesuai: Input tab/Variabel - opsional

Outputs

(*nama-tindakan/*) Outputs

(Opsional)

Mendefinisikan data yang dihasilkan oleh tindakan selama menjalankan alur kerja.

UI yang sesuai: Tab keluaran

Artifacts - output

(*nama-tindakan) /Outputs/* ***Artifacts***

(Opsional)

Tentukan nama artefak yang dihasilkan oleh tindakan. Nama artifak harus unik dalam alur kerja, dan terbatas pada karakter alfanumerik (a-z, A-Z, 0-9) dan garis bawah (_). Spasi, tanda hubung (-), dan karakter khusus lainnya tidak diperbolehkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan spasi, tanda hubung, dan karakter khusus lainnya dalam nama artefak keluaran.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab keluaran/Artefak

Name

(*nama-tindakan) /Outputs/Artifacts/* ***Name***

(Diperlukan Artifacts - output jika disertakan)

Tentukan nama artefak yang dihasilkan oleh tindakan. Nama artifak harus unik dalam alur kerja, dan terbatas pada karakter alfanumerik (a-z, A-Z, 0-9) dan garis bawah (_). Spasi, tanda hubung (-), dan karakter khusus lainnya tidak diperbolehkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan spasi, tanda hubung, dan karakter khusus lainnya dalam nama artefak keluaran.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab Keluaran/Artefak/Tambahkan artefak/Bangun nama artefak

Files

(nama-tindakan) /Outputs/Artifacts/ ***Files***

(Diperlukan Artifacts - output jika disertakan)

Tentukan file yang CodeCatalyst termasuk dalam artefak yang dihasilkan oleh tindakan. File-file ini dihasilkan oleh tindakan alur kerja saat dijalankan, dan juga tersedia di repositori sumber Anda. Jalur file dapat berada di repositori sumber atau artefak dari tindakan sebelumnya, dan relatif terhadap repositori sumber atau root artefak. Anda dapat menggunakan pola glob untuk menentukan jalur. Contoh:

- Untuk menentukan satu file yang ada di root lokasi build atau lokasi repositori sumber, gunakan. `my-file.jar`
- Untuk menentukan satu file dalam subdirektori, gunakan `directory/my-file.jar` atau`directory/subdirectory/my-file.jar`.
- Untuk menentukan semua file, gunakan`"**/*"`. Pola `**` glob menunjukkan untuk mencocokkan sejumlah subdirektori.
- Untuk menentukan semua file dan direktori dalam direktori bernama`directory`, gunakan`"directory/**/*"`. Pola `**` glob menunjukkan untuk mencocokkan sejumlah subdirektori.
- Untuk menentukan semua file dalam direktori bernama`directory`, tetapi tidak salah satu subdirektorinya, gunakan. `"directory/*"`

> **Note**
>
> Jika jalur file Anda menyertakan satu atau beberapa tanda bintang (*) atau karakter khusus lainnya, lampirkan jalur dengan tanda kutip ganda (). "" Untuk informasi lebih lanjut tentang karakter khusus, lihatPedoman dan konvensi sintaks.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

> **Note**
>
> Anda mungkin perlu menambahkan awalan ke jalur file untuk menunjukkan artefak atau sumber mana yang akan menemukannya. Untuk informasi selengkapnya, lihat Mereferensikan file dalam repositori sumber dan Merujuk file dalam artefak.

UI yang sesuai: Tab Keluaran/Artefak/Tambahkan artefak/File yang dihasilkan oleh build

Variables - output

(nama-tindakan) /Outputs/ ***Variables***

(Opsional)

Tentukan variabel yang ingin Anda ekspor tindakan sehingga tersedia untuk digunakan oleh tindakan selanjutnya.

Untuk informasi selengkapnya tentang variabel, termasuk contoh, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

UI yang sesuai: Tab keluaran/Variabel/Tambahkan variabel

variabel-nama-1

(***nama-aksi-variabel-nama-1***/Outputs/Variables)

(Opsional)

Tentukan nama variabel yang ingin Anda ekspor tindakan. Variabel ini harus sudah didefinisikan dalam `Inputs` atau `Steps` bagian dari tindakan yang sama.

Untuk informasi selengkapnya tentang variabel, termasuk contoh, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

UI yang sesuai: Tab keluaran/variabel/Tambahkan variabel/Nama

AutoDiscoverReports

(nama-tindakan) /Outputs/ ***AutoDiscoverReports***

(Opsional)

Mendefinisikan konfigurasi untuk fitur auto-discovery.

Saat Anda mengaktifkan penemuan otomatis, semua CodeCatalyst pencarian `Inputs` diteruskan ke tindakan serta semua file yang dihasilkan oleh tindakan itu sendiri, mencari laporan pengujian, cakupan kode, dan analisis komposisi perangkat lunak (SCA). Untuk setiap laporan yang ditemukan, CodeCatalyst mengubahnya menjadi CodeCatalyst laporan. CodeCatalyst Laporan adalah laporan yang sepenuhnya terintegrasi ke dalam CodeCatalyst layanan dan dapat dilihat dan dimanipulasi melalui CodeCatalyst konsol.

> **Note**
>
> Secara default, fitur auto-discover memeriksa semua file. Anda dapat membatasi file mana yang diperiksa menggunakan ExcludePaths properti IncludePaths atau.

UI yang sesuai: tidak ada

Enabled

(nama-tindakan) /Outputs/AutoDiscoverReports/ ***Enabled***

(Opsional)

Aktifkan atau nonaktifkan fitur penemuan otomatis.

Nilai-nilai yang valid adalah `true` atau `false`.

Jika `Enabled` dihilangkan, defaultnya adalah. `true`

UI yang sesuai: Tab Keluaran/Laporan/Secara otomatis menemukan laporan

ReportNamePrefix

(nama-tindakan) /Outputs/AutoDiscoverReports/ ***ReportNamePrefix***

(Diperlukan AutoDiscoverReports jika disertakan dan diaktifkan)

Tentukan CodeCatalyst awalan yang ditambahkan ke semua laporan yang ditemukannya untuk memberi nama laporan terkait. CodeCatalyst Misalnya, jika Anda menentukan awalan`AutoDiscovered`, dan CodeCatalyst otomatis menemukan dua laporan pengujian,

`TestSuiteOne.xml` dan`TestSuiteTwo.xml`, maka CodeCatalyst laporan terkait akan diberi nama dan. `AutoDiscoveredTestSuiteOne` `AutoDiscoveredTestSuiteTwo`

UI yang sesuai: Tab Keluaran/Laporan/Secara otomatis menemukan laporan/Awalan laporan

IncludePaths

(nama-tindakan) /Outputs/AutoDiscoverReports/ ***IncludePaths***

Atau

(aksi-nama-laporan-nama-1 /Outputs/Reports//) ***IncludePaths***

(Diperlukan jika AutoDiscoverReports disertakan dan diaktifkan, atau Reports jika disertakan)

Tentukan file dan jalur file yang CodeCatalyst disertakan saat mencari laporan mentah. Misalnya, jika Anda menentukan`"/test/report/*"`, akan CodeCatalyst mencari seluruh image build yang digunakan oleh tindakan mencari `/test/report/*` direktori. Ketika menemukan direktori itu, CodeCatalyst maka cari laporan di direktori itu.

> **Note**
>
> Jika jalur file Anda menyertakan satu atau beberapa tanda bintang (`*`) atau karakter khusus lainnya, lampirkan jalur dengan tanda kutip ganda (). `""` Untuk informasi lebih lanjut tentang karakter khusus, lihatPedoman dan konvensi sintaks.

Jika properti ini dihilangkan, defaultnya adalah`"**/*"`, artinya pencarian mencakup semua file di semua jalur.

> **Note**
>
> Untuk laporan yang dikonfigurasi secara manual, `IncludePaths` harus berupa pola glob yang cocok dengan satu file.

UI yang sesuai:

- Tab keluaran/Laporan/Secara otomatis menemukan laporan/'Sertakan/Kecualikan jalur'/ Sertakan jalur

- ***Tab keluaran/Laporan/Konfigurasikan laporan/laporan-nama-1 /'Sertakan/kecualikan jalur'/ Sertakan jalur***

ExcludePaths

(nama-tindakan) /Outputs/AutoDiscoverReports/ ***ExcludePaths***

Atau

(aksi-nama-laporan-nama-1 /Outputs/Reports//) ***ExcludePaths***

(Opsional)

Tentukan file dan jalur file yang CodeCatalyst dikecualikan saat mencari laporan mentah. Misalnya, jika Anda menentukan`"/test/my-reports/**/*"`, tidak CodeCatalyst akan mencari file di `/test/my-reports/` direktori. Untuk mengabaikan semua file dalam direktori, gunakan pola `**/*` glob.

> **Note**
>
> Jika jalur file Anda menyertakan satu atau beberapa tanda bintang (*) atau karakter khusus lainnya, lampirkan jalur dengan tanda kutip ganda (). "" Untuk informasi lebih lanjut tentang karakter khusus, lihatPedoman dan konvensi sintaks.

UI yang sesuai:

- Tab keluaran/Laporan/Secara otomatis menemukan laporan/'Sertakan/Kecualikan jalur'/Kecualikan jalur
- ***Tab keluaran/Laporan/Konfigurasikan laporan/laporan-nama-1 /'Sertakan/kecualikan jalur'/ Kecualikan jalur secara manual***

SuccessCriteria

(nama-tindakan) /Outputs/AutoDiscoverReports/ ***SuccessCriteria***

Atau

(aksi-nama-laporan-nama-1 /Outputs/Reports//) ***SuccessCriteria***

(Opsional)

Tentukan kriteria keberhasilan untuk pengujian, cakupan kode, analisis komposisi perangkat lunak (SCA), dan laporan analisis statis (SA).

Untuk informasi selengkapnya, lihat Mengkonfigurasi kriteria keberhasilan untuk laporan.

UI yang sesuai:

- Tab Keluaran/Laporan/Secara otomatis menemukan laporan/Kriteria sukses
- ***Tab keluaran/Laporan/Konfigurasi laporan/nama-laporan-1/Kriteria keberhasilan secara manual***

PassRate

(nama-tindakan) /Outputs/AutoDiscoverReports/SuccessCriteria/ ***PassRate***

Atau

(aksi-nama-laporan-nama-1/Outputs/Reports/) /SuccessCriteria/ ***PassRate***

(Opsional)

Tentukan persentase pengujian dalam laporan pengujian yang harus lulus agar CodeCatalyst laporan terkait ditandai sebagai lulus. Nilai yang valid termasuk angka desimal. Misalnya:`50`,`60.5`. Kriteria tingkat kelulusan hanya diterapkan pada laporan pengujian. Untuk informasi selengkapnya tentang laporan pengujian, lihatLaporan pengujian.

UI yang sesuai:

- Tab Keluaran/Laporan/Secara otomatis menemukan laporan/Kriteria sukses/Tingkat kelulusan
- ***Tab keluaran/Laporan/Konfigurasi laporan/nama-laporan-1/Kriteria sukses/Tingkat kelulusan secara manual***

LineCoverage

(nama-tindakan) /Outputs/AutoDiscoverReports/SuccessCriteria/ ***LineCoverage***

Atau

(aksi-nama-laporan-nama-1/Outputs/Reports/) /SuccessCriteria/ ***LineCoverage***

(Opsional)

Tentukan persentase baris dalam laporan cakupan kode yang harus dicakup agar CodeCatalyst laporan terkait ditandai sebagai lulus. Nilai yang valid termasuk angka desimal. Misalnya:`50`,`60.5`. Kriteria cakupan baris hanya diterapkan pada laporan cakupan kode. Untuk informasi selengkapnya tentang laporan cakupan kode, lihatLaporan cakupan kode.

UI yang sesuai:

- Tab Keluaran/Laporan/Secara otomatis menemukan laporan/Kriteria sukses/Cakupan garis
- ***Tab keluaran/Laporan/Konfigurasi laporan/nama-laporan-1/Kriteria sukses/Cakupan baris secara manual***

BranchCoverage

(nama-tindakan) /Outputs/AutoDiscoverReports/SuccessCriteria/ ***BranchCoverage***

Atau

(aksi-nama-laporan-nama-1/Outputs/Reports/) /SuccessCriteria/ ***BranchCoverage***

(Opsional)

Tentukan persentase cabang dalam laporan cakupan kode yang harus dicakup agar CodeCatalyst laporan terkait ditandai sebagai lulus. Nilai yang valid termasuk angka desimal. Misalnya:`50`,`60.5`. Kriteria cakupan cabang hanya diterapkan pada laporan cakupan kode. Untuk informasi selengkapnya tentang laporan cakupan kode, lihatLaporan cakupan kode.

UI yang sesuai:

- Tab Keluaran/Laporan/Secara otomatis menemukan laporan/Kriteria sukses/Cakupan cabang
- ***Tab keluaran/Laporan/Konfigurasi laporan/nama-laporan-1/Kriteria sukses/Cakupan cabang secara manual***

Vulnerabilities

(nama-tindakan) /Outputs/AutoDiscoverReports/SuccessCriteria/ ***Vulnerabilities***

Atau

(aksi-nama-laporan-nama-1/Outputs/Reports/) /SuccessCriteria/ ***Vulnerabilities***

(Opsional)

Tentukan jumlah maksimum dan tingkat keparahan kerentanan yang diizinkan dalam laporan SCA agar CodeCatalyst laporan terkait ditandai sebagai lulus. Untuk menentukan kerentanan, Anda harus menentukan:

- Tingkat keparahan minimum kerentanan yang ingin Anda sertakan dalam hitungan. Nilai yang valid, dari yang paling parah hingga yang paling parah, adalah: `CRITICALHIGH`,`MEDIUM`,,`LOW`,`INFORMATIONAL`.

  Misalnya, jika Anda memilih`HIGH`, maka `HIGH` dan `CRITICAL` kerentanan akan dihitung.
- Jumlah maksimum kerentanan dari tingkat keparahan yang ditentukan yang Anda inginkan memungkinkan. Melebihi angka ini menyebabkan CodeCatalyst laporan ditandai sebagai gagal. Nilai yang valid adalah bilangan bulat.

Kriteria kerentanan hanya diterapkan pada laporan SCA. Untuk informasi lebih lanjut tentang laporan SCA, lihatLaporan analisis komposisi perangkat lunak.

Untuk menentukan tingkat keparahan minimum, gunakan `Severity` properti. Untuk menentukan jumlah kerentanan maksimum, gunakan `Number` properti.

Untuk informasi lebih lanjut tentang laporan SCA, lihatJenis laporan kualitas.

UI yang sesuai:

- Tab Keluaran/Laporan/Secara otomatis menemukan laporan/Kriteria sukses/Kerentanan
- ***Tab keluaran/Laporan/Konfigurasi laporan/nama-laporan-1/Kriteria sukses/ Kerentanan secara manual***

Reports

(nama-tindakan) /Outputs/ ***Reports***

(Opsional)

Bagian yang menentukan konfigurasi untuk laporan pengujian.

UI yang sesuai: Tab keluaran/Laporan

laporan-nama-1

(***aksi-nama-laporan-nama-1***/Outputs/Reports/)

(Diperlukan Reports jika disertakan)

Nama yang ingin Anda berikan ke CodeCatalyst laporan yang akan dihasilkan dari laporan mentah Anda.

UI yang sesuai: Tab keluaran/Laporan/Konfigurasi laporan/Nama laporan secara manual

Format

(*aksi-nama-laporan-nama-1 /Outputs/Reports//)* ***Format***

(Diperlukan Reports jika disertakan)

Tentukan format file yang Anda gunakan untuk laporan Anda. Nilai yang mungkin adalah sebagai berikut.

- Untuk laporan pengujian:
  - Untuk Cucumber JSON, tentukan Cucumber (editor visual) atau `CUCUMBERJSON` (editor YAMB).
  - Untuk JUnit XHTML, tentukan JUnit (editor visual) atau `JUNITXML` (editor YAMG).
  - Untuk NUnit XHTML, tentukan NUnit (editor visual) atau `NUNITXML` (editor YAMG).
  - Untuk NUnit 3 XHTML, tentukan nUnit3 (editor visual) atau `NUNIT3XML` (editor YAMG).
  - Untuk Visual Studio TRX, tentukan Visual Studio TRX (editor visual) atau `VISUALSTUDIOTRX` (editor YAMG).
  - Untuk TestNG XML/TestNG, tentukan TestNG (editor visual) atau `TESTNGXML` (editor YAMB).
- Untuk laporan cakupan kode:
  - Untuk XHTML Clover, tentukan Clover (editor visual) atau `CLOVERXML` (editor YAMG).
  - Untuk Cobertura XHTML, tentukan Cobertura (editor visual) atau `COBERTURAXML` (editor YAMB).
  - Untuk JaCoCo XHTML, tentukan JaCoCo(editor visual) atau `JACOCOXML` (editor YAMB).
  - Untuk SimpleCov JSON yang dihasilkan oleh simplecov, bukan simplecov-json, tentukan Simplecov (editor visual) atau (editor YAMAL). `SIMPLECOV`
- Untuk laporan analisis komposisi perangkat lunak (SCA):
  - Untuk SARIF, tentukan SARIF (editor visual) atau `SARIFSCA` (editor YAMB).

UI yang sesuai: Tab Keluaran/Laporan/Konfigurasi laporan secara manual/ Tambahkan laporan/nama-laporan-1/Jenis laporan dan format Laporan

Configuration

(*nama-tindakan/*) Configuration

(Wajib) Bagian di mana Anda dapat menentukan properti konfigurasi tindakan.

UI yang sesuai: Tab konfigurasi

Steps

(*nama-tindakan) /Configuration/* ***Steps***

(Diperlukan)

Tentukan kode GitHub Tindakan Anda seperti yang muncul di halaman detail tindakan di GitHub Marketplace. Tambahkan kode berikut pedoman ini:

1. Tempel kode dari `steps:` bagian GitHub Action ke `Steps:` bagian CodeCatalyst alur kerja. Kode dimulai dengan tanda hubung (-) dan terlihat mirip dengan yang berikut ini.

   GitHub kode untuk ditempelkan:

   ```
   - name: Lint Code Base
     uses: github/super-linter@v4
     env:
       VALIDATE_ALL_CODEBASE: false
       DEFAULT_BRANCH: master
       GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
   ```

2. Tinjau kode yang baru saja Anda tempel dan modifikasi seperlunya agar sesuai dengan standar. CodeCatalyst Misalnya, dengan blok kode sebelumnya, Anda dapat menghapus kode dengan ***huruf miring merah***, dan menambahkan kode dalam huruf tebal.

   CodeCatalyst alur kerja yaml:

   ```
   Steps:
      - name: Lint Code Base
        uses: github/super-linter@v4
        env:
          VALIDATE_ALL_CODEBASE: false
   ```

```
        DEFAULT_BRANCH: master main
        GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
```

3. Untuk kode tambahan yang disertakan dengan GitHub Action tetapi tidak ada di dalam `steps:` bagian, tambahkan ke CodeCatalyst alur kerja menggunakan kode CodeCatalyst -equivalent. Anda dapat meninjau Alur kerja definisi YAMAL untuk mendapatkan wawasan tentang bagaimana Anda dapat mem-port GitHub kode Anda CodeCatalyst. Langkah-langkah migrasi terperinci berada di luar cakupan panduan ini.

Berikut adalah contoh cara menentukan jalur file dalam GitHub tindakan Tindakan:

```
Steps:
  - name: Lint Code Base
    uses: github/super-linter@v4
    ...
  - run: cd /sources/WorkflowSource/MyFolder/  && cat file.txt
  - run: cd /artifacts/MyGitHubAction/MyArtifact/MyFolder/  && cat file2.txt
```

Untuk informasi selengkapnya tentang menentukan jalur file, lihat Mereferensikan file dalam repositori sumber danMerujuk file dalam artefak.

UI yang sesuai: Tab GitHub konfigurasi/Tindakan YAMAL

Mengkonfigurasi gambar Docker lingkungan komputasi dan runtime untuk alur kerja

Dalam CodeCatalyst alur kerja, Anda dapat menentukan image lingkungan komputasi dan runtime yang CodeCatalyst digunakan untuk menjalankan tindakan alur kerja.

Compute mengacu pada mesin komputasi (CPU, memori, dan sistem operasi) yang dikelola dan dikelola oleh CodeCatalyst untuk menjalankan tindakan alur kerja.

> **Note**
>
> Jika komputasi didefinisikan sebagai properti alur kerja, maka komputasi tidak dapat didefinisikan sebagai properti dari tindakan apa pun dalam alur kerja tersebut. Demikian pula, jika komputasi didefinisikan sebagai properti dari tindakan apa pun, itu tidak dapat didefinisikan dalam alur kerja.

Gambar lingkungan runtime adalah wadah Docker di mana CodeCatalyst menjalankan tindakan alur kerja. Container Docker berjalan di atas platform komputasi yang Anda pilih, dan menyertakan sistem operasi dan alat tambahan yang mungkin diperlukan oleh tindakan alur kerja, seperti Node.js AWS CLI, dan .tar.

Topik

Jenis komputasi

CodeCatalyst menawarkan jenis komputasi berikut:

- Amazon EC2
- AWS Lambda

Amazon EC2 menawarkan fleksibilitas yang dioptimalkan selama aksi berjalan dan Lambda menawarkan kecepatan start-up aksi yang dioptimalkan. Lambda mendukung tindakan alur kerja yang lebih cepat berjalan karena latensi start-up yang lebih rendah. Lambda memungkinkan Anda menjalankan alur kerja dasar yang dapat membangun, menguji, dan menyebarkan aplikasi tanpa server dengan runtime umum. Runtime ini termasuk Node.js, Python, Java, .NET, dan Go. Namun, ada beberapa kasus penggunaan yang tidak didukung Lambda, dan jika berdampak pada Anda, gunakan jenis komputasi Amazon EC2:

- Lambda tidak mendukung gambar lingkungan runtime dari registri tertentu.
- Lambda tidak mendukung alat yang memerlukan izin root. Untuk alat seperti `yum` atau`rpm`, gunakan jenis komputasi Amazon EC2 atau alat lain yang tidak memerlukan izin root.

- Lambda tidak mendukung build atau run Docker. Tindakan berikut yang menggunakan image Docker tidak didukung: Deploy AWS CloudFormation stack, Deploy to Amazon ECS, Amazon S3 publish, AWS CDK bootstrap, AWS CDK deploy, invoke, dan Actions. AWS Lambda GitHub GitHub Tindakan berbasis Docker yang berjalan dalam CodeCatalyst GitHub tindakan Tindakan juga tidak didukung dengan komputasi Lambda. Anda dapat menggunakan alternatif yang tidak memerlukan izin root, seperti Podman.
- Lambda tidak mendukung penulisan ke file di luar. `/tmp` Saat mengonfigurasi tindakan alur kerja, Anda dapat mengonfigurasi ulang alat untuk menginstal atau menulis. `/tmp` Jika Anda memiliki tindakan build yang diinstal`npm`, pastikan Anda mengonfigurasinya untuk `/tmp` diinstal.
- Lambda tidak mendukung runtime lebih dari 15 menit.

Hitung armada

CodeCatalyst menawarkan armada komputasi berikut:

- Armada sesuai permintaan
- Armada yang disediakan

Dengan armada sesuai permintaan, ketika tindakan alur kerja dimulai, alur kerja menyediakan sumber daya yang dibutuhkan. Mesin-mesin dihancurkan ketika aksi selesai. Anda hanya membayar untuk jumlah menit yang Anda jalankan tindakan Anda. Armada sesuai permintaan dikelola sepenuhnya, dan mencakup kemampuan penskalaan otomatis untuk menangani lonjakan permintaan.

CodeCatalyst juga menawarkan armada yang disediakan yang berisi mesin yang ditenagai oleh Amazon EC2 yang dikelola oleh. CodeCatalyst Dengan armada yang disediakan, Anda mengonfigurasi satu set mesin khusus untuk menjalankan tindakan alur kerja Anda. Mesin-mesin ini tetap menganggur, siap untuk memproses tindakan segera. Dengan armada yang disediakan, mesin Anda selalu berjalan dan akan dikenakan biaya selama disediakan.

Untuk membuat, memperbarui, atau menghapus armada, Anda harus memiliki peran administrator Space atau peran administrator Proyek.

Properti armada sesuai permintaan

CodeCatalyst menyediakan armada sesuai permintaan berikut:

Nama	Sistem operasi	Arsitektur	vCPU	Memori (GiB)	Ruang disk	Jenis komputasi yang didukung
`Linux.Arm64.Large`	Amazon Linux 2	Arm64	2	4	64 GB	Amazon EC2
					10 GB	Lambda
`Linux.Arm64.XLarge`	Amazon Linux 2	Arm64	4	8	128 GB	Amazon EC2
					10 GB	Lambda
`Linux.Arm64.2XLarge`	Amazon Linux 2	Arm64	8	16	128 GB	Amazon EC2
`Linux.x86-64.Large`	Amazon Linux 2	x86-64	2	4	64 GB	Amazon EC2
					10 GB	Lambda
`Linux.x86-64.XLarge`	Amazon Linux 2	x86-64	4	8	128 GB	Amazon EC2
					10 GB	Lambda
`Linux.x86-64.2XLarge`	Amazon Linux 2	x86-64	8	16	128 GB	Amazon EC2

> **Note**
>
> Spesifikasi untuk armada sesuai permintaan akan bervariasi tergantung pada tingkat penagihan Anda. Untuk informasi selengkapnya, silakan lihat Harga .

Jika tidak ada armada yang dipilih, CodeCatalyst gunakan`Linux.x86-64.Large`.

Properti armada yang disediakan

Armada yang disediakan berisi properti berikut:

Sistem operasi

Sistem operasi. Sistem operasi berikut tersedia:

- Amazon Linux 2
- Windows Server 2022

> **Note**
>
> Armada Windows hanya didukung dalam aksi build. Tindakan lain saat ini tidak mendukung Windows.

Arsitektur

Arsitektur prosesor. Arsitektur berikut tersedia:

- x86_64
- Arm64

Jenis mesin

Jenis mesin untuk setiap contoh. Jenis mesin berikut tersedia:

vCPU	Memori (GiB)	Ruang disk	Sistem operasi
2	4	64 GB	Amazon Linux 2
4	8	128 GB	Amazon Linux 2
			Windows Server 2022
8	16	128 GB	Amazon Linux 2
			Windows Server 2022

Kapasitas

Jumlah awal mesin yang dialokasikan untuk armada, yang mendefinisikan jumlah tindakan yang dapat berjalan secara paralel.

Mode penskalaan

Mendefinisikan perilaku ketika jumlah tindakan melebihi kapasitas armada.

Penyediaan kapasitas tambahan sesuai permintaan

Mesin tambahan disiapkan sesuai permintaan yang secara otomatis ditingkatkan sebagai respons terhadap tindakan baru yang berjalan, dan kemudian diturunkan ke kapasitas dasar saat tindakan selesai. Ini dapat menimbulkan biaya tambahan, karena Anda membayar per menit untuk setiap mesin yang berjalan.

Tunggu hingga kapasitas armada tambahan tersedia

Tindakan berjalan ditempatkan dalam antrian sampai mesin tersedia. Ini membatasi biaya tambahan karena tidak ada mesin tambahan yang dialokasikan.

Membuat armada yang disediakan

Gunakan petunjuk berikut untuk membuat armada yang disediakan.

> **Note**
>
> Armada yang disediakan akan dinonaktifkan setelah 2 minggu tidak aktif. Jika digunakan lagi, mereka akan diaktifkan kembali secara otomatis, tetapi aktivasi ulang ini dapat menyebabkan latensi terjadi.

Untuk membuat armada yang disediakan

1. Di panel navigasi, pilih CI/CD, lalu pilih Compute.
2. Pilih Buat armada yang disediakan.
3. Di bidang teks nama armada yang disediakan, masukkan nama untuk armada Anda.
4. Dari menu drop-down sistem operasi, pilih sistem operasi.
5. Dari menu drop-down tipe Mesin, pilih jenis mesin untuk mesin Anda.
6. Di bidang teks Kapasitas, masukkan jumlah maksimum mesin di armada.

7. Dari menu drop-down mode Scaling, pilih perilaku overflow yang diinginkan. Untuk informasi lebih lanjut tentang bidang ini, lihatProperti armada yang disediakan.
8. Pilih Buat.

Setelah membuat armada yang disediakan, Anda siap untuk menetapkannya ke suatu tindakan. Untuk informasi selengkapnya, lihat Menetapkan armada yang disediakan atau komputasi sesuai permintaan untuk suatu tindakan.

Mengedit armada yang disediakan

Gunakan petunjuk berikut untuk mengedit armada yang disediakan.

> **Note**
>
> Armada yang disediakan akan dinonaktifkan setelah 2 minggu tidak aktif. Jika digunakan lagi, mereka akan diaktifkan kembali secara otomatis, tetapi aktivasi ulang ini dapat menyebabkan latensi terjadi.

Untuk mengedit armada yang disediakan

1. Di panel navigasi, pilih CI/CD, lalu pilih Compute.
2. Dalam daftar armada yang disediakan, pilih armada yang ingin Anda edit.
3. Pilih Edit.
4. Di bidang teks Kapasitas, masukkan jumlah maksimum mesin di armada.
5. Dari menu drop-down mode Scaling, pilih perilaku overflow yang diinginkan. Untuk informasi lebih lanjut tentang bidang ini, lihatProperti armada yang disediakan.
6. Pilih Simpan.

Menghapus armada yang disediakan

Gunakan petunjuk berikut untuk menghapus armada yang disediakan.

Untuk menghapus armada yang disediakan

> **Warning**
>
> Sebelum menghapus armada yang disediakan, hapus dari semua tindakan dengan menghapus `Fleet` properti dari kode YAMAL tindakan. Tindakan apa pun yang terus mereferensikan armada yang disediakan setelah dihapus akan gagal saat tindakan berjalan berikutnya.

1. Di panel navigasi, pilih CI/CD, lalu pilih Compute.
2. Dalam daftar Armada yang disediakan, pilih armada yang ingin Anda hapus.
3. Pilih Hapus.
4. Masukkan **`delete`** untuk mengonfirmasi penghapusan.
5. Pilih Hapus.

Menetapkan armada yang disediakan atau komputasi sesuai permintaan untuk suatu tindakan

Secara default, tindakan alur kerja menggunakan armada `Linux.x86-64.Large` sesuai permintaan dengan jenis komputasi Amazon EC2. Untuk menggunakan armada yang disediakan sebagai gantinya, atau menggunakan armada sesuai permintaan yang berbeda, sepertiLinux.x86-64.2XLarge, gunakan instruksi berikut.

Visual

Sebelum Anda mulai

- Jika Anda ingin menetapkan armada yang disediakan, Anda harus terlebih dahulu membuat armada yang disediakan. Untuk informasi selengkapnya, lihat Membuat armada yang disediakan.

Untuk menetapkan armada yang disediakan atau jenis armada yang berbeda untuk suatu tindakan

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.

2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Dalam diagram alur kerja, pilih tindakan yang ingin Anda tetapkan armada yang disediakan atau jenis armada baru.
8. Pilih tab Konfigurasi.
9. Dalam armada Compute, lakukan hal berikut:

   Tentukan mesin atau armada yang akan menjalankan alur kerja atau tindakan alur kerja Anda. Dengan armada sesuai permintaan, ketika suatu tindakan dimulai, alur kerja menyediakan sumber daya yang dibutuhkan, dan mesin dihancurkan ketika tindakan selesai. Contoh armada sesuai permintaan:`Linux.x86-64.Large`,. `Linux.x86-64.XLarge` Untuk informasi lebih lanjut tentang armada sesuai permintaan, lihat. Properti armada sesuai permintaan

   Dengan armada yang disediakan, Anda mengonfigurasi satu set mesin khusus untuk menjalankan tindakan alur kerja Anda. Mesin-mesin ini tetap menganggur, siap untuk memproses tindakan segera. Untuk informasi lebih lanjut tentang armada yang disediakan, lihat. Properti armada yang disediakan

   Jika `Fleet` dihilangkan, defaultnya adalah. `Linux.x86-64.Large`
10. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
11. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Sebelum Anda mulai

- Jika Anda ingin menetapkan armada yang disediakan, Anda harus terlebih dahulu membuat armada yang disediakan. Untuk informasi selengkapnya, lihat Membuat armada yang disediakan.

Untuk menetapkan armada yang disediakan atau jenis armada yang berbeda untuk suatu tindakan

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja. /
5. Pilih Edit.
6. Pilih YAMAL.
7. Temukan tindakan yang ingin Anda tetapkan untuk armada yang disediakan atau jenis armada baru.
8. Dalam aksi, tambahkan `Compute` properti dan atur `Fleet` ke nama armada Anda atau jenis armada sesuai permintaan. Untuk informasi selengkapnya, lihat deskripsi `Fleet` properti di Membangun dan menguji definisi YAMAL tindakan for your action.
9. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
10. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Berbagi komputasi di seluruh tindakan

Secara default, tindakan dalam alur kerja berjalan pada instance terpisah dalam sebuah fleet. Perilaku ini memberikan tindakan dengan isolasi dan prediktabilitas pada keadaan input. Perilaku default memerlukan konfigurasi eksplisit untuk berbagi konteks seperti file dan variabel antar tindakan.

Berbagi komputasi adalah kemampuan yang memungkinkan Anda menjalankan semua tindakan dalam alur kerja pada instance yang sama. Menggunakan berbagi komputasi dapat memberikan runtime alur kerja yang lebih cepat karena lebih sedikit waktu yang dihabiskan untuk menyediakan instance. Anda juga dapat berbagi file (artefak) antar tindakan tanpa konfigurasi alur kerja tambahan.

Saat alur kerja dijalankan menggunakan berbagi komputasi, instance dalam armada default atau yang ditentukan dicadangkan selama durasi semua tindakan dalam alur kerja tersebut. Ketika alur kerja berjalan selesai, reservasi instance dilepaskan.

Topik

- Menjalankan beberapa tindakan pada komputasi bersama

- Pertimbangan untuk berbagi komputasi
- Mengaktifkan berbagi komputasi
- Contoh

Menjalankan beberapa tindakan pada komputasi bersama

Anda dapat menggunakan `Compute` atribut dalam definisi YAMAL di tingkat alur kerja untuk menentukan properti tindakan fleet dan compute sharing. Anda juga dapat mengonfigurasi properti komputasi menggunakan editor visual di CodeCatalyst. Untuk menentukan armada, tetapkan nama armada yang ada, atur jenis komputasi ke EC2, dan aktifkan berbagi komputasi.

> **Note**
>
> Berbagi komputasi hanya didukung jika jenis komputasi disetel ke EC2, dan tidak didukung untuk sistem operasi Windows Server 2022. Untuk informasi selengkapnya tentang armada komputasi, tipe komputasi, dan properti, lihat. Mengkonfigurasi gambar Docker lingkungan komputasi dan runtime untuk alur kerja

> **Note**
>
> Jika Anda berada di tingkat Gratis dan Anda menentukan `Linux.x86-64.2XLarge` armada `Linux.x86-64.XLarge` atau secara manual dalam definisi alur kerja YAMAL, tindakan akan tetap berjalan pada armada default ()`Linux.x86-64.Large`. Untuk informasi selengkapnya tentang ketersediaan dan harga komputasi, lihat tabel untuk opsi tingkatan.

Saat berbagi komputasi diaktifkan, folder yang berisi sumber alur kerja secara otomatis disalin di seluruh tindakan. Anda tidak perlu mengonfigurasi artefak keluaran dan mereferensikannya sebagai artefak masukan di seluruh definisi alur kerja (file YAMAL). Sebagai penulis alur kerja, Anda perlu memasang variabel lingkungan menggunakan input dan output, seperti yang Anda lakukan tanpa menggunakan berbagi komputasi. Jika Anda ingin berbagi folder di antara tindakan di luar sumber alur kerja, pertimbangkan caching file. Untuk informasi selengkapnya, lihat Berbagi data antar tindakan dalam alur kerja menggunakan artefak dan Caching file di antara alur kerja berjalan.

Repositori sumber tempat file definisi alur kerja Anda berada diidentifikasi oleh label. `WorkflowSource` Saat menggunakan berbagi komputasi, sumber alur kerja diunduh dalam tindakan

pertama yang mereferensikannya dan secara otomatis tersedia untuk tindakan selanjutnya dalam alur kerja yang akan digunakan. Setiap perubahan yang dilakukan pada folder yang berisi sumber alur kerja oleh suatu tindakan, seperti menambahkan, memodifikasi, atau menghapus file, juga terlihat dalam tindakan selanjutnya dalam alur kerja. Anda dapat mereferensikan file yang berada di folder sumber alur kerja di salah satu tindakan alur kerja Anda, seperti yang Anda bisa tanpa menggunakan berbagi komputasi. Untuk informasi selengkapnya, lihat Mereferensikan file dalam repositori sumber.

> **Note**
>
> Alur kerja berbagi komputasi perlu menentukan urutan tindakan yang ketat, sehingga tindakan paralel tidak dapat diatur. Meskipun artefak keluaran dapat dikonfigurasi pada tindakan apa pun dalam urutan, artefak input tidak didukung.

Pertimbangan untuk berbagi komputasi

Anda dapat menjalankan alur kerja dengan berbagi komputasi untuk mempercepat alur kerja berjalan dan berbagi konteks antar tindakan dalam alur kerja yang menggunakan instance yang sama. Pertimbangkan hal berikut untuk menentukan apakah menggunakan berbagi komputasi sesuai untuk skenario Anda:

	Berbagi komputasi	Tanpa berbagi komputasi
Jenis komputasi	Amazon EC2	Amazon EC2, AWS Lambda
Penyediaan instans	Tindakan berjalan pada contoh yang sama	Tindakan berjalan pada instance terpisah
Sistem operasi	Amazon Linux 2	Amazon Linux 2, Windows Server 2022 (hanya tindakan build)
Referensi file	`$CATALYST_SOURCE_DIR_WorkflowSource` , `/sources/WorkflowSource/`	`$CATALYST_SOURCE_DIR_WorkflowSource` , `/sources/WorkflowSource/`

	Berbagi komputasi	Tanpa berbagi komputasi
Struktur alur kerja	Tindakan hanya dapat berjalan secara berurutan	Tindakan dapat berjalan paralel
Mengakses data di seluruh tindakan alur kerja	Akses sumber alur kerja yang di-cache () `WorkflowSource`	Akses output artefak bersama (memerlukan konfigurasi tambahan)

Mengaktifkan berbagi komputasi

Gunakan instruksi berikut untuk mengaktifkan berbagi komputasi untuk alur kerja.

Visual

Untuk mengaktifkan berbagi komputasi menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda.
5. Pilih Edit.
6. Pilih Visual.
7. Pilih properti Workflow.
8. Dari menu tarik-turun tipe Compute, pilih EC2.
9. (Opsional) Dari armada Compute - menu tarik-turun opsional, pilih armada yang ingin Anda gunakan untuk menjalankan tindakan alur kerja. Anda dapat memilih armada sesuai permintaan atau membuat dan memilih armada yang disediakan. Untuk informasi selengkapnya, lihat Membuat armada yang disediakan dan Menetapkan armada yang disediakan atau komputasi sesuai permintaan untuk suatu tindakan
10. Alihkan sakelar untuk mengaktifkan berbagi komputasi dan menjalankan tindakan dalam alur kerja pada armada yang sama.
11. (Opsional) Pilih mode lari untuk alur kerja. Untuk informasi selengkapnya, lihat Mengonfigurasi perilaku antrian run.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk mengaktifkan berbagi komputasi menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda.
5. Pilih Edit.
6. Pilih YAMAL.
7. Aktifkan berbagi komputasi menyetel `SharedInstance` bidang ke `TRUE` dan `Type` ke`EC2`. Setel `Fleet` ke armada komputasi yang ingin Anda gunakan untuk menjalankan tindakan alur kerja. Anda dapat memilih armada sesuai permintaan atau membuat dan memilih armada yang disediakan. Untuk informasi selengkapnya, lihat Membuat armada yang disediakan dan Menetapkan armada yang disediakan atau komputasi sesuai permintaan untuk suatu tindakan

   Dalam alur kerja YAMAL, tambahkan kode yang mirip dengan berikut ini:

   ```
   Name: MyWorkflow
   SchemaVersion: "1.0"
   Compute: # Define compute configuration.
     Type: EC2
     Fleet: MyFleet # Optionally, choose an on-demand or provisioned fleet.
     SharedInstance: true # Turn on compute sharing. Default is False.
   Actions:
     BuildFirst:
       Identifier: aws/build@v1
       Inputs:
         Sources:
           - WorkflowSource
       Configuration:
         Steps:
           - Run: ...
           ...
   ```
8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Contoh

Topik

- Contoh: Amazon S3 Publikasikan

Contoh: Amazon S3 Publikasikan

Contoh alur kerja berikut menunjukkan cara melakukan tindakan Publikasikan Amazon S3 Amazon S3 dengan dua cara: pertama menggunakan artefak input dan kemudian menggunakan berbagi komputasi. Dengan berbagi komputasi, artefak input tidak diperlukan karena Anda dapat mengakses cache. `WorkflowSource` Selain itu, artefak keluaran dalam aksi Build tidak lagi diperlukan. Tindakan S3 Publish dikonfigurasi untuk menggunakan `DependsOn` properti eksplisit untuk mempertahankan tindakan berurutan; tindakan Build harus berjalan dengan sukses agar tindakan S3 Publish dapat berjalan.

- Tanpa berbagi komputasi, Anda perlu menggunakan artefak input dan berbagi output dengan tindakan selanjutnya:

```
Name: S3PublishUsingInputArtifact
SchemaVersion: "1.0"
Actions:
  Build:
    Identifier: aws/build@v1
    Outputs:
      Artifacts:
        - Name: ArtifactToPublish
          Files: [output.zip]
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      Steps:
        - Run: ./build.sh # Build script that generates output.zip
  PublishToS3:
    Identifier: aws/s3-publish@v1
    Inputs:
      Artifacts:
      - ArtifactToPublish
    Environment:
      Connections:
```

```
        - Role: codecatalyst-deployment-role
          Name: dev-deployment-role
      Name: dev-connection
    Configuration:
      SourcePath: output.zip
      DestinationBucketName: dev-bucket
```

- Saat menggunakan berbagi komputasi dengan `SharedInstance` menyetelnya`TRUE`, Anda dapat menjalankan beberapa tindakan pada instance yang sama dan berbagi artefak dengan menentukan satu sumber alur kerja. Artefak masukan tidak diperlukan dan tidak dapat ditentukan:

```
Name: S3PublishUsingComputeSharing
SchemaVersion: "1.0"
Compute:
  Type: EC2
  Fleet: dev-fleet
  SharedInstance: TRUE
Actions:
  Build:
    Identifier: aws/build@v1
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      Steps:
        - Run: ./build.sh # Build script that generates output.zip
  PublishToS3:
    Identifier: aws/s3-publish@v1
    DependsOn:
      - Build
    Environment:
      Connections:
        - Role: codecatalyst-deployment-role
          Name: dev-deployment-role
      Name: dev-connection
    Configuration:
      SourcePath: output.zip
      DestinationBucketName: dev-bucket
```

Menentukan gambar Docker lingkungan runtime

Gambar lingkungan runtime adalah wadah Docker di mana CodeCatalyst menjalankan tindakan alur kerja. Container Docker berjalan di atas platform komputasi yang Anda pilih, dan menyertakan sistem operasi dan alat tambahan yang mungkin diperlukan oleh tindakan alur kerja, seperti Node.js AWS CLI, dan .tar.

Secara default, tindakan alur kerja akan berjalan pada salah satu gambar aktif yang disediakan dan dikelola oleh CodeCatalyst. Hanya tindakan build dan test yang mendukung gambar kustom. Untuk informasi selengkapnya, lihat Menetapkan image Docker lingkungan runtime kustom ke suatu tindakan.

Topik

Gambar aktif

Gambar aktif adalah gambar lingkungan runtime yang didukung penuh oleh CodeCatalyst dan menyertakan perkakas yang sudah diinstal sebelumnya. Saat ini ada dua set gambar aktif: satu dirilis pada Maret 2024, dan satu lagi dirilis pada November 2022.

Apakah suatu tindakan menggunakan gambar Maret 2024 atau November 2022 tergantung pada tindakannya:

- Membuat dan menguji tindakan yang ditambahkan ke alur kerja pada atau setelah 26 Maret 2024 akan menyertakan `Container` bagian dalam definisi YAML mereka yang secara eksplisit menentukan gambar Maret 2024. Anda dapat menghapus `Container` bagian tersebut secara opsional untuk kembali ke gambar November 2022.
- Tindakan pembuatan dan uji yang ditambahkan ke alur kerja sebelum 26 Maret 2024 tidak akan menyertakan `Container` bagian dalam definisi YAMLnya, dan akibatnya akan menggunakan gambar November 2022. Anda dapat menyimpan gambar November 2022, atau Anda dapat meningkatkannya. Untuk memutakhirkan gambar, buka tindakan di editor visual, pilih tab Konfigurasi, lalu pilih gambar Maret 2024 dari daftar drop-down gambar docker lingkungan

Runtime. Pilihan ini akan menambahkan `Container` bagian ke definisi YAMAL tindakan yang diisi dengan gambar Maret 2024 yang sesuai.

- Semua tindakan lain akan menggunakan gambar November 2022 terlepas dari kapan mereka ditambahkan ke alur kerja. Memutakhirkan tindakan ini untuk menggunakan gambar Maret 2024 saat ini tidak dimungkinkan.

Topik

Maret 2024 foto

Gambar Maret 2024 adalah gambar terbaru yang disediakan oleh CodeCatalyst. Ada satu gambar Maret 2024 per kombinasi jenis/armada komputasi.

Tabel berikut menunjukkan alat yang diinstal pada setiap gambar Maret 2024.

Alat gambar Maret 2024

Alat	CodeCatalyst Amazon EC2 untuk Linux x86_64 - **`CodeCatalystLinux_x86_64:2024_03`**	CodeCatalyst Lambda untuk Linux x86_64 - **`CodeCatalystLinuxLambda_x86_64:2024_03`**	CodeCatalyst Amazon EC2 untuk Linux Arm64 - **`CodeCatalystLinux_Arm64:2024_03`**	CodeCatal Lambda ur Arm64 - **`Co ystLinux ambda_Ar 64:2024_`**
AWS CLI	2.15.17	2.15.17	2.15.17	2.15.17
AWS Copilot CLI	1.32.1	1.32.1	1.32.1	1.32.1
Docker	24.0.9	N/A	24.0.9	N/A
Docker Compose	2.23.3	N/A	2.23.3	N/A
Git	2.43.0	2.43.0	2.43.0	2.43.0
Go	1.21.5	1.21.5	1.21.5	1.21.5

Alat	CodeCatalyst Amazon EC2 untuk Linux x86_64 - **CodeCatalystLinux_x86_64:2024_03**	CodeCatalyst Lambda untuk Linux x86_64 - **CodeCatalystLinuxLambda_x86_64:2024_03**	CodeCatalyst Amazon EC2 untuk Linux Arm64 - **CodeCatalystLinux_Arm64:2024_03**	CodeCatal Lambda un Arm64 - **Co ystLinux ambda_Ar 64:2024_**
Gradle	8.5	8.5	8.5	8.5
Java	Corretto17	Corretto17	Corretto17	Corretto17
Maven	3.9.6	3.9.6	3.9.6	3.9.6
Node.js	18.19.0	18.19.0	18.19.0	18.19.0
npm	10.2.3	10.2.3	10.2.3	10.2.3
Python	3.9.18	3.9.18	3.9.18	3.9.18
Python3	3.11.6	3.11.6	3.11.6	3.11.6
pip	22.3.1	22.3.1	22.3.1	22.3.1
.NET	8.0.100	8.0.100	8.0.100	8.0.100

Gambar November 2022

Ada satu gambar November 2022 per kombinasi jenis komputasi/armada. Ada juga image Windows November 2022 yang tersedia dengan tindakan build jika Anda telah mengonfigurasi armada yang disediakan.

Tabel berikut menunjukkan alat yang diinstal pada setiap gambar November 2022.

Alat gambar November 2022

Alat	CodeCatalyst Amazon EC2 untuk Linux x86_64 - **CodeCatalystLinux_x86_64:2022_11**	CodeCatalyst Lambda untuk Linux x86_64 - **CodeCatalystLinuxLambda_x86_64:2022_11**	CodeCatalyst Amazon EC2 untuk Linux Arm64 - **CodeCatalystLinux_Arm64:2022_11**	CodeCatalyst Lambda un Arm64 - **CodeCatalystLinuxLambda_Ar64:2022_**
AWS CLI	2.15.17	2.15.17	2.15.17	2.15.17
AWS Copilot CLI	0.6.0	0.6.0	N/A	N/A
Docker	23.01	N/A	23.0.1	N/A
Docker Compose	2.16.0	N/A	2.16.0	N/A
Git	2.40.0	2.40.0	2.39.2	2.39.2
Go	1.20.2	1.20.2	1.20.1	1.20.1
Gradle	8.0.2	8.0.2	8.0.1	8.0.1
Java	Corretto17	Corretto17	Corretto17	Corretto17
Maven	3.9.4	3.9.4	3.9.0	3.9.0
Node.js	16.20.2	16.20.2	16.19.1	16.14.2
npm	8.19.4	8.19.4	8.19.3	8.5.0
Python	3.9.15	2.7.18	3.11.2	2.7.18
Python3	N/A	3.9.15	N/A	3.11.2
pip	22.2.2	22.2.2	23.0.1	23.0.1
.NET	6.0.407	6.0.407	6.0.406	6.0.406

Bagaimana jika gambar aktif tidak menyertakan alat yang saya butuhkan?

Jika tidak ada gambar aktif yang disediakan oleh CodeCatalyst menyertakan alat yang Anda butuhkan, Anda memiliki beberapa opsi:

- Anda dapat memberikan image Docker lingkungan runtime kustom yang menyertakan alat yang diperlukan. Untuk informasi selengkapnya, lihat Menetapkan image Docker lingkungan runtime kustom ke suatu tindakan.

  > **Note**
  >
  > Jika Anda ingin memberikan image Docker lingkungan runtime kustom, pastikan gambar kustom Anda telah menginstal Git di dalamnya.

- Anda dapat meminta build atau test action alur kerja Anda menginstal alat yang Anda butuhkan.

  Misalnya, Anda dapat menyertakan petunjuk berikut di `Steps` bagian kode YAMP build atau test action:

  ```
  Configuration:
    Steps:
      - Run: ./setup-script
  ```

  Instruksi *`setup-script`* kemudian akan menjalankan skrip berikut untuk menginstal Node package manager (npm):

  ```
  #!/usr/bin/env bash
  echo "Setting up environment"

  touch ~/.bashrc
  curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
  source ~/.bashrc
  nvm install v16.1.0
  source ~/.bashrc
  ```

  Untuk informasi selengkapnya tentang aksi build YAMAL, lihatMembangun dan menguji definisi YAMAL tindakan.

Menetapkan image Docker lingkungan runtime kustom ke suatu tindakan

Jika Anda tidak ingin menggunakan gambar Aktif yang disediakan oleh CodeCatalyst, Anda dapat memberikan image Docker lingkungan runtime kustom. Jika Anda ingin memberikan gambar kustom, pastikan Git sudah terpasang di dalamnya. Gambar dapat berada di Docker Hub, Amazon Elastic Container Registry, atau repositori publik apa pun.

Untuk mempelajari cara membuat image Docker kustom, lihat Containerize aplikasi dalam dokumentasi Docker.

Gunakan petunjuk berikut untuk menetapkan image Docker lingkungan runtime kustom Anda ke tindakan. Setelah menentukan gambar, CodeCatalyst terapkan ke platform komputasi Anda saat tindakan dimulai.

> **Note**
>
> Tindakan berikut tidak mendukung lingkungan runtime kustom Gambar Docker: Deploy AWS CloudFormation stack, Deploy to ECS, dan GitHub Actions. Lingkungan runtime kustom Gambar Docker juga tidak mendukung jenis komputasi Lambda.

Visual

Untuk menetapkan lingkungan runtime kustom gambar Docker menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
3. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
4. Pilih Edit.
5. Pilih Visual.
6. Dalam diagram alur kerja, pilih tindakan yang akan menggunakan image Docker lingkungan runtime kustom Anda.
7. Pilih tab Konfigurasi.
8. Di dekat bagian bawah, isi bidang berikut.

   Lingkungan runtime Gambar Docker - opsional

Tentukan registri tempat gambar Anda disimpan. Nilai yang valid meliputi:

- `CODECATALYST`(Editor YAMAL)

  Gambar disimpan dalam CodeCatalyst registri.

- Docker Hub (editor visual) atau `DockerHub` (editor YAMAL)

  Gambar disimpan dalam registri gambar Docker Hub.

- Registri lain (editor visual) atau `Other` (editor YAMAL)

  Gambar disimpan dalam registri gambar khusus. Registri apa pun yang tersedia untuk umum dapat digunakan.

- Amazon Elastic Container Registry (editor visual) atau `ECR` (editor YAMAL)

  Gambar disimpan dalam repositori gambar Amazon Elastic Container Registry. Untuk menggunakan gambar di repositori Amazon ECR, tindakan ini memerlukan akses ke Amazon ECR. Untuk mengaktifkan akses ini, Anda harus membuat peran IAM yang mencakup izin berikut dan kebijakan kepercayaan khusus. (Anda dapat mengubah peran yang ada untuk menyertakan izin dan kebijakan, jika Anda mau.)

  Peran IAM harus menyertakan izin berikut dalam kebijakan perannya:

  - `ecr:BatchCheckLayerAvailability`
  - `ecr:BatchGetImage`
  - `ecr:GetAuthorizationToken`
  - `ecr:GetDownloadUrlForLayer`

  Peran IAM harus menyertakan kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
```

```
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Untuk informasi selengkapnya tentang membuat peran IAM, lihat Membuat peran menggunakan kebijakan kepercayaan khusus (konsol) di Panduan Pengguna IAM.

Setelah Anda membuat peran, Anda harus menetapkannya ke tindakan melalui lingkungan. Untuk informasi selengkapnya, lihat Mengaitkan lingkungan, koneksi akun, dan peran IAM dengan tindakan alur kerja.

URL gambar ECR, gambar Docker Hub atau URL Gambar

Tentukan satu dari yang berikut ini:

- Jika Anda menggunakan `CODECATALYST` registri, atur gambar ke salah satu gambar aktif berikut:
  - `CodeCatalystLinux_x86_64:2024_03`
  - `CodeCatalystLinux_x86_64:2022_11`
  - `CodeCatalystLinux_Arm64:2024_03`
  - `CodeCatalystLinux_Arm64:2022_11`
  - `CodeCatalystLinuxLambda_x86_64:2024_03`
  - `CodeCatalystLinuxLambda_x86_64:2022_11`
  - `CodeCatalystLinuxLambda_Arm64:2024_03`
  - `CodeCatalystLinuxLambda_Arm64:2022_11`
  - `CodeCatalystWindows_x86_64:2022_11`
- Jika Anda menggunakan registri Docker Hub, atur gambar ke nama gambar Docker Hub dan tag opsional.

  Contoh: `postgres:latest`
- Jika Anda menggunakan registri Amazon ECR, atur gambar ke URI registri Amazon ECR.

Contoh: `111122223333.dkr.ecr.us-west-2.amazonaws.com/codecatalyst-ecs-image-repo`

- Jika Anda menggunakan registri kustom, atur gambar ke nilai yang diharapkan oleh registri kustom.

9. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
10. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menetapkan image Docker lingkungan runtime kustom menggunakan editor YAMAL

1. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
3. Pilih Edit.
4. Pilih YAMAL.
5. Temukan tindakan yang ingin Anda tetapkan pada image Docker lingkungan runtime.
6. Dalam tindakan, tambahkan `Container` bagian dan yang mendasari `Registry` dan `Image` properti. Untuk informasi selengkapnya, lihat deskripsi`Container`, `Registry` dan `Image` properti di Tindakan untuk tindakan Anda.
7. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
8. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Contoh

Contoh berikut menunjukkan cara menetapkan image Docker lingkungan runtime kustom ke tindakan dalam file definisi alur kerja.

Topik

- Contoh: Menggunakan image Docker lingkungan runtime kustom untuk menambahkan dukungan untuk Node.js 18 dengan Amazon ECR
- Contoh: Menggunakan image Docker lingkungan runtime kustom untuk menambahkan dukungan untuk Node.js 18 dengan Docker Hub

Contoh: Menggunakan image Docker lingkungan runtime kustom untuk menambahkan dukungan untuk Node.js 18 dengan Amazon ECR

Contoh berikut menunjukkan cara menggunakan image Docker lingkungan runtime kustom untuk menambahkan dukungan untuk Node.js 18 dengan Amazon ECR.

```
Configuration:
  Container:
    Registry: ECR
    Image: public.ecr.aws/amazonlinux/amazonlinux:2023
```

Contoh: Menggunakan image Docker lingkungan runtime kustom untuk menambahkan dukungan untuk Node.js 18 dengan Docker Hub

Contoh berikut menunjukkan cara menggunakan image Docker lingkungan runtime kustom untuk menambahkan dukungan untuk Node.js 18 dengan Docker Hub.

```
Configuration:
  Container:
    Registry: DockerHub
    Image: node:18.18.2
```

Menghubungkan alur kerja ke repositori sumber

Sumber, juga disebut sumber input, adalah repositori sumber yang menghubungkan tindakan alur kerja untuk mendapatkan file yang dibutuhkan untuk menjalankan operasinya. Misalnya, tindakan alur kerja mungkin terhubung ke repositori sumber untuk mendapatkan file sumber aplikasi untuk membangun aplikasi.

CodeCatalyst alur kerja mendukung sumber-sumber berikut:

- CodeCatalyst repositori sumber - Untuk informasi lebih lanjut, lihat. Simpan dan berkolaborasi pada kode dengan repositori sumber di CodeCatalyst
- GitHub dan repositori Bitbucket - Untuk informasi lebih lanjut, lihat. Tambahkan fungsionalitas ke proyek dengan ekstensi di CodeCatalyst

Topik

- Menentukan sumber yang akan menyimpan file definisi alur kerja

- Menentukan sumber yang akan digunakan tindakan alur kerja
- Mereferensikan file dalam repositori sumber
- Variabel yang dihasilkan oleh sumber (BranchName“” dan CommidId “”)

Menentukan sumber yang akan menyimpan file definisi alur kerja

Gunakan petunjuk berikut untuk menentukan repositori CodeCatalyst sumber tempat Anda ingin menyimpan file definisi alur kerja Anda. Jika Anda lebih suka menentukan repositori GitHub atau Bitbucket, lihat saja. Tambahkan fungsionalitas ke proyek dengan ekstensi di CodeCatalyst

Repositori sumber tempat file definisi alur kerja Anda berada diidentifikasi oleh label,. `WorkflowSource`

> **Note**
>
> Anda menentukan repositori sumber tempat file definisi alur kerja Anda berada saat pertama kali mengkomit file definisi alur kerja Anda. Setelah komit ini, file definisi repositori dan alur kerja ditautkan bersama secara permanen. Satu-satunya cara untuk mengubah repositori setelah komit awal adalah dengan membuat ulang alur kerja di repositori yang berbeda.

Untuk menentukan repositori sumber yang akan menyimpan file definisi alur kerja

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih Buat alur kerja dan buat alur kerja. Untuk informasi selengkapnya, lihat Untuk membuat alur kerja menggunakan editor visual.

   Selama proses pembuatan alur kerja, Anda diminta untuk menentukan CodeCatalyst repositori tempat Anda ingin menyimpan file definisi alur kerja Anda.

Menentukan sumber yang akan digunakan tindakan alur kerja

Gunakan petunjuk berikut untuk menentukan repositori sumber yang akan digunakan dengan tindakan alur kerja. Saat startup, tindakan menggabungkan file di repositori sumber yang

dikonfigurasi menjadi artefak, mengunduh artefak ke image Docker lingkungan runtime tempat tindakan berjalan, dan kemudian menyelesaikan pemrosesannya menggunakan file yang diunduh.

> **Note**
>
> Saat ini, dalam tindakan alur kerja, Anda hanya dapat menentukan satu repositori sumber, yang merupakan repositori sumber tempat file definisi alur kerja berada (dalam direktori). `.codecatalyst/workflows/` Repositori sumber ini diwakili oleh label. `WorkflowSource`

Visual

Untuk menentukan repositori sumber yang akan digunakan tindakan (editor visual)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Dalam diagram alur kerja, pilih tindakan di mana Anda ingin menentukan sumbernya.
8. Pilih Input.
9. Dalam Sumber - opsional lakukan hal berikut:

   Tentukan label yang mewakili repositori sumber yang akan dibutuhkan oleh tindakan. Saat ini, satu-satunya label yang didukung adalah`WorkflowSource`, yang mewakili repositori sumber tempat file definisi alur kerja Anda disimpan.

   Jika Anda menghilangkan sumber, maka Anda harus menentukan setidaknya satu artefak input di bawah. *`action-name`*`/Inputs/Artifacts`

   Untuk informasi selengkapnya tentang sumber, lihat Menghubungkan alur kerja ke repositori sumber.
10. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
11. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menentukan repositori sumber yang akan digunakan tindakan (editor YAMG)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Dalam suatu tindakan, tambahkan kode yang mirip dengan berikut ini:

```
action-name:
 Inputs:
   Sources:
     - WorkflowSource
```

   Untuk informasi selengkapnya, lihat deskripsi `Sources` properti Alur kerja definisi YAMAL untuk tindakan Anda.
8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Mereferensikan file dalam repositori sumber

Jika Anda memiliki file yang berada di repositori sumber, dan Anda perlu merujuk ke file-file ini di salah satu tindakan alur kerja Anda, selesaikan prosedur berikut.

> **Note**
>
> Lihat juga Merujuk file dalam artefak.

Untuk mereferensikan file dalam repositori sumber

- Dalam tindakan di mana Anda ingin mereferensikan file, tambahkan kode yang mirip dengan yang berikut ini:

```
Actions:
  My-action:
    Inputs:
      Sources:
        - WorkflowSource
      Configuration:
        Steps:
        - run: cd my-app && cat file1.jar
```

Dalam kode sebelumnya, tindakan terlihat di `my-app` direktori di root repositori `WorkflowSource` sumber untuk menemukan dan menampilkan file. `file1.jar`

Variabel yang dihasilkan oleh sumber (BranchName“” dan CommidId “”)

CodeCatalyst Sumber menghasilkan dan menetapkan variabel "BranchName" dan "CommitId" saat alur kerja Anda berjalan. Ini dikenal sebagai variabel yang telah ditentukan sebelumnya. Lihat tabel berikut untuk informasi tentang variabel-variabel ini.

Untuk informasi tentang mereferensikan variabel-variabel ini dalam alur kerja, lihat. Menggunakan variabel yang telah ditentukan

Kunci	Nilai
CommitId	ID komit yang mewakili status repositori pada saat alur kerja dijalankan. Contoh: `example3819261db00a3ab59468c8b` Lihat juga: Contoh: Mereferensikan variabel CommitId standar ""
BranchName	Nama cabang tempat alur kerja dijalankan. Contoh:`main`,`feature/branch` , `test-LiJuan`

Kunci	Nilai
	Lihat juga: Contoh: Mereferensikan variabel BranchName standar ""

Menerbitkan dan mengimpor paket menggunakan alur kerja

Paket adalah bundel yang mencakup perangkat lunak dan metadata yang diperlukan untuk menginstal perangkat lunak dan menyelesaikan dependensi apa pun. CodeCatalyst mendukung format paket npm.

Paket terdiri dari:

- Nama (misalnya, `webpack` adalah nama paket npm populer)
- Namespace opsional (misalnya, `@types` di `@types/node`)
- Satu set versi (misalnya,`1.0.0`,`1.0.1`,`1.0.2`)
- Metadata tingkat paket (misalnya, tag dist npm)

Di CodeCatalyst, Anda dapat mempublikasikan paket ke dan mengkonsumsi paket dari repositori CodeCatalyst paket di alur kerja Anda. Anda dapat mengonfigurasi tindakan build atau test dengan repositori CodeCatalyst paket untuk secara otomatis mengonfigurasi klien npm tindakan untuk mendorong dan menarik paket dari repositori yang ditentukan.

Untuk informasi selengkapnya tentang paket, lihatPublikasikan dan bagikan paket perangkat lunak di CodeCatalyst.

> **Note**
>
> Saat ini, tindakan build dan uji mendukung repositori CodeCatalyst paket.

Topik

- Menentukan repositori CodeCatalyst paket dalam alur kerja
- Contoh menentukan repositori paket dalam alur kerja

Menentukan repositori CodeCatalyst paket dalam alur kerja

Di CodeCatalyst, Anda dapat menambahkan repositori CodeCatalyst paket ke tindakan build dan pengujian di alur kerja Anda. Repositori paket Anda harus dikonfigurasi dengan format paket, seperti npm. Anda juga dapat memilih untuk menyertakan urutan cakupan untuk repositori paket yang Anda pilih.

Gunakan petunjuk berikut untuk menentukan konfigurasi paket yang akan digunakan dengan tindakan alur kerja.

Visual

Untuk menentukan konfigurasi paket yang akan digunakan tindakan (editor visual)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Dalam diagram alur kerja, pilih tindakan yang ingin Anda konfigurasikan dengan repositori paket.
8. Pilih Paket.
9. Dari menu tarik-turun Tambahkan konfigurasi, pilih konfigurasi paket yang ingin Anda gunakan dengan tindakan alur kerja Anda.
10. Pilih Tambahkan repositori paket.
11. Di menu tarik-turun Package repository, tentukan nama repositori CodeCatalyst paket Anda yang ingin digunakan oleh tindakan.

    Untuk informasi lebih lanjut tentang repositori paket, lihat. Package repositori
12. (Opsional) Dalam Lingkup - opsional, tentukan urutan cakupan yang ingin Anda tentukan dalam registri paket Anda.

    Saat mendefinisikan cakupan, repositori paket yang ditentukan dikonfigurasi sebagai registri untuk semua cakupan yang terdaftar. Jika paket dengan cakupan diminta melalui klien npm,

itu akan menggunakan repositori itu alih-alih default. Setiap nama lingkup harus diawali dengan “@”.

Jika `Scopes` dihilangkan, maka repositori paket yang ditentukan dikonfigurasi sebagai registri default untuk semua paket yang digunakan oleh tindakan.

Untuk informasi selengkapnya tentang cakupan, lihat Ruang nama Package dan paket Scoped.

13. Pilih Tambahkan.
14. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
15. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menentukan konfigurasi paket yang akan digunakan tindakan (editor YAMG)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMG.
7. Dalam suatu tindakan, tambahkan kode yang mirip dengan berikut ini:

```
action-name:
 Configuration:
    Packages:
        NpmConfiguration:
          PackageRegistries:
            - PackagesRepository: package-repository
              Scopes:
                - "@scope"
```

Untuk informasi selengkapnya, lihat deskripsi `Packages` properti Membangun dan menguji definisi YAMAL tindakan untuk tindakan Anda.

8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.

9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Contoh menentukan repositori paket dalam alur kerja

Contoh berikut menunjukkan cara referensi paket dalam file definisi alur kerja.

Topik

- Contoh: Mendefinisikan paket dengan NpmConfiguration
- Contoh: Mengganti registri default
- Contoh: Mengganti cakupan dalam registri paket Anda

Contoh: Mendefinisikan paket dengan **`NpmConfiguration`**

Contoh berikut menunjukkan cara mendefinisikan paket dengan `NpmConfiguration` dalam file definisi alur kerja Anda.

```
Actions:
  Build:
  Identifier: aws/build-beta@v1
  Configuration:
    Packages:
        NpmConfiguration:
          PackageRegistries:
            - PackagesRepository: main-repo
            - PackagesRepository: scoped-repo
              Scopes:
                - "@scope1"
```

Contoh ini mengkonfigurasi klien npm seperti:

```
default: main-repo
@scope1: scoped-repo
```

Dalam contoh ini, ada dua repositori yang didefinisikan. Registri default diatur `main-repo` seperti yang didefinisikan tanpa ruang lingkup. Lingkup `@scope1` dikonfigurasi `PackageRegistries` untuk`scoped-repo`.

Contoh: Mengganti registri default

Contoh berikut menunjukkan cara mengganti registri default.

```
NpmConfiguration:
  PackageRegistries:
    - PackagesRepository: my-repo-1
    - PackagesRepository: my-repo-2
    - PackagesRepository: my-repo-3
```

Contoh ini mengkonfigurasi klien npm seperti:

```
default: my-repo-3
```

Jika Anda menentukan beberapa repositori default, repositori terakhir akan diprioritaskan. Dalam contoh ini, repositori terakhir yang terdaftar adalah`my-repo-3`, artinya npm akan terhubung ke. `my-repo-3` Ini mengesampingkan `my-repo-1` repositori dan. `my-repo-2`

Contoh: Mengganti cakupan dalam registri paket Anda

Contoh berikut menunjukkan cara mengganti cakupan dalam registri paket Anda.

```
NpmConfiguration:
  PackageRegistries:
    - PackagesRepository: my-default-repo
    - PackagesRepository: my-repo-1
      Scopes:
        - "@scope1"
        - "@scope2"
    - PackagesRepository: my-repo-2
      Scopes:
        - "@scope2"
```

Contoh ini mengkonfigurasi klien npm seperti:

```
default: my-default-repo
@scope1: my-repo-1
@scope2: my-repo-2
```

Jika Anda menyertakan cakupan utama, repositori terakhir akan diprioritaskan. Dalam contoh ini, terakhir kali lingkup `@scope2` dikonfigurasi `PackageRegistries` adalah untuk`my-repo-2`. Ini mengesampingkan ruang lingkup yang `@scope2` dikonfigurasi untuk. `my-repo-1`

Memanggil AWS Lambda fungsi menggunakan alur kerja

Bagian ini menjelaskan cara memanggil AWS Lambda fungsi menggunakan CodeCatalyst alur kerja. Untuk mencapai ini, Anda harus menambahkan tindakan AWS Lambda pemanggilan ke alur kerja Anda. Tindakan AWS Lambda pemanggilan memanggil fungsi Lambda yang Anda tentukan.

Selain menjalankan fungsi Anda, tindakan AWS Lambda pemanggilan juga mengonversi setiap kunci tingkat atas dalam payload respons yang diterima dari fungsi Lambda menjadi variabel keluaran alur kerja. Variabel-variabel ini kemudian dapat direferensikan dalam tindakan alur kerja berikutnya. Jika Anda tidak ingin semua kunci tingkat atas dikonversi ke variabel, Anda dapat menggunakan filter untuk menentukan yang tepat. Untuk informasi selengkapnya, lihat deskripsi ResponseFilters properti diTindakan “AWS Lambda memanggil” definisi YAMAL.

Kapan menggunakan tindakan ini

Gunakan tindakan ini jika Anda ingin menambahkan fungsionalitas ke alur kerja yang dienkapsulasi dalam, dan dilakukan oleh, fungsi Lambda.

Misalnya, Anda mungkin ingin alur kerja Anda mengirim `Build started` notifikasi ke saluran Slack sebelum memulai pembuatan aplikasi. Dalam hal ini, alur kerja Anda akan menyertakan tindakan AWS Lambda pemanggilan untuk memanggil Lambda untuk mengirimkan notifikasi Slack, dan tindakan build untuk membangun aplikasi Anda.

Sebagai contoh lain, Anda mungkin ingin alur kerja Anda melakukan pemindaian kerentanan pada aplikasi Anda sebelum diterapkan. Dalam hal ini, Anda akan menggunakan tindakan build untuk membangun aplikasi Anda, tindakan pemanggilan untuk AWS Lambda memanggil Lambda untuk memindai kerentanan, dan tindakan penerapan untuk menyebarkan aplikasi yang dipindai.

Topik

- Contoh alur kerja yang memanggil fungsi Lambda
- Menambahkan tindakan "AWS Lambda memanggil”
- Variabel yang dihasilkan oleh tindakan "AWS Lambda memanggil”
- Tindakan “AWS Lambda memanggil” definisi YAMAL

Contoh alur kerja yang memanggil fungsi Lambda

> **Note**
>
> Contoh alur kerja berikut adalah untuk tujuan ilustrasi saja, dan memerlukan pekerjaan penyiapan tambahan agar berfungsi dengan baik. Ini dimaksudkan untuk memberi Anda contoh seperti apa alur kerja ketika dikonfigurasi dengan tindakan AWS Lambda pemanggilan.

Alur kerja berikut mencakup tindakan AWS Lambda pemanggilan, bersama dengan tindakan penerapan. Alur kerja mengirimkan pemberitahuan Slack yang menunjukkan bahwa penerapan telah dimulai, dan kemudian menyebarkan aplikasi ke dalam AWS menggunakan templat. AWS CloudFormation Alur kerja terdiri dari blok bangunan berikut yang berjalan secara berurutan:

- Pemicu - Pemicu ini memulai alur kerja yang dijalankan secara otomatis saat Anda mendorong perubahan ke repositori sumber Anda. Untuk informasi lebih lanjut tentang menggunakan pemicu, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.
- Tindakan AWS Lambda pemanggilan (`LambdaNotify`) — Pada pemicu, tindakan ini memanggil fungsi `Notify-Start` Lambda di AWS akun dan Region yang ditentukan (`my-aws-account`, dan). `us-west-2` Pada pemanggilan, fungsi Lambda mengirimkan pemberitahuan Slack yang menunjukkan penerapan telah dimulai.
- Tindakan AWS CloudFormation tumpukan Deploy (`Deploy`) — Setelah menyelesaikan aksi AWS Lambda pemanggilan, tindakan AWS CloudFormation tumpukan Deploy menjalankan template (`cfn-template.yml`) untuk menerapkan tumpukan aplikasi Anda. Untuk informasi selengkapnya tentang tindakan Deploy AWS CloudFormation stack, lihatMenerapkan AWS CloudFormation tumpukan dengan alur kerja.

```
Name: codecatalyst-lamda-invoke-workflow
SchemaVersion: 1.0

Triggers:
  - Type: PUSH
    Branches:
      - main
Actions:
  LambdaNotify:
    Identifier: aws/lambda-invoke@v1
```

```
    Environment:
      Name: my-production-environment
      Connections:
        - Name: my-aws-account
          Role: codecatalyst-lambda-invoke-role
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      Function: Notify-Start
      AWSRegion: us-west-2

  Deploy:
    Identifier: aws/cfn-deploy@v1
    Environment:
      Name: my-production-environment
      Connections:
        - Name: my-aws-account
          Role: codecatalyst-deploy-role
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      name: my-application-stack
      region: us-west-2
      role-arn: arn:aws:iam::111122223333:role/StackRole
      template: ./cfn-template.yml
      capabilities: CAPABILITY_IAM,CAPABILITY_AUTO_EXPAND
```

Menambahkan tindakan "AWS Lambda memanggil”

Gunakan petunjuk berikut untuk menambahkan tindakan AWS Lambda pemanggilan ke alur kerja Anda.

Prasyarat

Sebelum memulai, pastikan AWS Lambda fungsi Anda dan peran eksekusi Lambda terkait sudah siap dan tersedia. AWS Untuk informasi selengkapnya, lihat topik peran eksekusi Lambda di Panduan AWS Lambda Pengembang.

Visual

Untuk menambahkan tindakan "AWS Lambda memanggil” menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan AWS Lambda pemanggilan, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih AWS Lambda panggil. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Lihat sumber untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Pada tab Input, Konfigurasi, dan Output, lengkapi bidang sesuai dengan kebutuhan Anda. Untuk deskripsi setiap bidang, lihatTindakan “AWS Lambda memanggil” definisi YAMAL. Referensi ini memberikan informasi rinci tentang setiap bidang (dan nilai properti YAMAL yang sesuai) seperti yang muncul di YAMAL dan editor visual.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menambahkan tindakan "AWS Lambda memanggil” menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.

2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan AWS Lambda pemanggilan, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih AWS Lambda panggil. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Lihat sumber untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Ubah properti dalam kode YAMAL sesuai dengan kebutuhan Anda. Penjelasan tentang setiap properti yang tersedia disediakan diTindakan “AWS Lambda memanggil” definisi YAMAL.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Variabel yang dihasilkan oleh tindakan "AWS Lambda memanggil”

Secara default, tindakan AWS Lambda pemanggilan menghasilkan satu variabel per kunci tingkat atas di payload respons Lambda.

Misalnya, jika payload respons terlihat seperti ini:

```
responsePayload = {
  "name": "Saanvi",
  "location": "Seattle",
```

```
  "department": {
    "company": "Amazon",
    "team": "AWS"
  }
}
```

... maka tindakan akan menghasilkan variabel berikut.

Kunci	Nilai
name	Saanvi
lokasi	Seattle
departemen	{"company": “Amazon”, “team”: "AWS“}

> **Note**
>
> Anda dapat mengubah variabel mana yang dihasilkan menggunakan properti `ResponseFilters` YAMAL. Untuk informasi lebih lanjut, lihat ResponseFilters diTindakan “AWS Lambda memanggil” definisi YAMAL.

Variabel yang dihasilkan dan diatur oleh tindakan "AWS Lambda memanggil” pada waktu berjalan dikenal sebagai variabel yang telah ditentukan.

Untuk informasi tentang mereferensikan variabel-variabel ini dalam alur kerja, lihat. Menggunakan variabel yang telah ditentukan

Tindakan “AWS Lambda memanggil” definisi YAMAL

Berikut ini adalah definisi YAMAL dari tindakan AWS Lambda pemanggilan. Untuk mempelajari cara menggunakan tindakan ini, lihatMemanggil AWS Lambda fungsi menggunakan alur kerja.

Definisi tindakan ini ada sebagai bagian dalam file definisi alur kerja yang lebih luas. Untuk informasi selengkapnya tentang file ini, lihatAlur kerja definisi YAMAL.

> **Note**
>
> Sebagian besar properti YAMAL yang mengikuti memiliki elemen UI yang sesuai di editor visual. Untuk mencari elemen UI, gunakan Ctrl+F. Elemen akan terdaftar dengan properti YAMLnya yang terkait.

```
# The workflow definition starts here.
# See Properti tingkat atas for details.

Name: MyWorkflow
SchemaVersion: 1.0
Actions:

# The action definition starts here.
  LambdaInvoke_nn:
    Identifier: aws/lambda-invoke@v1
    DependsOn:
      - dependent-action
    Compute:
      Type: EC2 | Lambda
      Fleet: fleet-name
    Timeout: timeout-minutes
    Inputs:
      # Specify a source or an artifact, but not both.
      Sources:
        - source-name-1
      Artifacts:
        - request-payload
      Variables:
        - Name: variable-name-1
          Value: variable-value-1
        - Name: variable-name-2
          Value: variable-value-2
    Environment:
      Name: environment-name
      Connections:
        - Name: account-connection-name
          Role: iam-role-name
    Configuration:
      Function: my-function|function-arn
      AWSRegion: us-west-2
```

```
      # Specify RequestPayload or RequestPayloadFile, but not both.
      RequestPayload: '{"firstname": "Li", lastname: "Jean", "company": "ExampleCo",
"team": "Development"}'
      RequestPayloadFile: my/request-payload.json
      ContinueOnError: true|false
      LogType: Tail|None
      ResponseFilters: '{"name": ".name", "company": ".department.company"}'
    Outputs:
      Artifacts:
        - Name: lambda_artifacts
          Files:
            - "lambda-response.json"
```

LambdaInvoke

(Diperlukan)

Tentukan nama tindakan. Semua nama tindakan harus unik dalam alur kerja. Nama aksi terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama tindakan.

Default: `Lambda_Invoke_Action_Workflow_nn`.

UI yang sesuai: Tab konfigurasi/Nama tindakan

Identifier

(*LambdaInvoke*/Identifier)

(Diperlukan)

Mengidentifikasi tindakan. Jangan mengubah properti ini kecuali Anda ingin mengubah versi. Untuk informasi selengkapnya, lihat Menentukan versi mayor, minor, atau patch dari suatu tindakan.

Default: `aws/lambda-invoke@v1`.

UI yang sesuai: Diagram alur LambdaInvoke kerja/_nn/ aws/lambda-invoke @v1 label

DependsOn

(*LambdaInvoke*/DependsOn)

(Opsional)

Tentukan tindakan, grup tindakan, atau gerbang yang harus berjalan dengan sukses agar tindakan ini berjalan.

Untuk informasi selengkapnya tentang fungsionalitas 'tergantung pada', lihat. Mengkonfigurasi tindakan untuk bergantung pada tindakan lain

UI yang sesuai: Tab masukan/Tergantung pada - opsional

Compute

(*LambdaInvoke*/Compute)

(Opsional)

Mesin komputasi yang digunakan untuk menjalankan tindakan alur kerja Anda. Anda dapat menentukan komputasi baik di tingkat alur kerja atau di tingkat tindakan, tetapi tidak keduanya. Ketika ditentukan pada tingkat alur kerja, konfigurasi komputasi berlaku untuk semua tindakan yang ditentukan dalam alur kerja. Pada tingkat alur kerja, Anda juga dapat menjalankan beberapa tindakan pada instance yang sama. Untuk informasi selengkapnya, lihat Berbagi komputasi di seluruh tindakan.

UI yang sesuai: tidak ada

Type

(*LambdaInvoke*/Compute/Type)

(Diperlukan Compute jika disertakan)

Jenis mesin komputasi. Anda dapat menggunakan salah satu nilai berikut:

- EC2 (editor visual) atau `EC2` (editor YAMG)

  Dioptimalkan untuk fleksibilitas selama aksi berjalan.
- Lambda (editor visual) atau `Lambda` (editor YAMG)

  Kecepatan start-up aksi yang dioptimalkan.

Untuk informasi selengkapnya tentang jenis komputasi, lihatJenis komputasi.

UI yang sesuai: Tab konfigurasi/Jenis komputasi

Fleet

(*LambdaInvoke*/Compute/Fleet)

(Opsional)

Tentukan mesin atau armada yang akan menjalankan alur kerja atau tindakan alur kerja Anda. Dengan armada sesuai permintaan, ketika suatu tindakan dimulai, alur kerja menyediakan sumber daya yang dibutuhkan, dan mesin dihancurkan ketika tindakan selesai. Contoh armada sesuai permintaan:`Linux.x86-64.Large`,. `Linux.x86-64.XLarge` Untuk informasi lebih lanjut tentang armada sesuai permintaan, lihat. Properti armada sesuai permintaan

Dengan armada yang disediakan, Anda mengonfigurasi satu set mesin khusus untuk menjalankan tindakan alur kerja Anda. Mesin-mesin ini tetap menganggur, siap untuk memproses tindakan segera. Untuk informasi lebih lanjut tentang armada yang disediakan, lihat. Properti armada yang disediakan

Jika `Fleet` dihilangkan, defaultnya adalah. `Linux.x86-64.Large`

UI yang sesuai: Tab konfigurasi/Armada komputasi

Timeout

(*LambdaInvoke*/Timeout)

(Diperlukan)

Tentukan jumlah waktu dalam menit (editor YAMAL), atau jam dan menit (editor visual), bahwa tindakan dapat berjalan sebelum CodeCatalyst mengakhiri tindakan. Minimal adalah 5 menit dan maksimum dijelaskan dalamKuota untuk alur kerja. Batas waktu default sama dengan batas waktu maksimum.

UI yang sesuai: Tab konfigurasi/Timeout - opsional

Inputs

(*LambdaInvoke*/Inputs)

(Diperlukan)

`Inputs`Bagian ini mendefinisikan data yang dibutuhkan tindakan AWS Lambda pemanggilan selama menjalankan alur kerja.

> **Note**
>
> Hanya satu input (baik sumber atau artefak) yang diizinkan per tindakan AWS Lambda pemanggilan. Variabel tidak dihitung terhadap total ini.

UI yang sesuai: Tab input

Sources

(*`LambdaInvoke`*/Inputs/Sources)

(Diperlukan jika RequestPayloadFiledisediakan)

Jika Anda ingin meneruskan file JSON payload permintaan AWS Lambda ke tindakan pemanggilan, dan file payload ini disimpan dalam repositori sumber, tentukan label repositori sumber tersebut. Saat ini, satu-satunya label yang didukung adalah`WorkflowSource`.

Jika file payload permintaan Anda tidak terkandung dalam repositori sumber, file tersebut harus berada dalam artefak yang dihasilkan oleh tindakan lain.

Untuk informasi selengkapnya tentang file payload, lihatRequestPayloadFile.

> **Note**
>
> Alih-alih menentukan file payload, Anda dapat menambahkan kode JSON payload langsung ke tindakan menggunakan properti. `RequestPayload` Untuk informasi selengkapnya, lihat RequestPayload.

Untuk informasi selengkapnya tentang sumber, lihat Menghubungkan alur kerja ke repositori sumber.

UI yang sesuai: Tab/Sumber Input - opsional

Artifacts - input

(*`LambdaInvoke`*/Inputs/Artifacts)

(Diperlukan jika RequestPayloadFiledisediakan)

Jika Anda ingin meneruskan file JSON payload permintaan AWS Lambda ke tindakan pemanggilan, dan file payload ini terdapat dalam artefak keluaran dari tindakan sebelumnya, tentukan artefak tersebut di sini.

Untuk informasi selengkapnya tentang file payload, lihatRequestPayloadFile.

> **Note**
>
> Alih-alih menentukan file payload, Anda dapat menambahkan kode JSON payload langsung ke tindakan menggunakan properti. `RequestPayload` Untuk informasi selengkapnya, lihat RequestPayload.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab konfigurasi/Artefak - opsional

Variables - input

(*LambdaInvoke*/Inputs/Variables)

(Opsional)

Tentukan urutan pasangan nama/nilai yang menentukan variabel input yang ingin Anda sediakan untuk tindakan. Nama variabel terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama variabel.

Untuk informasi selengkapnya tentang variabel, termasuk contoh, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

UI yang sesuai: Input tab/Variabel - opsional

Environment

(*LambdaInvoke*/Environment)

(Diperlukan)

Tentukan CodeCatalyst lingkungan yang akan digunakan dengan tindakan.

Untuk informasi lebih lanjut tentang lingkungan, lihat Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst danPembuatan lingkungan.

UI yang sesuai: Tab Konfigurasi/'Lingkungan/Akun/Peran'/Lingkungan

Name

(*LambdaInvoke*/Environment/Name)

(Diperlukan Environment jika disertakan)

Tentukan nama lingkungan yang ada yang ingin Anda kaitkan dengan tindakan.

UI yang sesuai: Tab Konfigurasi/'lingkungan/koneksi/peran'/Lingkungan

Connections

(*LambdaInvoke*/Environment/Connections)

(Diperlukan Environment jika disertakan)

Tentukan koneksi akun untuk dikaitkan dengan tindakan. Anda dapat menentukan maksimum satu koneksi akun di bawah`Environment`.

Untuk informasi selengkapnya tentang koneksi akun, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS. Untuk informasi tentang cara mengaitkan koneksi akun dengan lingkungan Anda, lihatPembuatan lingkungan.

UI yang sesuai: Tab Konfigurasi/'lingkungan/koneksi/peran'/ Koneksi

Name

(*LambdaInvoke*/Environment/Connections/Name)

(Diperlukan)

Tentukan nama koneksi akun.

UI yang sesuai: Tab Konfigurasi/'lingkungan/koneksi/peran'/ Koneksi

Role

(*LambdaInvoke*/Environment/Connections/Role)

(Diperlukan)

Tentukan nama peran IAM yang digunakan tindakan AWS Lambda pemanggilan untuk mengakses AWS dan menjalankan fungsi Lambda Anda. Pastikan bahwa peran ini meliputi:

- Kebijakan izin berikut:

> **Warning**
>
> Batasi izin untuk yang ditampilkan dalam kebijakan berikut. Menggunakan peran dengan izin yang lebih luas dapat menimbulkan risiko keamanan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "lambda:InvokeFunction",
            "Resource": "arn:aws:lambda:aws-region:aws-account:function:function-
name"
        }
    ]
}
```

- Kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
```

```
                    "codecatalyst-runner.amazonaws.com",
                    "codecatalyst.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Pastikan peran ini terkait dengan koneksi akun Anda. Untuk mempelajari lebih lanjut tentang mengaitkan peran IAM dengan koneksi akun, lihat. Menambahkan peran IAM ke koneksi akun

> **Note**
>
> Anda dapat menentukan nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran di sini, jika Anda mau. Untuk informasi selengkapnya tentang peran ini, silakan lihat Membuat CodeCatalystWorkflowDevelopmentRole-***`spaceName`***peran untuk akun dan ruang Anda. Pahami bahwa `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran tersebut memiliki izin yang sangat luas yang dapat menimbulkan risiko keamanan. Kami menyarankan Anda hanya menggunakan peran ini dalam tutorial dan skenario di mana keamanan kurang menjadi perhatian.

UI yang sesuai: Tab Konfigurasi/'lingkungan/koneksi/peran'/Peran

Configuration

(*`LambdaInvoke`*/Configuration)

(Diperlukan)

Bagian di mana Anda dapat menentukan properti konfigurasi tindakan.

UI yang sesuai: Tab konfigurasi

Function

(*`LambdaInvoke`*/Configuration/Function)

(Diperlukan)

Tentukan AWS Lambda fungsi yang akan dipanggil tindakan ini. Anda dapat menentukan nama fungsi, atau Nama Sumber Daya Amazon (ARN). Anda dapat menemukan nama atau ARN di konsol Lambda.

> **Note**
>
> AWS Akun tempat fungsi Lambda berada dapat berbeda dari akun yang ditentukan di bawah. `Connections:`

UI yang sesuai: Tab/Fungsi Konfigurasi

AWSRegion

(*LambdaInvoke*/Configuration/AWSRegion)

(Diperlukan)

Tentukan AWS Wilayah tempat AWS Lambda fungsi Anda berada. Untuk daftar kode Region, lihat Titik akhir Regional di. Referensi Umum AWS

UI yang sesuai: Tab konfigurasi/ember Tujuan - opsional

RequestPayload

(*LambdaInvoke*/Configuration/RequestPayload)

(Opsional)

Jika Anda ingin meneruskan payload permintaan AWS Lambda ke tindakan pemanggilan, tentukan payload permintaan di sini, dalam format JSON.

Contoh permintaan payload:

```
'{ "key": "value" }'
```

Jika Anda tidak ingin meneruskan payload permintaan ke fungsi Lambda Anda, maka hilangkan properti ini.

> **Note**
>
> Anda dapat menentukan salah satu dari `RequestPayload` atau `RequestPayloadFile`, bukan keduanya.

Untuk informasi selengkapnya tentang payload permintaan, lihat topik Memanggil di Referensi AWS Lambda API.

UI yang sesuai: Tab konfigurasi/Minta muatan - opsional

RequestPayloadFile

(*LambdaInvoke*/Configuration/RequestPayloadFile)

(Opsional)

Jika Anda ingin meneruskan payload permintaan AWS Lambda ke tindakan pemanggilan, tentukan jalur ke file payload permintaan ini di sini. File harus dalam format JSON.

File payload permintaan dapat berada di repositori sumber atau artefak dari tindakan sebelumnya. Jalur file relatif terhadap repositori sumber atau root artefak.

Jika Anda tidak ingin meneruskan payload permintaan ke fungsi Lambda Anda, maka hilangkan properti ini.

> **Note**
>
> Anda dapat menentukan salah satu dari `RequestPayload` atau `RequestPayloadFile`, bukan keduanya.

Untuk informasi selengkapnya tentang file payload permintaan, lihat topik Memanggil di Referensi AWS Lambda API.

UI yang sesuai: Tab konfigurasi/Minta file payload - opsional

ContinueOnError

(*LambdaInvoke*/Configuration/RequestPayloadFile)

(Opsional)

Tentukan apakah Anda ingin menandai tindakan AWS Lambda pemanggilan sebagai berhasil meskipun fungsi yang dipanggil AWS Lambda gagal. Pertimbangkan untuk menyetel properti ini `true` agar tindakan selanjutnya dalam alur kerja Anda dimulai meskipun Lambda gagal.

Defaultnya adalah gagal tindakan jika fungsi Lambda gagal ("mati" di editor visual atau `false` di editor YAMAL).

UI yang sesuai: Tab konfigurasi/Lanjutkan kesalahan

LogType

(*`LambdaInvoke`*/Configuration/LogType)

(Opsional)

Tentukan apakah Anda ingin menyertakan log kesalahan dalam respons dari fungsi Lambda setelah dipanggil. Anda dapat melihat log ini di tab Log aksi pemanggilan Lambda di konsol. CodeCatalyst Kemungkinan nilainya adalah:

- `Tail`— kembalikan log
- `None`— jangan kembalikan log

Defaultnya adalah Tail.

Untuk informasi selengkapnya tentang jenis log, lihat topik Memanggil di Referensi AWS Lambda API.

Untuk informasi selengkapnya tentang melihat log, lihat Melihat alur kerja menjalankan status dan detail.

UI yang sesuai: Tab konfigurasi/Jenis log

ResponseFilters

(*`LambdaInvoke`*/Configuration/ResponseFilters)

(Opsional)

Tentukan kunci mana di payload respons Lambda yang ingin Anda konversi ke variabel keluaran. Anda kemudian dapat mereferensikan variabel output dalam tindakan selanjutnya dalam alur kerja Anda. Untuk informasi lebih lanjut tentang variabel di CodeCatalyst, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

Misalnya, jika payload respons Anda terlihat seperti ini:

```
responsePayload = {
  "name": "Saanvi",
  "location": "Seattle",
  "department": {
    "company": "Amazon",
    "team": "AWS"
  }
}
```

... dan filter respons Anda terlihat seperti ini:

```
Configuration:
  ...
  ResponseFilters: '{"name": ".name", "company": ".department.company"}'
```

... maka tindakan menghasilkan variabel output berikut:

Kunci	Nilai
name	Saanvi
perusahaan	Amazon

Anda kemudian dapat mereferensikan `company` variabel `name` dan dalam tindakan selanjutnya.

Jika Anda tidak menentukan kunci apa pun`ResponseFilters`, maka tindakan akan mengubah setiap kunci tingkat atas dalam respons Lambda menjadi variabel keluaran. Untuk informasi selengkapnya, lihat Variabel yang dihasilkan oleh tindakan "AWS Lambda memanggil” .

Pertimbangkan untuk menggunakan filter respons untuk membatasi variabel keluaran yang dihasilkan hanya untuk variabel yang benar-benar ingin Anda gunakan.

UI yang sesuai: Tab konfigurasi/Filter respons - opsional

Outputs

(*`LambdaInvoke`*/Outputs)

(Opsional)

Mendefinisikan data yang dihasilkan oleh tindakan selama menjalankan alur kerja.

UI yang sesuai: Tab keluaran

Artifacts

(*LambdaInvoke*/Outputs/Artifacts)

(Opsional)

Tentukan artefak yang dihasilkan oleh tindakan. Anda dapat mereferensikan artefak ini sebagai masukan dalam tindakan lain.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab Keluaran/Artefak/Bangun nama artefak

Name

(*LambdaInvoke*/Outputs/Artifacts/Name)

(Opsional)

Tentukan nama artefak yang akan berisi muatan respons Lambda yang dikembalikan oleh fungsi Lambda. Nilai default-nya adalah `lambda_artifacts`. Jika Anda tidak menentukan artefak, maka payload respons Lambda dapat dilihat di log tindakan, yang tersedia di tab Log untuk tindakan di konsol. CodeCatalyst Untuk informasi selengkapnya tentang melihat log, lihat Melihat alur kerja menjalankan status dan detail.

UI yang sesuai: Tab Keluaran/Artefak/Bangun nama artefak

Files

(*LambdaInvoke*/Outputs/Artifacts/Files)

(Opsional)

Tentukan file yang akan disertakan dalam artefak. Anda harus menentukan `lambda-response.json` sehingga file payload respons Lambda akan disertakan.

UI yang sesuai: Mengeluarkan Tab/Artefak/File yang dihasilkan oleh build

Memodifikasi file definisi tugas Amazon ECS menggunakan alur kerja

Bagian ini menjelaskan cara memperbarui `image` bidang dalam file definisi tugas Amazon Elastic Container Service (Amazon ECS) menggunakan alur kerja. CodeCatalyst Untuk mencapai ini, Anda harus menambahkan tindakan definisi tugas Render Amazon ECS ke alur kerja Anda. Tindakan ini memperbarui bidang gambar dalam file definisi tugas dengan nama gambar Docker yang disediakan oleh alur kerja Anda saat runtime.

> **Note**
>
> Anda juga dapat menggunakan tindakan ini untuk memperbarui `environment` bidang definisi tugas dengan variabel lingkungan.

Kapan menggunakan tindakan ini

Gunakan ini jika Anda memiliki alur kerja yang membangun dan menandai image Docker dengan konten dinamis, seperti ID komit atau stempel waktu.

Jangan gunakan tindakan ini jika file definisi tugas Anda berisi nilai gambar yang selalu tetap sama. Dalam hal ini, Anda dapat memasukkan nama gambar Anda secara manual ke dalam file definisi tugas.

Cara kerja tindakan “Render Amazon ECS task definition”

Anda harus menggunakan tindakan definisi tugas Render Amazon ECS dengan tindakan build dan Deploy ke Amazon ECS di alur kerja Anda. Bersama-sama, tindakan ini bekerja sebagai berikut:

1. Tindakan build membangun image Docker Anda dan menandainya dengan nama, ID komit, stempel waktu, atau konten dinamis lainnya. Misalnya, tindakan build Anda mungkin terlihat seperti ini:

```
MyECSWorkflow
  Actions:
    BuildAction:
```

```
      Identifier: aws/build@v1
      ...
      Configuration:
        Steps:
        # Build, tag, and push the Docker image...
          - Run: docker build -t MyDockerImage:${WorkflowSource.CommitId} .
          ...
```

Dalam kode sebelumnya, `docker build -t` arahan menunjukkan untuk membangun image Docker dan menandainya dengan ID komit saat runtime tindakan. Nama gambar yang dihasilkan mungkin terlihat seperti ini:

`MyDockerImage:a37bd7e`

2. Tindakan definisi tugas Render Amazon ECS menambahkan nama gambar yang dihasilkan secara dinamis`MyDockerImage:a37bd7e`, ke file definisi tugas Anda, seperti ini:

```
{
    "executionRoleArn": "arn:aws:iam::account_ID:role/codecatalyst-ecs-task-
execution-role",
    "containerDefinitions": [
        {
            "name": "codecatalyst-ecs-container",
            "image":  MyDockerImage:a37bd7e,
            "essential": true,
            ...
            "portMappings": [
                {
                    "hostPort": 80,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ]
        }
    ],
...
}
```

Secara opsional, Anda juga dapat meminta tindakan definisi tugas Render Amazon ECS menambahkan variabel lingkungan ke definisi tugas, seperti ini:

```
{
```

```
  "executionRoleArn": "arn:aws:iam::account_ID:role/codecatalyst-ecs-task-execution-
role",
  "containerDefinitions": [
    {
      "name": "codecatalyst-ecs-container",
      "image":  MyDockerImage:a37bd7e,
      ...
      "environment": [
        {
          name": "ECS_LOGLEVEL",
          value": "info"
        }
      ]
    }
  ],
...
}
```

Untuk informasi selengkapnya tentang variabel lingkungan, lihat Menentukan variabel lingkungan di Panduan Pengembang Layanan Amazon Elastic Container.

3. Tindakan Deploy to Amazon ECS mendaftarkan file definisi tugas yang diperbarui dengan Amazon ECS. Mendaftarkan file definisi tugas yang diperbarui menyebarkan gambar baru, `MyDockerImage:a37bd7e` ke Amazon ECS.

Topik

- Contoh alur kerja yang memodifikasi file definisi tugas Amazon ECS
- Menambahkan tindakan “Render definisi tugas Amazon ECS”
- Melihat file definisi tugas yang diperbarui
- Variabel yang dihasilkan oleh tindakan “Render Amazon ECS task definition”
- Referensi definisi tindakan YAMM “Render Amazon ECS”

Contoh alur kerja yang memodifikasi file definisi tugas Amazon ECS

Berikut ini adalah contoh alur kerja lengkap yang menyertakan tindakan definisi tugas Render Amazon ECS, bersama dengan tindakan build dan deploy. Tujuan alur kerja adalah untuk membangun dan menerapkan image Docker ke dalam klaster Amazon ECS Anda. Alur kerja terdiri dari blok bangunan berikut yang berjalan secara berurutan:

- Pemicu - Pemicu ini memulai alur kerja yang dijalankan secara otomatis saat Anda mendorong perubahan ke repositori sumber Anda. Untuk informasi lebih lanjut tentang menggunakan pemicu, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.
- Tindakan build (`BuildDocker`) — Pada pemicu, aksi membangun image Docker menggunakan Dockerfile, memberi tag dengan ID komit, dan mendorong gambar ke Amazon ECR. Untuk informasi selengkapnya tentang tindakan build, lihatMembangun dengan alur kerja.
- Tindakan definisi tugas Amazon ECS Render (`RenderTaskDef`) — Setelah menyelesaikan tindakan build, tindakan ini memperbarui yang ada di `taskdef.json` root repositori sumber Anda dengan nilai `image` bidang yang menyertakan ID komit yang benar. Ini menyimpan file yang diperbarui dengan nama file baru (`task-definition-`*`random-string`*`.json`) dan kemudian membuat artefak keluaran yang berisi file ini. Tindakan render juga menghasilkan variabel yang disebut `task-definition` dan menyetelnya ke nama file definisi tugas baru. Artefak dan variabel akan digunakan tindakan penyebaran, yang berikutnya.
- Tindakan Deploy to Amazon ECS (`DeployToECS`) — Setelah menyelesaikan tindakan definisi tugas Render Amazon ECS, aksi Deploy to Amazon ECS mencari artefak keluaran yang dihasilkan oleh tindakan render `TaskDefArtifact` (), menemukan `task-definition-`*`random-string`*`.json` file di dalamnya, dan mendaftarkannya dengan layanan Amazon ECS Anda. Layanan Amazon ECS kemudian mengikuti petunjuk dalam `task-definition-`*`random-string`*`.json` file untuk menjalankan tugas Amazon ECS—dan container image Docker terkait—di dalam cluster Amazon ECS Anda.

```
Name: codecatalyst-ecs-workflow
SchemaVersion: 1.0

Triggers:
  - Type: PUSH
    Branches:
      - main
Actions:
  BuildDocker:
    Identifier: aws/build@v1
    Environment:
      Name: codecatalyst-ecs-environment
      Connections:
        - Name: codecatalyst-account-connection
          Role: codecatalyst-ecs-build-role
    Inputs:
      Variables:
```

```
        - Name: REPOSITORY_URI
          Value: 111122223333.dkr.ecr.us-east-2.amazonaws.com/codecatalyst-ecs-image-
repo
        - Name: IMAGE_TAG
          Value: ${WorkflowSource.CommitId}
    Configuration:
      Steps:
        #pre_build:
        - Run: echo Logging in to Amazon ECR...
        - Run: aws --version
        - Run: aws ecr get-login-password --region us-east-2 | docker login --username
 AWS --password-stdin 111122223333.dkr.ecr.us-east-2.amazonaws.com
        #build:
        - Run: echo Build started on `date`
        - Run: echo Building the Docker image...
        - Run: docker build -t $REPOSITORY_URI:latest .
        - Run: docker tag $REPOSITORY_URI:latest $REPOSITORY_URI:$IMAGE_TAG
        #post_build:
        - Run: echo Build completed on `date`
        - Run: echo Pushing the Docker images...
        - Run: docker push $REPOSITORY_URI:latest
        - Run: docker push $REPOSITORY_URI:$IMAGE_TAG

  RenderTaskDef:
    DependsOn:
      - BuildDocker
    Identifier: aws/ecs-render-task-definition@v1
    Inputs:
      Variables:
        - Name: REPOSITORY_URI
          Value: 111122223333.dkr.ecr.us-east-2.amazonaws.com/codecatalyst-ecs-image-
repo
        - Name: IMAGE_TAG
          Value: ${WorkflowSource.CommitId}
    Configuration:
      task-definition: taskdef.json
      container-definition-name: codecatalyst-ecs-container
      image: $REPOSITORY_URI:$IMAGE_TAG
    # The output artifact contains the updated task definition file.
    # The new file is prefixed with 'task-definition'.
    # The output variable is set to the name of the updated task definition file.
    Outputs:
      Artifacts:
        - Name: TaskDefArtifact
```

```
        Files:
          - "task-definition*"
    Variables:
      - task-definition

DeployToECS:
  Identifier: aws/ecs-deploy@v1
  Environment:
    Name: codecatalyst-ecs-environment
    Connections:
      - Name: codecatalyst-account-connection
        Role: codecatalyst-ecs-deploy-role
  #Input artifact contains the updated task definition file.
  Inputs:
    Sources: []
    Artifacts:
      - TaskDefArtifact
  Configuration:
    region: us-east-2
    cluster: codecatalyst-ecs-cluster
    service: codecatalyst-ecs-service
    task-definition: ${RenderTaskDef.task-definition}
```

Menambahkan tindakan “Render definisi tugas Amazon ECS”

Gunakan petunjuk berikut untuk menambahkan tindakan definisi tugas Render Amazon ECS ke alur kerja Anda.

Prasyarat

Sebelum memulai, pastikan Anda memiliki alur kerja yang menyertakan tindakan build yang menghasilkan image Docker secara dinamis. Lihat alur kerja contoh sebelumnya untuk detailnya.

Visual

Untuk menambahkan tindakan “Render Amazon ECS task definition” menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.

5. Pilih Edit.
6. Pilih Visual.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.
8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan definisi tugas Render Amazon ECS, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih Render definisi tugas Amazon ECS. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Lihat sumber untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Pada tab Input dan Konfigurasi, lengkapi bidang sesuai dengan kebutuhan Anda. Untuk deskripsi setiap bidang, lihatReferensi definisi tindakan YAMM “Render Amazon ECS”. Referensi ini memberikan informasi rinci tentang setiap bidang (dan nilai properti YAMAL yang sesuai) seperti yang muncul di YAMAL dan editor visual.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menambahkan tindakan “Render definisi tugas Amazon ECS” menggunakan editor YAMAL

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Di kiri atas, pilih + Tindakan untuk membuka katalog tindakan.

8. Dari daftar drop-down, pilih Amazon CodeCatalyst.
9. Cari tindakan definisi tugas Render Amazon ECS, dan lakukan salah satu hal berikut:

   - Pilih tanda plus (+) untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.

     Atau

   - Pilih Render definisi tugas Amazon ECS. Kotak dialog detail tindakan muncul. Pada kotak dialog ini:
     - (Opsional) Pilih Lihat sumber untuk melihat kode sumber tindakan.
     - Pilih Tambahkan ke alur kerja untuk menambahkan tindakan ke diagram alur kerja dan buka panel konfigurasinya.
10. Ubah properti dalam kode YAMAL sesuai dengan kebutuhan Anda. Penjelasan tentang setiap properti yang tersedia disediakan diReferensi definisi tindakan YAMM “Render Amazon ECS”.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Langkah selanjutnya

Setelah menambahkan tindakan render, tambahkan tindakan Deploy ke Amazon ECS ke alur kerja Anda mengikuti petunjuk di. Menyebarkan aplikasi ke Amazon Elastic Container Service (ECS) dengan alur kerja Saat menambahkan tindakan penerapan, lakukan hal berikut:

1. Di tab Input dari tindakan penerapan, di Artefak - opsional, pilih artefak yang dihasilkan oleh tindakan render. Ini berisi file definisi tugas yang diperbarui.

   Untuk informasi lebih lanjut tentang artifact, lihat Berbagi data antar tindakan dalam alur kerja menggunakan artefak.
2. Di tab Konfigurasi tindakan penerapan, di bidang Definisi tugas, tentukan variabel tindakan berikut: `${action-name.task-definition}` di mana nama *tindakan adalah nama* tindakan render Anda, misalnya,. `RenderTaskDef` Tindakan render menetapkan variabel ini ke nama baru dari file definisi tugas.

   Untuk informasi lebih lanjut tentang variabel, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

Untuk informasi selengkapnya tentang cara mengonfigurasi tindakan penerapan, lihat alur kerja contoh sebelumnya.

Melihat file definisi tugas yang diperbarui

Anda dapat melihat nama dan isi file definisi tugas yang diperbarui.

Untuk melihat nama file definisi tugas yang diperbarui, setelah tindakan definisi tugas Render Amazon ECS memprosesnya.

1. Temukan proses yang menyertakan tindakan render yang telah selesai:
    a. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
    b. Pilih proyek Anda.
    c. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
    d. Pilih nama alur kerja yang berisi tindakan render. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
    e. Pilih run yang menyertakan tindakan render selesai.
2. Dalam diagram alur kerja, pilih tindakan render.
3. Pilih Output.
4. Pilih Variabel.
5. Nama file definisi tugas ditampilkan. Ini terlihat mirip dengan`task-definition--259-0a2r7gxlTF5X-.json`.

Untuk melihat isi file definisi tugas yang diperbarui

1. Temukan proses yang menyertakan tindakan render yang telah selesai:
    a. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
    b. Pilih proyek Anda.
    c. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
    d. Pilih nama alur kerja yang berisi tindakan render. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.

e. Pilih run yang menyertakan tindakan render selesai.

2. Dalam alur kerja berjalan, di bagian atas, di samping Visual dan YAMAL, pilih Output alur kerja.
3. Di bagian Artefak, pilih Unduh di samping artefak yang berisi file definisi tugas yang diperbarui. Artefak ini akan memiliki kolom Diproduksi oleh disetel ke nama tindakan render Anda.
4. Buka file.zip untuk melihat file definisi tugas .json.

Variabel yang dihasilkan oleh tindakan “Render Amazon ECS task definition”

Tindakan definisi tugas Render Amazon ECS menghasilkan dan menetapkan variabel berikut pada waktu berjalan. Ini dikenal sebagai variabel yang telah ditentukan.

Untuk informasi tentang mereferensikan variabel-variabel ini dalam alur kerja, lihat. Menggunakan variabel yang telah ditentukan

Kunci	Nilai
definisi tugas	Nama yang diberikan ke file definisi tugas yang diperbarui oleh tindakan definisi tugas Amazon ECS Render. Nama mengikuti format`task-definition- random-string .json`. Contoh: `task-definition--259-0a2r7g xlTF5Xr.json`

Referensi definisi tindakan YAMM “Render Amazon ECS”

Berikut ini adalah definisi YAMAL dari tindakan definisi tugas Render Amazon ECS. Untuk mempelajari cara menggunakan tindakan ini, lihatMemodifikasi file definisi tugas Amazon ECS menggunakan alur kerja.

Definisi tindakan ini ada sebagai bagian dalam file definisi alur kerja yang lebih luas. Untuk informasi selengkapnya tentang file ini, lihatAlur kerja definisi YAMAL.

> **Note**
>
> Sebagian besar properti YAMM yang mengikuti memiliki elemen UI yang sesuai di editor visual. Untuk mencari elemen UI, gunakan Ctrl+F. Elemen akan terdaftar dengan properti YAMLnya yang terkait.

```yaml
# The workflow definition starts here.
# See Properti tingkat atas for details.

Name: MyWorkflow
SchemaVersion: 1.0
Actions:

# The action definition starts here.
  ECSRenderTaskDefinition_nn:
    Identifier: aws/ecs-render-task-definition@v1
    DependsOn:
      - build-action
    Compute:
      Type: EC2 | Lambda
      Fleet: fleet-name
    Timeout: timeout-minutes
    Inputs:
      # Specify a source or an artifact, but not both.
      Sources:
        - source-name-1
      Artifacts:
        - task-definition-artifact
      Variables:
        - Name: variable-name-1
          Value: variable-value-1
        - Name: variable-name-2
          Value: variable-value-2
    Configuration
      task-definition: task-definition-path
      container-definition-name: container-definition-name
      image: docker-image-name
      environment-variables:
        - variable-name-1=variable-value-1
        - variable-name-2=variable-value-2
    Outputs:
```

```
      Artifacts:
        - Name: TaskDefArtifact
          Files: "task-definition*"
      Variables:
        - task-definition
```

ECSRenderTaskDefinition

(Diperlukan)

Tentukan nama tindakan. Semua nama tindakan harus unik dalam alur kerja. Nama aksi terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama tindakan.

Default: `ECSRenderTaskDefinition_nn`.

UI yang sesuai: Tab konfigurasi/Nama tindakan

Identifier

(*ECSRenderTaskDefinition*/Identifier)

(Diperlukan)

Mengidentifikasi tindakan. Jangan mengubah properti ini kecuali Anda ingin mengubah versi. Untuk informasi selengkapnya, lihat Menentukan versi mayor, minor, atau patch dari suatu tindakan.

Default: `aws/ecs-render-task-definition@v1`.

UI yang sesuai: Diagram alur ECSRenderTaskDefinition kerja/_nn/ aws/ @v1 label ecs-render-task-definition

DependsOn

(*ECSRenderTaskDefinition*/DependsOn)

(Opsional)

Tentukan tindakan, grup tindakan, atau gerbang yang harus berjalan dengan sukses agar tindakan ini berjalan.

Untuk informasi selengkapnya tentang fungsionalitas 'tergantung pada', lihat. Mengkonfigurasi tindakan untuk bergantung pada tindakan lain

UI yang sesuai: Tab masukan/Tergantung pada - opsional

Compute

(*ECSRenderTaskDefinition*/Compute)

(Opsional)

Mesin komputasi yang digunakan untuk menjalankan tindakan alur kerja Anda. Anda dapat menentukan komputasi baik di tingkat alur kerja atau di tingkat tindakan, tetapi tidak keduanya. Ketika ditentukan pada tingkat alur kerja, konfigurasi komputasi berlaku untuk semua tindakan yang ditentukan dalam alur kerja. Pada tingkat alur kerja, Anda juga dapat menjalankan beberapa tindakan pada instance yang sama. Untuk informasi selengkapnya, lihat Berbagi komputasi di seluruh tindakan.

UI yang sesuai: tidak ada

Type

(*ECSRenderTaskDefinition*/Compute/Type)

(Diperlukan Compute jika disertakan)

Jenis mesin komputasi. Anda dapat menggunakan salah satu nilai berikut:

- EC2 (editor visual) atau `EC2` (editor YAMG)

  Dioptimalkan untuk fleksibilitas selama aksi berjalan.
- Lambda (editor visual) atau `Lambda` (editor YAMG)

  Kecepatan start-up aksi yang dioptimalkan.

Untuk informasi selengkapnya tentang jenis komputasi, lihatJenis komputasi.

UI yang sesuai: Tab konfigurasi/Jenis komputasi

Fleet

(*ECSRenderTaskDefinition*/Compute/Fleet)

(Opsional)

Tentukan mesin atau armada yang akan menjalankan alur kerja atau tindakan alur kerja Anda. Dengan armada sesuai permintaan, ketika suatu tindakan dimulai, alur kerja menyediakan sumber daya yang dibutuhkan, dan mesin dihancurkan ketika tindakan selesai. Contoh armada sesuai permintaan:`Linux.x86-64.Large`,. `Linux.x86-64.XLarge` Untuk informasi lebih lanjut tentang armada sesuai permintaan, lihat. Properti armada sesuai permintaan

Dengan armada yang disediakan, Anda mengonfigurasi satu set mesin khusus untuk menjalankan tindakan alur kerja Anda. Mesin-mesin ini tetap menganggur, siap untuk memproses tindakan segera. Untuk informasi lebih lanjut tentang armada yang disediakan, lihat. Properti armada yang disediakan

Jika `Fleet` dihilangkan, defaultnya adalah. `Linux.x86-64.Large`

UI yang sesuai: Tab konfigurasi/Armada komputasi

Timeout

(*`ECSRenderTaskDefinition`*/Timeout)

(Opsional)

Tentukan jumlah waktu dalam menit (editor YAMM), atau jam dan menit (editor visual), bahwa tindakan dapat berjalan sebelum CodeCatalyst mengakhiri tindakan. Minimal adalah 5 menit dan maksimum dijelaskan dalamKuota untuk alur kerja. Batas waktu default sama dengan batas waktu maksimum.

UI yang sesuai: Tab konfigurasi/Timeout - opsional

Inputs

(*`ECSRenderTaskDefinition`*/Inputs)

(Opsional)

`Inputs`Bagian ini mendefinisikan data yang `ECSRenderTaskDefinition` dibutuhkan selama menjalankan alur kerja.

> **Note**
>
> Hanya satu input (baik sumber atau artefak) yang diizinkan per tindakan definisi tugas Amazon ECS Render. Variabel tidak dihitung terhadap total ini.

UI yang sesuai: Tab input

Sources

(*ECSRenderTaskDefinition*/Inputs/Sources)

(Diperlukan jika file definisi tugas Anda disimpan dalam repositori sumber)

Jika file definisi tugas Anda disimpan dalam repositori sumber, tentukan label repositori sumber tersebut. Saat ini, satu-satunya label yang didukung adalah`WorkflowSource`.

Jika file definisi tugas Anda tidak terkandung dalam repositori sumber, itu harus berada dalam artefak yang dihasilkan oleh tindakan lain.

Untuk informasi selengkapnya tentang sumber, lihat Menghubungkan alur kerja ke repositori sumber.

UI yang sesuai: Tab/Sumber Input - opsional

Artifacts - input

(*ECSRenderTaskDefinition*/Inputs/Artifacts)

(Diperlukan jika file definisi tugas Anda disimpan dalam artefak keluaran dari tindakan sebelumnya)

Jika file definisi tugas yang ingin Anda terapkan terkandung dalam artefak yang dihasilkan oleh tindakan sebelumnya, tentukan artefak tersebut di sini. Jika file definisi tugas Anda tidak terkandung dalam artefak, itu harus berada di repositori sumber Anda.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab konfigurasi/Artefak - opsional

Variables - input

(*ECSRenderTaskDefinition*/Inputs/Variables)

(Diperlukan)

Tentukan urutan pasangan nama/nilai yang menentukan variabel input yang ingin Anda sediakan untuk tindakan. Nama variabel terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung

(-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama variabel.

Untuk informasi selengkapnya tentang variabel, termasuk contoh, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

UI yang sesuai: Input tab/Variabel - opsional

Configuration

(*ECSRenderTaskDefinition*/Configuration)

(Diperlukan)

Bagian di mana Anda dapat menentukan properti konfigurasi tindakan.

UI yang sesuai: Tab konfigurasi

task-definition

(*ECSRenderTaskDefinition*/Configuration/task-definition)

(Diperlukan)

Tentukan jalur ke file definisi tugas yang ada. Jika file berada di repositori sumber Anda, jalurnya relatif terhadap folder root repositori sumber. Jika file Anda berada dalam artefak dari tindakan alur kerja sebelumnya, jalurnya relatif terhadap folder root artefak. Untuk informasi selengkapnya tentang file definisi tugas, lihat Definisi tugas di Panduan Pengembang Layanan Kontainer Elastis Amazon.

UI yang sesuai: Tab konfigurasi/Definisi tugas

container-definition-name

(*ECSRenderTaskDefinition*/Configuration/container-definition-name)

(Diperlukan)

Tentukan nama wadah tempat image Docker Anda akan berjalan. Anda dapat menemukan nama ini di `name` bidang`containerDefinitions`, di file definisi tugas Anda. Untuk informasi selengkapnya, lihat Nama di Panduan Pengembang Layanan Kontainer Elastis Amazon.

UI yang sesuai: Tab konfigurasi/Nama kontainer

image

(*ECSRenderTaskDefinition*/Configuration/image)

(Diperlukan)

Tentukan nama gambar Docker yang ingin ditambahkan tindakan definisi tugas Render Amazon ECS ke file definisi tugas Anda. Tindakan menambahkan nama ini ke `image` bidang`containerDefinitions`, dalam file definisi tugas Anda. Jika nilai sudah ada di `image` bidang, maka tindakan menimpa itu. Anda dapat memasukkan variabel dalam nama gambar.

Contoh:

Jika Anda menentukan`MyDockerImage:${WorkflowSource.CommitId}`, tindakan akan ditambahkan `MyDockerImage:`*`commit-id`* ke file definisi tugas, di mana *`commit-id adalah ID`* komit yang dihasilkan saat runtime oleh alur kerja.

`Jika Anda menentukan``my-ecr-repo/image-repo:`*`$(date +%m-%d-%y-%H-%m-%s), tindakan menambahkan`* `my-ecr-repo/image-repo: date +%m-%d-%y-%h-%m-%s` *`ke file definisi tugas, di mana URI dari Amazon Elastic Container Registry (ECR) dan tanggal +%M-%D-%y-%h-%m-%s`* `my-ecr-repo` *`adalah stempel waktu dalam format yang dihasilkan saat runtime oleh alur kerja.`* `month-day-year-hour-minute-second`

Untuk informasi selengkapnya tentang `image` bidang ini, lihat Gambar di Panduan Pengembang Layanan Kontainer Elastis Amazon. Untuk informasi lebih lanjut tentang variabel, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.

UI yang sesuai: Tab konfigurasi/Nama gambar

environment-variables

(*ECSRenderTaskDefinition*/Configuration/environment-variables)

(Diperlukan)

Tentukan variabel lingkungan yang Anda inginkan tindakan definisi tugas Render Amazon ECS untuk ditambahkan ke file definisi tugas Anda. Tindakan menambahkan variabel ke `environment`

bidang`containerDefinitions`, dalam file definisi tugas Anda. Jika variabel sudah ada dalam file, tindakan menimpa nilai variabel yang ada dan menambahkan variabel baru. Untuk informasi selengkapnya tentang variabel lingkungan Amazon ECS, lihat Menentukan variabel lingkungan di Panduan Pengembang Layanan Amazon Elastic Container.

UI yang sesuai: Tab konfigurasi/Variabel lingkungan - opsional

Outputs

(*`ECSRenderTaskDefinition`*/Outputs)

(Diperlukan)

Mendefinisikan data yang dihasilkan oleh tindakan selama menjalankan alur kerja.

UI yang sesuai: Tab keluaran

Artifacts

(*`ECSRenderTaskDefinition`*/Outputs/Artifacts)

(Diperlukan)

Tentukan artefak yang dihasilkan oleh tindakan. Anda dapat mereferensikan artefak ini sebagai masukan dalam tindakan lain.

Untuk informasi selengkapnya tentang artefak, termasuk contoh, lihatBerbagi data antar tindakan dalam alur kerja menggunakan artefak.

UI yang sesuai: Tab keluaran/Artefak

Name

(*`ECSRenderTaskDefinition`*/Outputs/Artifacts/Name)

(Diperlukan)

Tentukan nama artefak yang akan berisi file definisi tugas yang diperbarui. Nilai default-nya adalah `MyTaskDefinitionArtifact`. Anda kemudian harus menentukan artefak ini sebagai input ke dalam tindakan Deploy to Amazon ECS. Untuk memahami cara menambahkan artefak ini sebagai

masukan ke tindakan Deploy to Amazon ECS, lihat. Contoh alur kerja yang memodifikasi file definisi tugas Amazon ECS

UI yang sesuai: Tab Keluaran/Artefak/Nama

Files

(*ECSRenderTaskDefinition*/Outputs/Artifacts/Files)

(Diperlukan)

Tentukan file yang akan disertakan dalam artefak. Anda harus menentukan `task-definition-*` sehingga file definisi tugas yang diperbarui, yang dimulai dengan`task-definition-`, akan disertakan.

UI yang sesuai: Tab Keluaran/Artefak/File

Variables

(*ECSRenderTaskDefinition*/Outputs/Variables)

(Diperlukan)

Tentukan nama variabel yang akan diatur oleh tindakan render. Tindakan render akan mengatur nilai variabel ini ke nama file definisi tugas yang diperbarui (misalnya,`task-definition-`*`random-string`*`.json`). Anda kemudian harus menentukan variabel ini di properti Definisi tugas (editor visual) atau `task-definition` (editor yaml) tindakan Deploy to Amazon ECS. Untuk memahami cara menambahkan variabel ini ke tindakan Deploy ke Amazon ECS, lihat. Contoh alur kerja yang memodifikasi file definisi tugas Amazon ECS

Default: `task-definition`

UI yang sesuai: Tab Keluaran/Variabel/Bidang nama

Mengkonfigurasi dan menggunakan variabel dalam alur kerja

Variabel adalah pasangan kunci-nilai yang berisi informasi yang dapat Anda referensikan dalam alur kerja Anda CodeCatalyst .

Ada dua jenis variabel yang dapat Anda gunakan dalam alur kerja:

- Variabel yang ditentukan pengguna - Ini adalah pasangan nilai kunci yang Anda tentukan.
- Variabel yang telah ditentukan sebelumnya - Ini adalah pasangan kunci-nilai yang dipancarkan oleh alur kerja secara otomatis. Tidak perlu bagi Anda untuk mendefinisikannya.

> **Note**
>
> CodeCatalyst juga mendukung parameter GitHub output, yang berperilaku seperti variabel dan dapat direferensikan dalam tindakan lain. Untuk informasi selengkapnya, lihat Mengekspor parameter GitHub output sehingga tindakan lain dapat menggunakannya dan Mereferensikan parameter GitHub keluaran

Topik

Menggunakan variabel yang ditentukan pengguna

Variabel yang ditentukan pengguna adalah pasangan nilai kunci yang Anda tentukan. Ada dua jenis:

- Variabel teks biasa, atau hanya variabel - Ini adalah pasangan nilai kunci yang Anda tentukan dalam teks biasa dalam file definisi alur kerja.
- Rahasia — Ini adalah pasangan nilai kunci yang Anda tentukan di halaman Rahasia terpisah di konsol Amazon CodeCatalyst . Kunci (nama) adalah label publik, dan nilainya berisi informasi yang ingin Anda jaga kerahasiaannya. Anda hanya menentukan kunci dalam file definisi alur kerja. Gunakan rahasia sebagai pengganti kata sandi dan informasi sensitif lainnya dalam file definisi alur kerja.

> **Note**
>
> Untuk singkatnya, panduan ini menggunakan istilah variabel untuk berarti variabel plaintext.

Topik

- Mendefinisikan variabel
- Mendefinisikan rahasia
- Mengekspor variabel sehingga tindakan lain dapat menggunakannya
- Merujuk variabel dalam tindakan yang mendefinisikannya
- Mereferensikan output variabel dengan tindakan lain
- Mereferensikan rahasia
- Contoh variabel yang ditentukan pengguna

Mendefinisikan variabel

Anda dapat menentukan variabel dengan dua cara:

- Di `Inputs` bagian tindakan alur kerja - lihat Untuk menentukan variabel di bagian “Input”
- Di `Steps` bagian tindakan alur kerja - lihat Untuk menentukan variabel di bagian “Langkah”

> **Note**
>
> `Steps`Metode ini hanya berfungsi dengan GitHub tindakan CodeCatalyst build, test, dan Actions, karena ini adalah satu-satunya tindakan yang menyertakan `Steps` bagian.

Sebagai contoh, lihat Contoh variabel yang ditentukan pengguna.

Visual

Untuk menentukan variabel di bagian “Input” (editor visual)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Dalam diagram alur kerja, pilih tindakan di mana Anda ingin mengatur variabel.

8. Pilih Input.
9. Dalam Variabel - opsional, pilih Tambahkan variabel, dan kemudian lakukan hal berikut:

   Tentukan urutan pasangan nama/nilai yang menentukan variabel input yang ingin Anda sediakan untuk tindakan. Nama variabel terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama variabel.

   Untuk informasi selengkapnya tentang variabel, termasuk contoh, lihatMengkonfigurasi dan menggunakan variabel dalam alur kerja.
10. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
11. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menentukan variabel di bagian “Input” (editor YAMM)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Dalam tindakan alur kerja, tambahkan kode yang mirip dengan berikut ini:

```
action-name:
  Inputs:
    Variables:
      - Name: variable-name
        Value: variable-value
```

   Untuk contoh lainnya, lihat Contoh variabel yang ditentukan pengguna. Untuk informasi selengkapnya, lihat Alur kerja definisi YAMAL untuk tindakan Anda.
8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Visual

Untuk menentukan variabel di bagian “Langkah” (editor visual)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Dalam diagram alur kerja, pilih tindakan di mana Anda ingin mengatur variabel.
8. Pilih Konfigurasi.
9. Dalam perintah Shell atau GitHubActions YAMAL, mana pun yang tersedia, tentukan variabel dalam tindakan, baik secara eksplisit maupun implisit. `Steps`

   - Untuk mendefinisikan variabel secara eksplisit, sertakan dalam perintah bash langsung ke bagian. `Steps`
   - Untuk mendefinisikan variabel secara implisit, tentukan dalam file yang direferensikan di bagian tindakan. `Steps`

     Sebagai contoh, lihat Contoh variabel yang ditentukan pengguna. Untuk informasi lebih lanjut, lihat Alur kerja definisi YAMAL untuk tindakan.
10. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
11. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk menentukan variabel di bagian “Langkah” (editor YAMM)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.

5. Pilih Edit.
6. Pilih YAMAL.
7. Dalam tindakan alur kerja, tentukan variabel di `Steps` bagian tindakan, baik secara eksplisit maupun implisit.

   - Untuk mendefinisikan variabel secara eksplisit, sertakan dalam perintah bash langsung ke bagian. `Steps`
   - Untuk mendefinisikan variabel secara implisit, tentukan dalam file yang direferensikan di bagian tindakan. `Steps`

     Sebagai contoh, lihat Contoh variabel yang ditentukan pengguna. Untuk informasi lebih lanjut, lihat Alur kerja definisi YAMAL untuk tindakan.
8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Mendefinisikan rahasia

Anda mendefinisikan rahasia di halaman Rahasia CodeCatalyst konsol. Untuk informasi selengkapnya, lihat Mengkonfigurasi dan menggunakan rahasia dalam alur kerja.

Misalnya, Anda mungkin mendefinisikan rahasia yang terlihat seperti ini:

- Nama (kunci): **`my-password`**
- Nilai: **`^*H3#!b9`**

Setelah rahasia didefinisikan, Anda dapat menentukan kunci rahasia (**`my-password`**) dalam file definisi alur kerja. Untuk contoh cara melakukannya, lihat Contoh: Mereferensikan rahasia.

Mengekspor variabel sehingga tindakan lain dapat menggunakannya

Gunakan petunjuk berikut untuk mengekspor variabel dari tindakan sehingga Anda dapat mereferensikannya dalam tindakan lain.

Sebelum Anda mengekspor variabel, perhatikan hal berikut:

- Jika Anda hanya perlu mereferensikan variabel dalam tindakan yang ditentukan, maka Anda tidak perlu mengekspornya.

- Tidak semua tindakan mendukung variabel ekspor. Untuk menentukan apakah tindakan Anda mendukung fitur ini, jalankan melalui instruksi editor visual yang mengikuti, dan lihat apakah tindakan menyertakan tombol Variabel pada tab Output. Jika ya, variabel ekspor didukung.
- Untuk mengekspor variabel dari GitHub Action, lihatMengekspor parameter GitHub output sehingga tindakan lain dapat menggunakannya.

Prasyarat

Pastikan Anda telah menentukan variabel yang ingin Anda ekspor. Untuk informasi selengkapnya, lihat Mendefinisikan variabel.

Visual

Untuk mengekspor variabel (editor visual)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Dalam diagram alur kerja, pilih tindakan yang ingin Anda ekspor variabelnya.
8. Pilih Output.
9. Dalam Variabel - opsional, pilih Tambahkan variabel, lalu lakukan hal berikut:

   Tentukan nama variabel yang ingin Anda ekspor tindakan. Variabel ini harus sudah didefinisikan dalam `Inputs` atau `Steps` bagian dari tindakan yang sama.
10. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
11. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

YAML

Untuk mengekspor variabel (editor YAMM)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.

2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Dalam tindakan yang ingin Anda ekspor variabel dari, tambahkan kode yang mirip dengan berikut ini:

```
action-name:
  Outputs:
    Variables:
      - Name: variable-name
```

   Untuk contoh lainnya, lihat Contoh variabel yang ditentukan pengguna.
8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Merujuk variabel dalam tindakan yang mendefinisikannya

Gunakan instruksi berikut untuk mereferensikan variabel dalam tindakan yang mendefinisikannya.

> **Note**
>
> Untuk mereferensikan variabel yang dihasilkan oleh GitHub Action, lihatMereferensikan parameter GitHub keluaran.

Prasyarat

Pastikan Anda telah menentukan variabel yang ingin Anda referensikan. Untuk informasi selengkapnya, lihat Mendefinisikan variabel.

Visual

Tidak tersedia. Pilih YAMM untuk melihat instruksi YAMM.

YAML

Untuk mereferensikan variabel dalam tindakan yang mendefinisikannya

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Dalam CodeCatalyst tindakan yang mendefinisikan variabel yang ingin Anda rujuk, tambahkan variabel menggunakan sintaks bash berikut:

   ```
   $variable-name
   ```

   Sebagai contoh:

   ```
   MyAction:
       Configuration:
         Steps:
           - Run: $variable-name
   ```

   Untuk contoh lainnya, lihat Contoh variabel yang ditentukan pengguna. Untuk informasi selengkapnya, lihat informasi referensi untuk tindakan Anda diAlur kerja definisi YAMAL.
8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Mereferensikan output variabel dengan tindakan lain

Gunakan instruksi berikut untuk referensi variabel output oleh tindakan lain.

> **Note**
>
> Untuk mereferensikan output variabel dari GitHub Action, lihatMereferensikan parameter GitHub keluaran.

Prasyarat

Pastikan Anda telah mengekspor variabel yang ingin Anda referensikan. Untuk informasi selengkapnya, lihat Mengekspor variabel sehingga tindakan lain dapat menggunakannya.

Visual

Tidak tersedia. Pilih YAMM untuk melihat instruksi YAMM.

YAML

Untuk mereferensikan output variabel dengan tindakan lain (editor YAMAL)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMAL.
7. Dalam CodeCatalyst tindakan, tambahkan referensi ke variabel menggunakan sintaks berikut:

```
${action-group-name.action-name.variable-name}
```

Ganti:

- *action-group-name*dengan nama grup aksi yang berisi tindakan yang menghasilkan variabel.

> **Note**
>
> Anda dapat menghilangkan *action-group-name*jika tidak ada grup tindakan, atau jika variabel dihasilkan oleh tindakan dalam grup tindakan yang sama.

- *action-name* dengan nama action yang mengeluarkan variabel.
- *variabel-nama* dengan nama variabel.

Sebagai contoh:

```
MySecondAction:
    Configuration:
      Steps:
        - Run: ${MyFirstAction.TIMESTAMP}
```

Untuk contoh lainnya, lihat Contoh variabel yang ditentukan pengguna. Untuk informasi selengkapnya, lihat Alur kerja definisi YAMAL untuk tindakan Anda.

8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Mereferensikan rahasia

Untuk petunjuk tentang referensi rahasia dalam file definisi alur kerja, lihat. Menggunakan rahasia

Sebagai contoh, lihat Contoh: Mereferensikan rahasia.

Contoh variabel yang ditentukan pengguna

Contoh berikut menunjukkan bagaimana mendefinisikan dan referensi variabel dalam file definisi alur kerja.

Contoh

- Contoh: Mendefinisikan variabel menggunakan properti Input
- Contoh: Mendefinisikan variabel menggunakan properti Steps
- Contoh: Mengekspor variabel menggunakan properti Output
- Contoh: Mereferensikan variabel yang didefinisikan dalam tindakan yang sama
- Contoh: Mereferensikan variabel yang didefinisikan dalam tindakan lain
- Contoh: Mereferensikan rahasia

Contoh: Mendefinisikan variabel menggunakan properti Input

Contoh berikut menunjukkan cara mendefinisikan dua variabel, `VAR1` dan`VAR2`, di `Inputs` bagian.

```
Actions:
  Build:
    Identifier: aws/build@v1
    Inputs:
      Variables:
      - Name: VAR1
        Value: "My variable 1"
      - Name: VAR2
        Value: "My variable 2"
```

Contoh: Mendefinisikan variabel menggunakan properti Steps

Contoh berikut menunjukkan cara mendefinisikan `DATE` variabel di `Steps` bagian secara eksplisit.

```
Actions:
  Build:
    Identifier: aws/build@v1
    Configuration:
      Steps:
        - Run: DATE=$(date +%m-%d-%y)
```

Contoh: Mengekspor variabel menggunakan properti Output

Contoh berikut menunjukkan kepada Anda bagaimana mendefinisikan dua variabel, `REPOSITORY-URI` dan`TIMESTAMP`, dan mengekspornya menggunakan `Outputs` bagian.

```
Actions:
  Build:
    Identifier: aws/build@v1
    Inputs:
      Variables:
        - Name: REPOSITORY-URI
          Value: 111122223333.dkr.ecr.us-east-2.amazonaws.com/codecatalyst-ecs-image-
repo
    Configuration:
      Steps:
        - Run: TIMESTAMP=$(date +%m-%d-%y-%H-%m-%s)
    Outputs:
      Variables:
        - REPOSITORY-URI
        - TIMESTAMP
```

Contoh: Mereferensikan variabel yang didefinisikan dalam tindakan yang sama

Contoh berikut menunjukkan cara menentukan `VAR1` variabel dalam`MyBuildAction`, dan kemudian referensi dalam tindakan yang sama menggunakan`$VAR1`.

```
Actions:
  MyBuildAction:
    Identifier: aws/build@v1
    Inputs:
      Variables:
        - Name: VAR1
          Value: my-value
    Configuration:
      Steps:
        - Run: $VAR1
```

Contoh: Mereferensikan variabel yang didefinisikan dalam tindakan lain

Contoh berikut menunjukkan cara menentukan `TIMESTAMP` variabel`BuildActionA`, mengekspornya menggunakan `Outputs` properti, dan kemudian mereferensikannya dalam `BuildActionB` menggunakan`${BuildActionA.TIMESTAMP}`.

```
Actions:
  BuildActionA:
    Identifier: aws/build@v1
    Configuration:
      Steps:
        - Run: TIMESTAMP=$(date +%m-%d-%y-%H-%m-%s)
    Outputs:
      Variables:
        - TIMESTAMP
  BuildActionB:
    Identifier: aws/build@v1
    Configuration:
      Steps:
        - Run: docker build -t my-ecr-repo/image-repo:latest .
        - Run: docker tag my-ecr-repo/image-repo:${BuildActionA.TIMESTAMP}

        # Specifying just '$TIMESTAMP' here will not work
        # because TIMESTAMP is not a variable
        # in the BuildActionB action.
```

Contoh: Mereferensikan rahasia

Contoh berikut menunjukkan kepada Anda cara mereferensikan `my-password` rahasia. `my-password`Itu adalah kunci rahasianya. Kunci rahasia ini dan nilai kata sandi yang sesuai harus ditentukan pada halaman Rahasia CodeCatalyst konsol sebelum digunakan dalam file definisi alur kerja. Untuk informasi selengkapnya, lihat Mengkonfigurasi dan menggunakan rahasia dalam alur kerja.

```
Actions:
  BuildActionA:
    Identifier: aws/build@v1
    Configuration:
      Steps:
        - Run: curl -u LiJuan:${Secrets.my-password} https://example.com
```

Menggunakan variabel yang telah ditentukan

Variabel yang telah ditentukan adalah pasangan nilai kunci yang dipancarkan oleh alur kerja secara otomatis, dan tersedia untuk Anda gunakan dalam tindakan alur kerja.

Anda dapat menggunakan variabel yang telah ditentukan dalam tindakan alur kerja apa pun.

Topik

- Mereferensikan variabel yang telah ditentukan
- Menentukan variabel standar mana yang dipancarkan alur kerja Anda
- Contoh variabel yang telah ditentukan

Mereferensikan variabel yang telah ditentukan

Gunakan petunjuk berikut untuk referensi variabel yang telah ditentukan.

Prasyarat

Tentukan nama variabel yang telah ditentukan yang ingin Anda referensikan, sepertiCommitId. Untuk informasi selengkapnya, lihat Menentukan variabel standar mana yang dipancarkan alur kerja Anda.

Visual

Tidak tersedia. Pilih YAMG untuk melihat instruksi YAMG.

YAML

Untuk mereferensikan variabel yang telah ditentukan (editor YAMG)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMG.
7. Dalam CodeCatalyst tindakan, tambahkan referensi variabel yang telah ditentukan menggunakan sintaks berikut:

```
${action-group-name.action-name-or-WorkflowSource.variable-name}
```

Ganti:

- *action-group-name* dengan nama grup aksi.

  > **Note**
  >
  > Anda dapat menghilangkan *action-group-name* jika tidak ada grup tindakan, atau jika variabel dihasilkan oleh tindakan dalam grup tindakan yang sama.

- *action-name-or- WorkflowSource* dengan:

  Nama tindakan yang mengeluarkan variabel.

  atau

  `WorkflowSource`, jika variabelnya adalah `BranchName` atau `CommitId` variabel.

- *variabel-nama* dengan nama variabel.

Sebagai contoh:

```
MySecondAction:
    Configuration:
```

```
      Steps:
        - Run: echo ${MyFirstECSAction.cluster}
```

Contoh lain:

```
MySecondAction:
    Configuration:
      Steps:
        - Run: echo ${WorkflowSource.CommitId}
```

Untuk contoh lainnya, lihat Contoh variabel yang telah ditentukan. Untuk informasi selengkapnya, lihat Alur kerja definisi YAMAL untuk tindakan Anda.

8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Menentukan variabel standar mana yang dipancarkan alur kerja Anda

Anda dapat menentukan variabel standar mana yang dipancarkan alur kerja Anda dengan dua cara:

- Jalankan alur kerja sekali. Setelah proses selesai, variabel yang dipancarkan oleh alur kerja ditampilkan pada tab Variabel pada halaman rincian jalankan. Untuk informasi selengkapnya, lihat Melihat alur kerja menjalankan status dan detail.
- Konsultasikan Daftar variabel yang telah ditentukan. Referensi ini mencantumkan nama variabel (kunci) dan nilai untuk setiap variabel yang telah ditentukan.

> **Note**
>
> Ukuran total maksimum variabel alur kerja tercantum dalamKuota untuk alur kerja. Jika ukuran total melebihi maksimum, tindakan yang terjadi setelah maksimum tercapai mungkin gagal.

Contoh variabel yang telah ditentukan

Contoh berikut menunjukkan cara mereferensikan variabel yang telah ditentukan dalam file definisi alur kerja.

Contoh

- Contoh: Mereferensikan variabel CommitId standar ""
- Contoh: Mereferensikan variabel BranchName standar ""

Contoh: Mereferensikan variabel CommitId standar ""

Contoh berikut menunjukkan kepada Anda bagaimana merujuk ke variabel yang `CommitId` telah ditentukan dalam `MyBuildAction` tindakan. `CommitId`Variabel adalah output secara otomatis oleh CodeCatalyst.

Meskipun contoh menunjukkan variabel yang digunakan dalam aksi build, Anda dapat menggunakannya `CommitId` dalam tindakan apa pun.

```
MyBuildAction:
    Identifier: aws/build@v1
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      Steps:
      #Build Docker image and tag it with a commit ID
        - Run: docker build -t image-repo/my-docker-image:latest .
        - Run: docker tag image-repo/my-docker-image:${WorkflowSource.CommitId}
```

Contoh: Mereferensikan variabel BranchName standar ""

Contoh berikut menunjukkan kepada Anda bagaimana merujuk ke variabel yang `BranchName` telah ditentukan dalam `CDKDeploy` tindakan. `BranchName`Variabel adalah output secara otomatis oleh CodeCatalyst.

Meskipun contoh menunjukkan variabel yang digunakan dalam tindakan AWS CDK penerapan, Anda dapat menggunakan `BranchName` dalam tindakan apa pun.

```
CDKDeploy:
    Identifier: aws/cdk-deploy@v1
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      StackName: app-stack-${WorkflowSource.BranchName}
```

Daftar variabel yang telah ditentukan

Konsultasikan bagian berikut untuk melihat variabel yang telah ditentukan yang dihasilkan secara otomatis oleh CodeCatalyst tindakan.

> **Note**
>
> Daftar ini hanya mencakup variabel standar yang dipancarkan oleh CodeCatalyst sumber dan tindakan. CodeCatalyst Jika Anda menggunakan jenis tindakan lain, seperti Tindakan atau GitHub tindakan CodeCatalyst Labs, lihat sebagai gantinyaMenentukan variabel standar mana yang dipancarkan alur kerja Anda.

Daftar

> **Note**
>
> Tidak semua CodeCatalyst tindakan menghasilkan variabel yang telah ditentukan. Jika tindakan tidak ada dalam daftar, maka itu tidak menghasilkan variabel.

- Variabel yang dihasilkan oleh sumber (BranchName“” dan CommidId “”)
- Variabel yang dihasilkan oleh tindakan “Deploy AWS CloudFormation stack”
- Variabel yang dihasilkan oleh tindakan “Terapkan ke Amazon ECS”
- Variabel yang dihasilkan oleh aksi “Deploy to Kubernetes cluster”
- Variabel yang dihasilkan oleh tindakan "AWS CDK menyebarkan”
- Variabel yang dihasilkan oleh tindakan "AWS CDK bootstrap”
- Variabel yang dihasilkan oleh tindakan "AWS Lambda memanggil”
- Variabel yang dihasilkan oleh tindakan “Render Amazon ECS task definition”

Mengkonfigurasi dan menggunakan rahasia dalam alur kerja

Mungkin ada saat-saat ketika Anda perlu menggunakan data sensitif, seperti kredensi otentikasi, dalam alur kerja Anda. Menyimpan nilai-nilai ini dalam plaintext di mana saja di repositori Anda harus dihindari karena siapa pun yang memiliki akses ke repositori yang berisi rahasia dapat melihatnya. Demikian pula, nilai-nilai ini tidak boleh digunakan secara langsung dalam definisi alur kerja apa pun

karena akan terlihat sebagai file di repositori Anda. Dengan CodeCatalyst, Anda dapat melindungi nilai-nilai ini dengan menambahkan rahasia ke proyek Anda, dan kemudian mereferensikan rahasia dalam file definisi alur kerja Anda. Perhatikan bahwa Anda dapat memiliki maksimal lima rahasia per tindakan.

> **Note**
>
> Rahasia hanya dapat digunakan untuk mengganti kata sandi dan informasi sensitif dalam file definisi alur kerja.

Topik

Membuat rahasia

Gunakan prosedur berikut untuk membuat rahasia. Rahasianya berisi informasi sensitif yang ingin Anda sembunyikan dari pandangan.

> **Note**
>
> Rahasia terlihat oleh tindakan dan tidak disamarkan saat ditulis ke file.

Untuk membuat rahasia

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih CI/CD, lalu pilih Rahasia.
3. Pilih Buat rahasia.
4. Masukkan informasi berikut:

   Nama

   Masukkan nama untuk rahasia Anda.

Nilai

Masukkan nilai untuk rahasianya. Ini adalah informasi sensitif yang ingin Anda sembunyikan dari pandangan. Secara default, nilainya tidak ditampilkan. Untuk menampilkan nilai, pilih Tampilkan nilai.

Deskripsi

(Opsional) Masukkan deskripsi untuk rahasia Anda.

5. Pilih Buat.

Mengedit rahasia

Gunakan prosedur berikut untuk mengedit rahasia.

Untuk mengedit rahasia

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih CI/CD, lalu pilih Rahasia.
3. Dalam daftar rahasia, pilih rahasia yang ingin Anda edit.
4. Pilih Edit.
5. Edit properti berikut:

   Nilai

   Masukkan nilai untuk rahasianya. Ini adalah nilai yang ingin Anda sembunyikan dari tampilan. Secara default, nilainya tidak ditampilkan.

   Deskripsi

   (Opsional) Masukkan deskripsi untuk rahasia Anda.

6. Pilih Simpan.

Menggunakan rahasia

Untuk menggunakan rahasia dalam tindakan alur kerja, Anda harus mendapatkan pengenal referensi rahasia dan menggunakan pengenal itu dalam tindakan alur kerja.

Topik

- Memperoleh pengenal rahasia
- Mereferensikan rahasia dalam alur kerja

Memperoleh pengenal rahasia

Gunakan prosedur berikut untuk mendapatkan pengenal referensi rahasia. Anda akan menambahkan pengenal ini ke alur kerja Anda.

Untuk mendapatkan pengenal referensi rahasia

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih CI/CD, lalu pilih Rahasia.
3. Dalam daftar rahasia, temukan rahasia yang ingin Anda gunakan.
4. Di kolom ID Referensi, salin pengenal rahasia. Berikut ini adalah sintaks untuk ID Referensi:

```
${Secrets.<name>}
```

Mereferensikan rahasia dalam alur kerja

Gunakan prosedur berikut untuk mereferensikan rahasia dalam alur kerja.

Untuk mereferensikan rahasia

1. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
2. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
3. Pilih Edit.
4. Pilih YAMAL.
5. Ubah YAMAL untuk menggunakan pengenal rahasia. Misalnya, untuk menggunakan nama pengguna dan kata sandi yang disimpan sebagai rahasia dengan `curl` perintah, Anda akan menggunakan `Run` perintah yang mirip dengan berikut ini:

```
- Run: curl -u <username-secret-identifier>:<password-secret-identifier> https://
example.com
```

6. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.

7. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Menghapus rahasia

Gunakan prosedur berikut untuk menghapus rahasia dan pengenal referensi rahasia.

> **Note**
>
> Sebelum menghapus rahasia, kami sarankan Anda menghapus pengenal referensi rahasia dari semua tindakan alur kerja. Jika Anda menghapus rahasia tanpa menghapus pengenal referensi, tindakan akan gagal saat berikutnya dijalankan.

Untuk menghapus pengenal referensi rahasia dari alur kerja

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
3. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
4. Pilih Edit.
5. Pilih YAMAL.
6. Cari alur kerja untuk string berikut:

   ```
   ${Secrets.
   ```

   Ini menemukan semua pengidentifikasi referensi dari semua rahasia.
7. Hapus pengenal referensi dari rahasia yang dipilih, atau ganti dengan nilai teks biasa.
8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMAL alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.

Untuk menghapus rahasia

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih CI/CD, lalu pilih Rahasia.
3. Dalam daftar rahasia, pilih rahasia yang ingin Anda hapus.

4. Pilih Hapus.
5. Masukkan **`delete`** untuk mengonfirmasi penghapusan.
6. Pilih Hapus.

Melihat status alur kerja

Anda mungkin ingin melihat status alur kerja untuk melihat apakah ada masalah konfigurasi alur kerja yang perlu Anda atasi, atau untuk memecahkan masalah yang gagal untuk memulai. CodeCatalystmengevaluasi status alur kerja setiap kali Anda membuat atau memperbarui file definisi alur kerja yang mendasari alur kerja.

> **Note**
>
> Anda juga dapat melihat status run alur kerja, yang berbeda dari status alur kerja. Untuk informasi selengkapnya, lihat Melihat alur kerja menjalankan status dan detail.

Untuk daftar kemungkinan status alur kerja, lihatStatus alur kerja.

Untuk melihat status alur kerja

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Temukan alur kerja yang statusnya ingin Anda lihat. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.

   Status ditampilkan dengan alur kerja dalam daftar.
5. (Opsional) Pilih nama alur kerja, dan temukan bidang Definisi alur kerja. Ini menunjukkan status alur kerja.

Kuota untuk alur kerja

Tabel berikut menjelaskan kuota dan batas untuk alur kerja di Amazon. CodeCatalyst

Untuk informasi selengkapnya tentang kuota di Amazon CodeCatalyst, lihatKuota untuk CodeCatalyst.

Jumlah maksimum alur kerja per ruang	800
Ukuran file definisi alur kerja maksimum	256 KB
Jumlah maksimum file alur kerja yang diproses dalam satu peristiwa sumber	50
Jumlah maksimum file yang diproses dalam satu peristiwa sumber	4.000
Jumlah maksimum armada aktif per ruang	10
Jumlah maksimum instans komputasi aktif per armada	20
Jumlah maksimum artefak masukan per tindakan	10
Jumlah maksimum artefak keluaran per tindakan	10
Ukuran total maksimum dari variabel keluaran tindakan tunggal	120 KB
Panjang maksimum nilai variabel keluaran	500 karakter atau lebih, tergantung pada tindakan yang memancarkan nilai. **Note** Nilai dapat terpotong jika melebihi batas tindakan.
Jumlah hari maksimum untuk menyimpan artefak yang dihasilkan selama alur kerja berjalan	30

Jumlah maksimum laporan per tindakan	50
Jumlah maksimum kasus uji per laporan pengujian	20.000
Jumlah maksimum file per laporan cakupan kode	20.000
Jumlah maksimum temuan analisis komposisi perangkat lunak per laporan	20.000
Jumlah maksimum file per laporan analisis statis	20.000
Jumlah maksimum alur kerja bersamaan berjalan per ruang	100
Jumlah maksimum tindakan per alur kerja	50
Jumlah maksimum tindakan yang berjalan secara bersamaan per alur kerja	50
Jumlah maksimum tindakan yang berjalan secara bersamaan per spasi	200
Jumlah maksimum waktu suatu tindakan dapat dijalankan	Untuk tindakan build dan test, batas waktu adalah 8 jam. Untuk semua tindakan lainnya, batas waktu adalah 1 jam.
Jumlah maksimum lingkungan yang terkait dengan Akun AWS per ruang	5.000
Jumlah maksimum rahasia per tindakan	5
Jumlah maksimum rahasia per ruang	500.000

Alur kerja menjalankan status

Jalankan alur kerja dapat berada di salah satu status berikut:

- Berhasil - Alur kerja berjalan berhasil diproses.
- Gagal - Satu atau beberapa tindakan dalam alur kerja berjalan gagal.
- Sedang berlangsung - Proses alur kerja saat ini sedang diproses.
- Berhenti — Seseorang menghentikan alur kerja saat sedang berlangsung.
- Berhenti — Alur kerja berjalan saat ini sedang dihentikan.
- Dibatalkan - Proses alur kerja dibatalkan CodeCatalyst karena alur kerja terkait telah dihapus atau diperbarui saat proses sedang berlangsung.
- Digantikan - Hanya terjadi jika Anda telah mengonfigurasi mode lari yang digantikan. Proses alur kerja dibatalkan oleh CodeCatalyst karena alur kerja selanjutnya menggantikannya.

Status alur kerja

Alur kerja dapat memiliki salah satu status berikut:

- Valid - Alur kerja dapat dijalankan dan dapat diaktifkan oleh pemicu.

  Agar alur kerja ditandai sebagai valid, kedua kondisi berikut harus benar:
  - File definisi alur kerja harus valid.
  - Alur kerja harus tidak memiliki pemicu, tidak ada pemicu push, atau pemicu push yang berjalan menggunakan file di cabang saat ini. Untuk informasi selengkapnya, lihat Memicu pertimbangan saat bercabang.
- Tidak valid - File definisi alur kerja tidak valid. Alur kerja tidak dapat dijalankan secara manual, atau secara otomatis melalui pemicu. Alur kerja yang tidak valid muncul dengan definisi Alur Kerja memiliki ***n*** pesan kesalahan (atau serupa) di CodeCatalyst konsol.

  Agar alur kerja ditandai sebagai tidak valid, kondisi berikut harus benar:
  - File definisi alur kerja harus salah dikonfigurasi.

    Untuk memperbaiki file definisi alur kerja yang salah konfigurasi, lihat. Bagaimana cara memperbaiki kesalahan “Definisi alur kerja memiliki *n* kesalahan”?
- Tidak aktif — Definisi alur kerja valid tetapi tidak dapat dijalankan secara manual, atau secara otomatis melalui pemicu.

Agar alur kerja ditandai sebagai tidak aktif, kedua kondisi berikut harus benar:

- File definisi alur kerja harus valid.
- File definisi alur kerja harus menyertakan pemicu push yang menentukan cabang yang berbeda dari cabang tempat file definisi alur kerja saat ini aktif. Untuk informasi selengkapnya, lihat Memicu pertimbangan saat bercabang.

  Untuk mengalihkan alur kerja dari Tidak Aktif ke Aktif, lihat. Bagaimana cara memperbaiki pesan "Alur kerja tidak aktif"?

> **Note**
>
> Jika alur kerja menentukan sumber daya yang kemudian Anda hapus (misalnya, repositori paket), tidak CodeCatalyst akan mendeteksi perubahan ini dan akan terus menandai alur kerja sebagai valid. Jenis masalah ini akan tertangkap saat alur kerja berjalan.

Alur kerja definisi YAMAL

Berikut ini adalah dokumentasi referensi untuk file definisi alur kerja.

File definisi alur kerja adalah file YAMAL yang menjelaskan alur kerja Anda. File disimpan dalam `~/.codecatalyst/workflows/` folder di root repositori sumber Anda. File dapat memiliki ekstensi.yml.yaml.

Untuk membuat dan mengedit file definisi alur kerja, Anda dapat menggunakan editor seperti vim, atau Anda dapat menggunakan editor visual CodeCatalyst konsol atau editor YAMM. Untuk informasi selengkapnya, lihat Menggunakan editor visual dan YAMAL CodeCatalyst konsol.

> **Note**
>
> Sebagian besar properti YAMAL yang mengikuti memiliki elemen UI yang sesuai di editor visual. Untuk mencari elemen UI, gunakan Ctrl+F. Elemen akan terdaftar dengan properti YAMLnya yang terkait.

Topik

- Contoh file definisi alur kerja

- Pedoman dan konvensi sintaks
- Properti tingkat atas

Contoh file definisi alur kerja

Berikut ini adalah contoh file definisi alur kerja sederhana. Ini mencakup beberapa properti tingkat atas, `Triggers` bagian, dan `Actions` bagian dengan dua tindakan: `Build` dan`Test`. Untuk informasi selengkapnya, lihat Tentang file definisi alur kerja.

```
Name: MyWorkflow
SchemaVersion: 1.0
RunMode: QUEUED
Triggers:
  - Type: PUSH
    Branches:
      - main
Actions:
  Build:
    Identifier: aws/build@v1
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      Steps:
        - Run: docker build -t MyApp:latest .
  Test:
    Identifier: aws/managed-test@v1
    DependsOn:
      - Build
    Inputs:
      Sources:
        - WorkflowSource
    Configuration:
      Steps:
        - Run: npm install
        - Run: npm run test
```

Pedoman dan konvensi sintaks

Bagian ini menjelaskan aturan sintaks untuk file definisi alur kerja, serta konvensi penamaan yang digunakan dalam dokumentasi referensi ini.

Pedoman sintaks YAMAL

File definisi alur kerja ditulis dalam YAMG dan mengikuti spesifikasi YAMM 1.1, jadi apa pun yang diizinkan dalam spesifikasi itu juga diperbolehkan dalam alur kerja YAMM. Jika Anda baru mengenal YAMAL, berikut adalah beberapa panduan cepat untuk memastikan Anda menyediakan kode YAMAL yang valid.

- Case-sensitivity: File definisi alur kerja peka huruf besar/kecil, jadi pastikan Anda menggunakan casing yang ditunjukkan dalam dokumentasi ini.
- Karakter khusus: Sebaiknya gunakan tanda kutip atau tanda kutip ganda di sekitar nilai properti yang menyertakan salah satu karakter khusus berikut: `{ } [],,,*,#,?,|,-,,,,,,=,!,%,@,:, ` ,`

  Jika Anda tidak menyertakan tanda kutip, karakter khusus yang tercantum sebelumnya dapat ditafsirkan dengan cara yang tidak terduga.
- Nama properti: Nama properti (sebagai lawan dari nilai properti) terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip atau tanda kutip ganda untuk mengaktifkan karakter dan spasi khusus dalam nama properti.

  Tidak diizinkan:

  ```
  'My#Build@action'
  ```

  ```
  My#Build@action
  ```

  ```
  My Build Action
  ```

  Diizinkan:

  ```
  My-Build-Action_1
  ```
- Kode Escape: Jika nilai properti Anda menyertakan kode escape (misalnya, `\n` atau`\t`), ikuti panduan berikut:
  - Gunakan tanda kutip tunggal untuk mengembalikan kode escape sebagai string. Misalnya`'my string \n my string'`, mengembalikan string`my string \n my string`.
  - Gunakan tanda kutip ganda untuk mengurai kode escape. Misalnya,`"my string \n my new line"`, mengembalikan:

    ```
    my string
    ```

```
my new line
```

- Komentar: Kata pengantar komentar dengan#.

  Contoh:

```
Name: MyWorkflow
# This is a comment.
SchemaVersion: 1.0
```

- Triple dash (`---`): Jangan gunakan `---` dalam kode YAMAL Anda. CodeCatalyst mengabaikan segalanya setelah. `---`

Konvensi penamaan

Dalam panduan ini, kami menggunakan properti dan bagian istilah untuk merujuk ke item utama dalam file definisi alur kerja.

- Properti adalah item apa pun yang menyertakan titik dua (`:`). Misalnya, dalam cuplikan kode berikut, semua yang berikut adalah properti:`Name`,,`SchemaVersion`, `RunMode` `TriggersType`, dan. `Branches`
- Bagian adalah properti apa pun yang memiliki sub-properti. Dalam cuplikan kode berikut, ada satu `Triggers` bagian.

> **Note**
>
> Dalam panduan ini, 'bagian' kadang-kadang disebut sebagai 'properti', dan sebaliknya, tergantung pada konteksnya.

```
Name: MyWorkflow
SchemaVersion: 1.0
RunMode: QUEUED
Triggers:
  - Type: PUSH
    Branches:
      - main
```

Properti tingkat atas

Berikut ini adalah dokumentasi referensi untuk properti tingkat atas dalam file definisi alur kerja.

```
# Name
Name: workflow-name

# Schema version
SchemaVersion: 1.0

# Run mode
RunMode: QUEUED|SUPERSEDED|PARALLEL

# Compute
Compute:
...

# Triggers
Triggers:
...

# Actions
Actions:
...
```

Name

(Diperlukan)

Nama alur kerja. Nama alur kerja ditampilkan dalam daftar alur kerja dan disebutkan dalam notifikasi dan log. Nama alur kerja dan nama file definisi alur kerja dapat cocok, atau Anda dapat menamainya secara berbeda. Nama alur kerja tidak harus unik. Nama alur kerja terbatas pada karakter alfanumerik (a-z, A-Z, 0-9), tanda hubung (-), dan garis bawah (_). Spasi tidak diizinkan. Anda tidak dapat menggunakan tanda kutip untuk mengaktifkan karakter dan spasi khusus dalam nama alur kerja.

UI yang sesuai: editor visual/properti alur kerja/nama alur kerja

SchemaVersion

(Diperlukan)

Versi skema definisi alur kerja. Saat ini, satu-satunya nilai yang valid adalah`1.0`.

UI yang sesuai: tidak ada

RunMode

(Opsional)

Bagaimana CodeCatalyst menangani beberapa proses. Anda dapat menggunakan salah satu nilai berikut:

- `QUEUED`— Beberapa run diantrian dan dijalankan satu demi satu. Anda dapat memiliki hingga 50 run dalam antrian.
- `SUPERSEDED`— Beberapa run diantrian dan dijalankan satu demi satu. Antrian hanya dapat memiliki satu run, jadi jika dua run berakhir bersama dalam antrian yang sama, run selanjutnya menggantikan (mengambil alih dari) run sebelumnya, dan run sebelumnya dibatalkan.
- `PARALLEL`— Beberapa run terjadi secara bersamaan.

Jika properti ini dihilangkan, defaultnya adalah. `QUEUED`

Untuk informasi selengkapnya, lihat Mengonfigurasi perilaku antrian run.

UI yang sesuai: editor visual/properti alur kerja/Advanced/Run mode

Compute

(Opsional)

Mesin komputasi yang digunakan untuk menjalankan tindakan alur kerja Anda. Anda dapat menentukan komputasi baik di tingkat alur kerja atau di tingkat tindakan, tetapi tidak keduanya. Ketika ditentukan pada tingkat alur kerja, konfigurasi komputasi berlaku untuk semua tindakan yang ditentukan dalam alur kerja. Pada tingkat alur kerja, Anda juga dapat menjalankan beberapa tindakan pada instance yang sama. Untuk informasi selengkapnya, lihat Berbagi komputasi di seluruh tindakan.

Untuk informasi selengkapnya tentang komputasi, lihatMengkonfigurasi gambar Docker lingkungan komputasi dan runtime untuk alur kerja.

UI yang sesuai: tidak ada

```
Name: MyWorkflow
```

```
SchemaVersion: 1.0
...
Compute:
  Type: EC2 | Lambda
  Fleet: fleet-name
  SharedInstance: true | false
```

Type

(Compute/Type)

(Diperlukan `Compute` jika diatur)

Jenis mesin komputasi. Anda dapat menggunakan salah satu nilai berikut:

- EC2 (editor visual) atau `EC2` (editor YAMG)

  Dioptimalkan untuk fleksibilitas selama aksi berjalan.
- Lambda (editor visual) atau `Lambda` (editor YAMG)

  Kecepatan start-up aksi yang dioptimalkan.

Untuk informasi selengkapnya tentang jenis komputasi, lihatJenis komputasi.

UI yang sesuai: editor visual/properti alur kerja/Lanjutan/Jenis komputasi

Fleet

(Compute/Fleet)

(Opsional)

Tentukan mesin atau armada yang akan menjalankan alur kerja atau tindakan alur kerja Anda. Dengan armada sesuai permintaan, ketika suatu tindakan dimulai, alur kerja menyediakan sumber daya yang dibutuhkan, dan mesin dihancurkan ketika tindakan selesai. Contoh armada sesuai permintaan:`Linux.x86-64.Large`,. `Linux.x86-64.XLarge` Untuk informasi lebih lanjut tentang armada sesuai permintaan, lihat. Properti armada sesuai permintaan

Dengan armada yang disediakan, Anda mengonfigurasi satu set mesin khusus untuk menjalankan tindakan alur kerja Anda. Mesin-mesin ini tetap menganggur, siap untuk memproses tindakan segera. Untuk informasi lebih lanjut tentang armada yang disediakan, lihat. Properti armada yang disediakan

Jika `Fleet` dihilangkan, defaultnya adalah. `Linux.x86-64.Large`

Untuk informasi selengkapnya tentang armada komputasi, lihat. Hitung armada

UI yang sesuai: editor visual/properti alur kerja/Advanced/Compute fleet

SharedInstance

(Compute/SharedInstance)

(Opsional)

Tentukan kemampuan berbagi komputasi untuk tindakan Anda. Dengan berbagi komputasi, tindakan dalam alur kerja berjalan pada instance yang sama (gambar lingkungan runtime). Anda dapat menggunakan salah satu nilai berikut:

- `TRUE`berarti bahwa citra lingkungan runtime dibagikan di antara tindakan alur kerja.
- `FALSE`berarti bahwa image lingkungan runtime terpisah dimulai dan digunakan untuk setiap tindakan dalam alur kerja, sehingga Anda tidak dapat berbagi sumber daya seperti artefak dan variabel tanpa konfigurasi tambahan.

Untuk informasi selengkapnya tentang berbagi komputasi, lihatBerbagi komputasi di seluruh tindakan.

UI yang sesuai: tidak ada

Triggers

(Opsional)

Urutan satu atau lebih pemicu untuk alur kerja ini. Jika pemicu tidak ditentukan, maka Anda harus memulai alur kerja Anda secara manual.

Untuk informasi lebih lanjut tentang menggunakan pemicu, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.

UI yang sesuai: editor visual/diagram alur kerja/Pemicu

```
Name: MyWorkflow
SchemaVersion: 1.0
...
```

```
Triggers:
  - Type: PUSH
    Branches:
      - branch-name
    FilesChanged:
      - folder1/file
      - folder2/

  - Type: PULLREQUEST
    Events:
      - OPEN
      - CLOSED
      - REVISION
    Branches:
      - branch-name
    FilesChanged:
      - file1.txt

  - Type: SCHEDULE
    # Run the workflow at 10:15 am (UTC+0) every Saturday
    Expression: "15 10 ? * 7 *"
    Branches:
      - branch-name
```

Type

(Triggers/Type)

(Diperlukan `Triggers` jika diatur)

Tentukan jenis pemicu. Anda dapat menggunakan salah satu nilai berikut:

- Dorong (editor visual) atau `PUSH` (editor YAMG)

  Pemicu push memulai alur kerja saat perubahan didorong ke repositori sumber Anda. Workflow run akan menggunakan file di cabang yang Anda dorong (yaitu, cabang tujuan).
- Permintaan tarik (editor visual) atau `PULLREQUEST` (editor YAMG)

  Pemicu permintaan tarik memulai alur kerja saat permintaan tarik dibuka, diperbarui, atau ditutup di repositori sumber Anda. Workflow run akan menggunakan file di cabang yang Anda tarik dari (yaitu, cabang sumber).
- Jadwal (editor visual) atau `SCHEDULE` (editor YAMG)

Pemicu jadwal memulai alur kerja berjalan pada jadwal yang ditentukan oleh ekspresi cron yang Anda tentukan. Jalankan alur kerja terpisah akan dimulai untuk setiap cabang di repositori sumber Anda menggunakan file cabang. (Untuk membatasi cabang tempat pemicu diaktifkan, gunakan bidang Cabang (editor visual) atau `Branches` properti (editor YAMM).)

Saat mengonfigurasi pemicu jadwal, ikuti panduan ini:

- Hanya gunakan satu pemicu jadwal per alur kerja.
- Jika Anda telah menentukan beberapa alur kerja di CodeCatalyst ruang Anda, sebaiknya Anda menjadwalkan tidak lebih dari 10 alur kerja untuk memulai secara bersamaan.
- Pastikan Anda mengonfigurasi ekspresi cron pemicu dengan waktu yang cukup di antara proses. Untuk informasi selengkapnya, lihat Expression.

Sebagai contoh, silakan lihat Contoh pemicu.

UI yang sesuai: editor visual/diagram alur kerja/pemicu/tipe pemicu

Events

(Triggers/Events)

(Diperlukan jika pemicu `Type` diatur ke`PULLREQUEST`)

Tentukan jenis peristiwa permintaan tarik yang akan memulai alur kerja. Berikut ini adalah nilai yang valid:

- Permintaan tarik dibuat (editor visual) atau `OPEN` (editor YAMAL)

  Proses alur kerja dimulai saat permintaan tarik dibuat.
- Permintaan tarik ditutup (editor visual) atau `CLOSED` (editor YAMAL)

  Proses alur kerja dimulai saat permintaan tarik ditutup. Perilaku `CLOSED` acara itu rumit, dan paling baik dipahami melalui sebuah contoh. Untuk informasi selengkapnya, lihat Contoh: Pemicu dengan tarikan, cabang, dan acara 'CLOSED'.
- Revisi baru dibuat untuk menarik permintaan (editor visual) atau `REVISION` (editor YAMAL)

  Alur kerja dijalankan ketika revisi ke permintaan tarik dibuat. Revisi pertama dibuat saat permintaan tarik dibuat. Setelah itu, revisi baru dibuat setiap kali seseorang mendorong komit baru ke cabang sumber yang ditentukan dalam permintaan tarik. Jika Anda menyertakan `REVISION` peristiwa

dalam pemicu permintaan tarik Anda, Anda dapat menghilangkan `OPEN` acara tersebut, karena `REVISION` merupakan superset dari. `OPEN`

Anda dapat menentukan beberapa peristiwa dalam pemicu permintaan tarik yang sama.

Sebagai contoh, lihat Contoh pemicu.

UI yang sesuai: editor visual/diagram alur kerja/pemicu/Acara untuk permintaan tarik

Branches

(Triggers/Branches)

(Opsional)

Tentukan cabang di repositori sumber Anda yang dipantau pemicu untuk mengetahui kapan harus memulai alur kerja. Anda dapat menggunakan pola regex untuk menentukan nama cabang Anda. Misalnya, gunakan `main.*` untuk mencocokkan semua cabang yang dimulai dengan`main`.

Cabang yang akan ditentukan berbeda tergantung pada jenis pemicu:

- Untuk pemicu dorong, tentukan cabang yang Anda dorong, yaitu cabang tujuan. Satu alur kerja akan dimulai per cabang yang cocok, menggunakan file di cabang yang cocok.

  Contoh:`main.*`, `mainline`
- Untuk pemicu permintaan tarik, tentukan cabang yang Anda dorong, yaitu cabang tujuan. Satu alur kerja akan dimulai per cabang yang cocok, menggunakan file definisi alur kerja dan file sumber di cabang sumber (bukan cabang yang cocok).

  Contoh:`main.*`,`mainline`, `v1\-.*` (cocok dengan cabang yang dimulai dengan`v1-`)
- Untuk pemicu jadwal, tentukan cabang yang berisi file yang ingin Anda jalankan terjadwal untuk digunakan. Satu alur kerja akan dimulai per cabang yang cocok, menggunakan file definisi alur kerja dan file sumber di cabang yang cocok.

  Contoh:`main.*`, `version\-1\.0`

> **Note**
>
> Jika Anda tidak menentukan cabang, pemicu memantau semua cabang di repositori sumber Anda, dan akan memulai alur kerja menggunakan file definisi alur kerja dan file sumber di:

- Cabang yang Anda dorong (untuk pemicu push). Untuk informasi selengkapnya, lihat Contoh: Pemicu push kode sederhana.
- Cabang yang Anda tarik (untuk pemicu permintaan tarik). Untuk informasi selengkapnya, lihat Contoh: Pemicu permintaan tarik sederhana.
- Semua cabang (untuk pemicu jadwal). Satu alur kerja akan dimulai per cabang di repositori sumber Anda. Untuk informasi selengkapnya, lihat Contoh: Pemicu jadwal sederhana.

Untuk informasi lebih lanjut tentang cabang dan pemicu, lihatMemicu pertimbangan saat bercabang.

Untuk contoh lainnya, lihat Contoh pemicu.

UI yang sesuai: editor visual/diagram alur kerja/pemicu/Cabang

FilesChanged

(Triggers/FilesChanged)

(Opsional jika pemicu `Type` diatur ke`PUSH`, atau`PULLREQUEST`. Tidak didukung jika pemicu `Type` disetel ke`SCHEDULE`.)

Tentukan file atau folder di repositori sumber Anda yang dipantau pemicu untuk mengetahui kapan harus memulai alur kerja. Anda dapat menggunakan ekspresi reguler untuk mencocokkan nama file atau jalur.

Sebagai contoh, lihat Contoh pemicu.

UI yang sesuai: editor visual/diagram alur kerja/pemicu/File diubah

Expression

(Triggers/Expression)

(Diperlukan jika pemicu `Type` diatur ke`SCHEDULE`)

Tentukan ekspresi cron yang menjelaskan kapan Anda ingin alur kerja terjadwal Anda berjalan terjadi.

Ekspresi cron CodeCatalyst menggunakan sintaks enam bidang berikut, di mana setiap bidang dipisahkan oleh spasi:

```
menit jam days-of-monthbulan days-of-weektahun
```

Contoh ekspresi cron

Menit	Jam	Hari dalam sebulan	Bulan	Hari dalam seminggu	Tahun	Arti
0	0	?	*	MON-FRI	*	Menjalankan alur kerja pada tengah malam (UTC+0) setiap Senin sampai Jumat.
0	2	*	*	?	*	Menjalankan alur kerja pada pukul 2:00 pagi (UTC+0) setiap hari.
15	22	*	*	?	*	Menjalankan alur kerja pada pukul 10:15 malam (UTC+0) setiap hari.
0/30	22-2	?	*	SAT-MATAHARI	*	Menjalankan alur kerja setiap 30 menit Sabtu

Menit	Jam	Hari dalam sebulan	Bulan	Hari dalam seminggu	Tahun	Arti
						hingga Minggu antara pukul 10:00 malam pada hari mulai dan 2:00 pagi pada hari berikutnya (UTC+0).
45	13	L	*	?	2023-2027	Menjalankan alur kerja pada pukul 1:45 siang (UTC+0) pada hari terakhir bulan antara tahun 2023 dan 2027 inklusif.

Saat menentukan ekspresi cron di CodeCatalyst, pastikan Anda mengikuti panduan ini:

- Tentukan ekspresi cron tunggal per `SCHEDULE` pemicu.
- Lampirkan ekspresi cron dalam tanda kutip ganda (") di editor YAMAL.
- Tentukan waktu di Coordinated Universal Time (UTC). Zona waktu lain tidak didukung.
- Konfigurasikan setidaknya 30 menit di antara proses. Irama yang lebih cepat tidak didukung.

- Tentukan *days-of-week*bidang *days-of-month*atau, tetapi tidak keduanya. Jika Anda menentukan nilai atau tanda bintang (*) di salah satu bidang, Anda harus menggunakan tanda tanya (?) di bidang lainnya. Tanda bintang berarti 'semua' dan tanda tanya berarti 'apa saja'.

Untuk lebih banyak contoh ekspresi cron dan informasi tentang wildcard seperti?,*, danL, lihat referensi ekspresi Cron di Panduan Pengguna Amazon EventBridge . Ekspresi cron masuk EventBridge dan CodeCatalyst bekerja dengan cara yang persis sama.

Untuk contoh pemicu jadwal, lihatContoh pemicu.

UI yang sesuai: editor visual/diagram alur kerja/pemicu/Jadwal

Tindakan

Urutan satu atau lebih tindakan untuk alur kerja ini. CodeCatalyst mendukung beberapa jenis tindakan, seperti membangun dan menguji tindakan, yang menawarkan berbagai jenis fungsionalitas. Setiap jenis tindakan memiliki:

- `Identifier`properti yang menunjukkan ID unik dan hard-code tindakan. Misalnya, `aws/build@v1` mengidentifikasi tindakan build.
- `Configuration`bagian yang berisi properti yang spesifik untuk tindakan.

Untuk informasi selengkapnya tentang setiap jenis tindakan, lihatJenis tindakan. Jenis tindakanTopik memiliki tautan ke dokumentasi untuk setiap tindakan.

Berikut ini adalah referensi YAMAL untuk tindakan dan kelompok tindakan dalam file definisi alur kerja.

```
Name: MyWorkflow
SchemaVersion: 1.0
...
Actions:
  action-or-gate-name:
    Identifier: identifier
    Configuration:
    ...
  #Action groups
  action-group-name:
    Actions:
```

```
      ...
```

action-or-gate-name

(Actions/*action-or-gate-name*)

(Diperlukan)

Ganti *nama tindakan* dengan nama yang ingin Anda berikan tindakan. Nama tindakan harus unik dalam alur kerja, dan hanya boleh menyertakan karakter alfanumerik, tanda hubung, dan garis bawah. Untuk informasi selengkapnya tentang aturan sintaks, lihatPedoman sintaks YAMAL.

Untuk informasi selengkapnya tentang praktik penamaan untuk tindakan, termasuk pembatasan, lihataction-or-gate-name.

UI yang sesuai: editor ***visual/nama tindakan/tab Konfigurasi/Nama*** tindakan atau Nama tampilan Tindakan

action-group-name

(Actions/*action-group-name*)

(Opsional)

Grup aksi berisi satu atau lebih tindakan. Mengelompokkan tindakan ke dalam grup tindakan membantu Anda menjaga alur kerja tetap teratur, dan juga memungkinkan Anda mengonfigurasi dependensi di antara grup yang berbeda.

Ganti *action-group-name*dengan nama yang ingin Anda berikan pada grup aksi. Nama grup tindakan harus unik dalam alur kerja, dan hanya boleh menyertakan karakter alfanumerik, tanda hubung, dan garis bawah. Untuk informasi selengkapnya tentang aturan sintaks, lihatPedoman sintaks YAMAL.

Untuk informasi selengkapnya tentang grup aksi, lihatMengelompokkan tindakan ke dalam kelompok aksi.

UI yang sesuai: tidak ada

Lacak dan atur pekerjaan dengan masalah di CodeCatalyst

Di CodeCatalyst, Anda dapat memantau fitur, bug, dan pekerjaan lain yang terlibat dalam proyek Anda. Setiap karya disimpan dalam catatan berbeda yang disebut masalah. Anda dapat memecah masalah menjadi tujuan yang lebih kecil dengan menambahkan daftar tugas ke dalamnya. Setiap masalah dapat memiliki deskripsi, penerima tugas, status, dan properti lainnya, yang dapat Anda cari, kelompokkan, dan filter. Anda dapat melihat masalah menggunakan tampilan default, atau Anda dapat membuat tampilan sendiri dengan pemfilteran, pengurutan, atau pengelompokan kustom. Untuk informasi selengkapnya tentang konsep yang terkait dengan masalah, lihatIsu konsep. Untuk mempelajari cara membuat edisi pertama Anda, lihatMembuat masalah di CodeCatalyst.

Berikut adalah salah satu alur kerja yang mungkin untuk tim yang menggunakan masalah:

Jorge Souza adalah pengembang yang bekerja dalam sebuah proyek. Dia dan rekan-rekan anggota proyek Li Juan, Mateo Jackson, dan Wang Xiulan berkolaborasi untuk menentukan pekerjaan apa yang perlu dilakukan. Setiap hari, dia dan rekan-rekan pengembangnya mengadakan pertemuan sinkronisasi, yang dipimpin oleh Wang Xiulan. Mereka menarik papan dengan menavigasi ke salah satu pandangan tim mereka tentang dewan. Dengan membuat tampilan, pengguna dan tim dapat menyimpan filter, pengelompokan, dan penyortiran masalah untuk dengan mudah melihat masalah yang memenuhi kriteria yang ditentukan. Pandangan mereka berisi masalah yang dikelompokkan berdasarkan Penerima Tugas dan diurutkan berdasarkan Prioritas untuk menunjukkan masalah dan status masalah yang paling penting untuk setiap pengembang. Karena Jorge diberi tugas untuk diselesaikan, ia merencanakan pekerjaannya dengan menciptakan masalah untuk setiap tugas. Saat membuat masalah, Jorge dapat memilih Status, Prioritas, dan upaya Estimasi pekerjaan yang sesuai. Untuk masalah yang lebih besar, Jorge menambahkan tugas ke masalah ini, untuk memecah pekerjaan menjadi tujuan yang lebih kecil. Jorge menciptakan masalahnya dengan status draf, seperti backlog, karena dia tidak berencana untuk segera memulainya. Masalah dalam status draf muncul pada tampilan Draf di mana mereka akan direncanakan dan diprioritaskan. Setelah Jorge siap untuk memulai pekerjaan, ia memindahkan masalah yang sesuai ke dewan dengan memperbarui statusnya ke status di kategori lain (Tidak Dimulai, Dimulai, atau Selesai). Saat setiap tugas sedang dikerjakan, tim dapat memfilter berdasarkan judul, status, penerima tugas, label, prioritas, dan estimasi untuk menemukan masalah tertentu atau masalah serupa yang cocok dengan parameter yang ditentukan. Dengan menggunakan papan, Jorge dan timnya dapat melihat jumlah tugas yang diselesaikan untuk setiap masalah, dan melacak day-to-day kemajuan dengan menyeret setiap masalah dari satu status ke status berikutnya hingga tugas selesai. Saat proyek berlangsung, masalah yang sudah selesai terakumulasi dalam status Selesai. Wang Xiulan memutuskan untuk menghapusnya dari tampilan dengan mengarsipkannya menggunakan tombol arsip cepat, sehingga

pengembang dapat fokus pada masalah yang terkait dengan pekerjaan saat ini dan yang akan datang.

Saat merencanakan pekerjaan mereka, pengembang yang mengerjakan proyek memilih Urutkan berdasarkan dan Kelompokkan berdasarkan untuk menemukan masalah yang ingin mereka pindahkan dari backlog ke papan tulis. Mereka mungkin memilih untuk menambahkan masalah ke papan berdasarkan permintaan pelanggan prioritas tertinggi, sehingga mereka mengelompokkan papan berdasarkan label permintaan Pelanggan dan mengurutkan berdasarkan Prioritas. Mereka mungkin juga mengurutkan berdasarkan perkiraan untuk memastikan bahwa mereka mengambil volume pekerjaan yang dapat mereka capai. Manajer proyek, Saanvi Sarkar, secara teratur meninjau dan merawat backlog untuk membantu memastikan bahwa prioritas secara akurat mencerminkan pentingnya setiap masalah bagi keberhasilan proyek.

Topik

Isu konsep

Membuat masalah adalah cara cepat dan efisien untuk melacak pekerjaan yang dilakukan dalam suatu proyek. Anda dapat menggunakan masalah untuk membantu Anda mendiskusikan pekerjaan dalam rapat sinkronisasi harian, memprioritaskan pekerjaan, dan banyak lagi.

Halaman ini mencakup daftar konsep yang akan membantu Anda menggunakan masalah secara efektif CodeCatalyst.

Masalah aktif

Masalah aktif adalah masalah yang merupakan masalah apa pun yang tidak dalam status Draf atau diarsipkan. Dengan kata lain, masalah aktif adalah masalah dengan status di salah satu kategori status berikut: Tidak dimulai, Dimulai, dan Selesai. Untuk informasi selengkapnya tentang status dan kategori status, lihatKategori status dan status.

Anda dapat melihat semua masalah aktif dalam proyek Anda dari tampilan Masalah aktif default.

Masalah yang diarsipkan

Masalah yang diarsipkan adalah masalah yang tidak lagi relevan dengan proyek Anda. Misalnya, Anda dapat mengarsipkan masalah jika masalah selesai dan Anda tidak perlu lagi melihatnya di kolom Selesai, atau jika itu dibuat karena kesalahan. Masalah yang diarsipkan dapat dibatalkan jika diperlukan.

Penerima tugas

Penerima tugas adalah orang yang ditugaskan untuk masalah tersebut. Jika orang tersebut tidak muncul dalam daftar saat Anda mencarinya, mereka belum ditambahkan ke proyek Anda. Untuk menambahkannya, lihatMengundang pengguna ke proyek. Untuk mengaktifkan beberapa penerima tugas ke suatu masalah, lihat. Mengaktifkan atau menonaktifkan beberapa penerima tugas Masalah dengan beberapa penerima tugas akan muncul di papan Anda dengan avatar berwarna berbeda, masing-masing mewakili salah satu penerima tugas.

Bidang kustom

Bidang kustom memungkinkan Anda untuk menyesuaikan atribut yang berbeda dari suatu masalah sesuai dengan kebutuhan Anda untuk melacak dan memelihara masalah dalam proyek. Misalnya, Anda dapat menambahkan bidang untuk pemetaan jalan, tanggal jatuh tempo tertentu, atau bidang pemohon.

Estimasi

Dalam perkembangan tangkas, perkiraan tersebut dikenal sebagai poin cerita. Anda dapat menggunakan estimasi untuk suatu masalah untuk mewakili jumlah pekerjaan yang diperlukan, selain ambiguitas dan kompleksitas masalah. Pertimbangkan untuk menggunakan perkiraan yang lebih tinggi untuk masalah dengan risiko, kesulitan, dan tidak diketahui yang lebih besar.

Untuk informasi selengkapnya tentang jenis estimasi dan cara mengonfigurasinya, lihatMengkonfigurasi estimasi upaya masalah.

Isu

Masalah adalah catatan yang melacak pekerjaan yang terkait dengan proyek Anda. Anda dapat membuat masalah untuk fitur, tugas, bug, atau badan pekerjaan lain yang terkait dengan proyek Anda. Jika Anda menggunakan pengembangan tangkas, masalah juga dapat menggambarkan kisah epik atau pengguna.

Label

Label digunakan untuk mengelompokkan, mengurutkan, dan memfilter masalah. Anda dapat memasukkan nama label baru atau memilih salah satu label dari daftar yang diisi. Daftar ini terdiri dari label yang baru digunakan dalam proyek. Masalah dapat memiliki beberapa label, dan label dapat dihapus dari masalah. Untuk menyesuaikan label, lihatMengkategorikan pekerjaan dengan label.

Prioritas

Prioritas mengacu pada tingkat kepentingan masalah. Ada empat opsi: Rendah, Sedang, Tinggi, dan Tidak ada prioritas.

Kategori status dan status

Status adalah status masalah saat ini dan digunakan untuk memeriksa kemajuan masalah dengan cepat melalui siklus hidupnya, dari awal hingga penyelesaian. Semua masalah harus memiliki status, dan setiap status termasuk dalam kategori status. Kategori status digunakan untuk membantu mengatur status Anda dan mengisi tampilan masalah default.

Ada lima status default dan empat kategori status di CodeCatalyst. Anda dapat membuat status lain, tetapi Anda tidak dapat membuat kategori status lainnya. Daftar berikut berisi status default dan kategori statusnya dalam tanda kurung: Backlog (Draft), To do (Not started), In progress (Started), In review (Started), and Done (Completed).

Untuk informasi lebih lanjut tentang bekerja dengan status, lihatMelacak pekerjaan dengan status khusus.

Tugas

Tugas dapat ditambahkan ke masalah untuk memecah lebih lanjut dan mengatur pekerjaan masalah itu. Anda dapat menambahkan tugas ke masalah saat pembuatan, atau menambahkan tugas ke masalah yang ada. Saat melihat masalah, Anda dapat menyusun ulang, menghapus, atau menandai tugasnya sebagai selesai.

Tampilan

Masalah dalam CodeCatalyst proyek Anda ditampilkan dalam tampilan. Tampilan dapat berupa tampilan kisi yang menampilkan masalah dalam format daftar atau tampilan papan yang menampilkan masalah sebagai ubin di kolom yang diatur berdasarkan status masalah. Ada

empat tampilan default, dan Anda dapat membuat tampilan Anda sendiri dengan pengelompokan, pemfilteran, dan pengurutan khusus. Daftar berikut berisi rincian tentang empat tampilan default.

- Tampilan Draf adalah tampilan kisi yang menunjukkan masalah yang saat ini tidak sedang dikerjakan. Masalah apa pun yang dibuat dengan status dalam kategori status Draf muncul di tampilan ini. Tampilan ini dapat digunakan oleh tim untuk melihat masalah mana yang masih ditentukan atau sedang menunggu untuk ditugaskan dan dikerjakan.
- Tampilan Masalah aktif adalah tampilan papan dari semua masalah yang saat ini sedang dikerjakan. Masalah apa pun dengan status di kategori status Tidak dimulai, Dimulai, atau Selesai akan muncul di tampilan ini.
- Tampilan Semua masalah adalah tampilan kisi yang menunjukkan semua masalah dalam proyek, baik konsep maupun masalah aktif.
- Tampilan yang diarsipkan menunjukkan semua masalah yang diarsipkan.

Melacak pekerjaan dengan masalah

Anda dapat merencanakan dan melacak pekerjaan Anda pada suatu proyek dengan menggunakan masalah. Setiap masalah adalah karya yang disimpan dalam catatan yang berbeda. Masalah dapat dipecah menjadi tugas untuk lebih mengatur dan melacak pekerjaan masalah itu. Anda juga dapat membuat tautan antar masalah untuk membantu Anda melacak pekerjaan terkait, menambahkan label untuk membantu Anda mengatur dan mengkategorikan pekerjaan, masalah kelompok, menetapkan prioritas untuk bekerja, dan menunjukkan apakah pekerjaan diblokir.

Ketika Anda siap untuk mengerjakan masalah atau serangkaian masalah, Anda dapat memperkirakan pekerjaan, menetapkannya kepada pengguna, dan menambahkan komentar untuk membantu orang lain memahami pekerjaan dan kemajuannya. Anda juga dapat mengekspor masalah untuk membantu membawa informasi yang dikandungnya ke format lain.

Membuat masalah di CodeCatalyst

Tim pengembangan menciptakan masalah untuk membantu melacak dan mengelola pekerjaan mereka. Anda dapat membuat masalah dalam proyek berdasarkan kebutuhan Anda. Misalnya, Anda dapat membuat masalah untuk melacak pembaruan variabel dalam kode Anda. Anda dapat menetapkan masalah kepada pengguna lain dalam proyek, menggunakan label untuk membantu Anda melacak pekerjaan Anda, dan banyak lagi.

Ikuti petunjuk ini untuk membuat masalah di CodeCatalyst.

Untuk membuat masalah

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke proyek tempat Anda ingin membuat masalah.
3. Pada halaman beranda proyek, pilih Buat masalah. Atau, di panel navigasi, pilih Masalah.
4. Pilih Buat masalah.

> **Note**
>
> Anda juga dapat menambahkan masalah sebaris saat menggunakan tampilan kisi.

5. Masukkan judul untuk masalah ini.
6. (Opsional) Masukkan Deskripsi. Anda dapat menggunakan Markdown untuk menambahkan pemformatan.
7. (Opsional) Pilih Status, Prioritas, dan Estimasi untuk masalah ini.

> **Note**
>
> Jika pengaturan estimasi proyek disetel ke Sembunyikan estimasi, tidak akan ada bidang Estimasi.

8. (Opsional) Tambahkan tugas ke masalah. Tugas dapat digunakan untuk memecah pekerjaan suatu masalah menjadi tujuan yang lebih kecil. Untuk menambahkan tugas, pilih + Tambahkan tugas. Kemudian, masukkan nama tugas di bidang teks dan tekan enter. Setelah menambahkan tugas, Anda dapat menandainya sebagai selesai dengan memilih kotak centang, atau menyusun ulang dengan memilih dan menyeret tugas dari sisi kiri kotak centang.
9. (Opsional) Tambahkan label yang ada atau buat label baru dan tambahkan dengan memilih + Tambahkan label.
    a. Untuk menambahkan label yang ada, pilih label dari daftar. Anda dapat memasukkan istilah pencarian di bidang untuk mencari semua label yang berisi istilah itu dalam proyek.
    b. Untuk membuat label baru dan menambahkannya, masukkan nama label yang ingin Anda buat di bidang pencarian dan tekan enter.
10. (Opsional) Tambahkan penerima tugas dengan memilih + Tambahkan penerima tugas. Anda dapat dengan cepat menambahkan diri Anda sebagai penerima tugas dengan memilih + Tambahkan saya.

> **Tip**
>
> Anda dapat memilih untuk menetapkan masalah ke Amazon Q agar Amazon Q mencoba menyelesaikan masalah tersebut. Untuk informasi selengkapnya, lihat Tutorial: Menggunakan fitur AI CodeCatalyst generatif untuk mempercepat pekerjaan pengembangan Anda. Fitur ini hanya tersedia di Wilayah AS Barat (Oregon). Fungsionalitas ini mengharuskan fitur AI generatif diaktifkan untuk ruang tersebut. Untuk informasi selengkapnya, lihat Mengelola fitur AI generatif.

11. (Opsional) Tambahkan bidang kustom yang ada atau buat bidang kustom baru. Masalah dapat memiliki beberapa bidang khusus.

    a. Untuk menambahkan bidang kustom yang ada, pilih bidang kustom dari daftar. Anda dapat memasukkan istilah pencarian di bidang untuk mencari semua bidang kustom yang berisi istilah itu dalam proyek.

    b. Untuk membuat bidang khusus baru dan menambahkannya, masukkan nama bidang khusus yang ingin Anda buat di bidang pencarian dan tekan enter. Kemudian pilih jenis bidang khusus yang ingin Anda buat dan tetapkan nilainya.

12. Pilih Buat masalah. Pemberitahuan muncul di sudut kanan bawah: Jika masalah berhasil dibuat, pesan konfirmasi muncul yang mengatakan masalah berhasil dibuat. Jika masalah tidak berhasil dibuat, pesan kesalahan dengan alasan kegagalan muncul. Anda kemudian dapat memilih Coba lagi untuk mengedit dan mencoba lagi membuat masalah, atau memilih Buang untuk membuang masalah. Kedua opsi akan mengabaikan notifikasi.

> **Note**
>
> Anda tidak dapat menautkan permintaan tarik ke masalah saat Anda membuatnya. Namun, Anda dapat mengeditnya setelah Anda membuatnya untuk menambahkan tautan ke permintaan tarik.

Praktik terbaik saat membuat dan menangani masalah yang ditetapkan ke Amazon Q

Saat Anda membuat masalah, terkadang beberapa di antaranya berlama-lama. Penyebabnya bisa kompleks dan bervariasi. Terkadang itu karena tidak jelas siapa yang harus mengerjakannya. Di lain waktu masalah ini membutuhkan penelitian atau keahlian dengan bagian tertentu dari basis kode dan kandidat terbaik untuk pekerjaan sibuk dengan masalah lain. Seringkali ada pekerjaan

mendesak lainnya yang harus dihadiri terlebih dahulu. Salah satu atau semua penyebab ini dapat mengakibatkan masalah yang tidak dikerjakan. CodeCatalyst Termasuk integrasi dengan asisten AI generatif yang disebut Amazon Q yang dapat menganalisis masalah berdasarkan judul dan deskripsinya. Jika Anda menetapkan masalah ke Amazon Q, itu akan mencoba membuat solusi draf untuk Anda evaluasi. Ini dapat membantu Anda dan tim Anda untuk fokus dan mengoptimalkan pekerjaan pada masalah yang memerlukan perhatian Anda, sementara Amazon Q bekerja pada solusi untuk masalah yang tidak memiliki sumber daya untuk segera ditangani.

> **Note**
>
> Didukung oleh Amazon Bedrock: AWS mengimplementasikan deteksi penyalahgunaan otomatis. Karena fitur Tetapkan masalah ke Amazon Q dengan Agen Pengembang Amazon Q untuk pengembangan perangkat lunak dibangun di Amazon Bedrock, pengguna dapat memanfaatkan sepenuhnya kontrol yang diterapkan di Amazon Bedrock untuk menegakkan keselamatan, keamanan, dan penggunaan kecerdasan buatan (AI) yang bertanggung jawab.

Amazon Q berkinerja terbaik pada masalah sederhana dan masalah langsung. Untuk hasil terbaik, gunakan bahasa sederhana untuk menjelaskan dengan jelas apa yang ingin Anda lakukan. Berikut ini adalah beberapa praktik terbaik untuk membantu Anda membuat masalah yang dioptimalkan agar Amazon Q dapat dikerjakan.

> **Important**
>
> Fitur AI generatif hanya tersedia di Wilayah AS Barat (Oregon).

- Tetap sederhana. Amazon Q melakukan yang terbaik dengan perubahan kode sederhana dan perbaikan yang dapat dijelaskan dalam judul dan deskripsi masalah. Jangan menetapkan masalah dengan judul yang tidak jelas atau deskripsi yang terlalu berbunga-bunga atau kontradiktif.
- Jadilah spesifik. Semakin banyak informasi yang dapat Anda berikan tentang perubahan tepat yang diperlukan untuk menyelesaikan masalah, semakin besar kemungkinan Amazon Q akan dapat membuat solusi yang memecahkan masalah. Jika memungkinkan, sertakan detail spesifik seperti nama API yang ingin diubah, metode yang ingin diperbarui, pengujian yang memerlukan perubahan, dan detail lainnya yang dapat Anda pikirkan.
- Pastikan Anda memiliki semua detail yang disertakan dalam judul dan deskripsi masalah sebelum menetapkannya ke Amazon Q. Anda tidak dapat mengubah judul atau deskripsi masalah setelah

Anda menetapkannya ke Amazon Q, jadi pastikan Anda memiliki semua informasi yang diperlukan dalam masalah sebelum Anda menetapkannya ke Amazon Q.

- Hanya tetapkan masalah yang memerlukan perubahan kode dalam satu repositori sumber. Amazon Q hanya dapat bekerja pada kode dalam satu repositori sumber di. CodeCatalyst Repositori tertaut tidak didukung. Pastikan bahwa masalah hanya memerlukan perubahan dalam satu repositori sumber sebelum Anda menetapkan masalah itu ke Amazon Q.
- Gunakan default yang disarankan oleh Amazon Q untuk menyetujui setiap langkah. Secara default, Amazon Q akan memerlukan persetujuan Anda untuk setiap langkah yang diperlukan. Ini memungkinkan Anda untuk berinteraksi dengan Amazon Q dalam komentar tidak hanya tentang masalah ini, tetapi juga pada permintaan tarik apa pun yang dibuatnya. Ini memberikan pengalaman yang lebih interaktif dengan Amazon Q yang membantu Anda menyesuaikan pendekatannya dan menyempurnakan kode yang dibuatnya untuk menyelesaikan masalah.

  > **Note**
  >
  > Amazon Q tidak menanggapi komentar individu dalam masalah atau permintaan tarik, tetapi akan meninjaunya ketika diminta untuk mempertimbangkan kembali pendekatannya atau membuat revisi.

- Selalu hati-hati meninjau pendekatan yang disarankan oleh Amazon Q. Setelah Anda menyetujui pendekatannya, Amazon Q akan mulai bekerja menghasilkan kode berdasarkan pendekatan itu. Pastikan pendekatannya tampak benar dan mencakup semua detail yang Anda harapkan sebelum Anda memberi tahu Amazon Q untuk melanjutkan.
- Pastikan untuk hanya mengizinkan Amazon Q bekerja pada alur kerja jika Anda tidak memiliki alur kerja yang ada yang mungkin menerapkannya sebelum ditinjau. Proyek Anda mungkin memiliki alur kerja yang dikonfigurasi untuk mulai berjalan pada peristiwa permintaan tarik. Jika demikian, permintaan tarik apa pun yang dibuat Amazon Q yang menyertakan pembuatan atau pembaruan alur kerja YAMAL dapat memulai menjalankan alur kerja yang disertakan dalam permintaan tarik. Sebagai praktik terbaik, jangan memilih untuk mengizinkan Amazon Q bekerja pada file alur kerja kecuali Anda yakin tidak ada alur kerja dalam proyek Anda yang akan secara otomatis menjalankan alur kerja ini sebelum Anda meninjau dan menyetujui permintaan tarik yang dibuatnya.

Untuk informasi selengkapnya, lihat Tutorial: Menggunakan fitur AI CodeCatalyst generatif untuk mempercepat pekerjaan pengembangan Anda dan Mengelola fitur AI generatif.

Memperkirakan masalah

Dalam perkembangan tangkas, perkiraan tersebut dikenal sebagai poin cerita. Anda dapat menggunakan estimasi untuk suatu masalah untuk mewakili jumlah pekerjaan yang diperlukan, selain ambiguitas dan kompleksitas masalah. Pertimbangkan untuk menggunakan perkiraan yang lebih tinggi untuk masalah dengan risiko, kesulitan, dan tidak diketahui yang lebih besar.

Sebelum Anda dapat mulai memperkirakan masalah Anda, Anda harus terlebih dahulu memilih jenis perkiraan yang ingin Anda gunakan untuk proyek Anda. Ada dua jenis untuk dipilih secara default. Untuk menggunakan ukuran T-shirt atau sekuensing Fibonacci secara efektif, tim Anda harus menyelaraskan apa yang diwakili oleh setiap ukuran. Putuskan bersama apa yang masing-masing estimasi mewakili Anda, dan kemudian mulai menerapkan perkiraan tersebut untuk setiap masalah. Pertimbangkan untuk meninjau secara berkala

Ikuti langkah-langkah ini untuk mengonfigurasi pengaturan estimasi upaya untuk masalah di CodeCatalyst.

Untuk mengonfigurasi estimasi upaya untuk masalah

1. Di panel navigasi, pilih Masalah.
2. Pilih Masalah aktif untuk membuka menu tarik-turun pengalih tampilan masalah dan pilih Pengaturan.
3. Di Estimasi di bagian Pengaturan dasar, pilih bagaimana nilai estimasi akan ditampilkan. Jenis perkiraan yang tersedia adalah ukuran T-shirt, pengurutan Fibonacci, atau perkiraan Sembunyikan. Jika pengaturan estimasi proyek disetel ke Sembunyikan estimasi, tidak akan ada bidang Estimasi dalam masalah untuk proyek.

   Ketika jenis estimasi diperbarui, tidak ada data yang akan hilang dan nilai estimasi semua masalah akan dikonversi secara otomatis. Pemetaan konversi ditunjukkan pada tabel berikut.

Ukuran kaos	Urutan Fibonacci
XS	1
XS	2
D	3
M	5

Ukuran kaos	Urutan Fibonacci
L	8
XL	13

Untuk menambah atau mengubah perkiraan masalah, Anda dapat mengedit masalah.

Mengedit dan berkolaborasi dalam masalah di CodeCatalyst

Daftar Isi

Mengedit masalah

Ikuti langkah-langkah ini untuk mengedit judul, deskripsi, status, penerima tugas, prioritas, perkiraan, atau label masalah.

Untuk mengedit masalah

1. Pilih masalah yang ingin Anda edit untuk melihat detail masalah. Untuk bantuan dalam menemukan masalah Anda, lihatMenemukan dan melihat masalah.
2. Untuk mengedit judul terbitan, pilih judul, masukkan judul baru, dan tekan enter.
3. Untuk mengedit deskripsi, pilih deskripsi, masukkan deskripsi baru, dan tekan enter. Anda dapat menggunakan Markdown untuk menambahkan pemformatan.
4. Di Tugas, Anda dapat melihat dan mengelola tugas untuk masalah tersebut. Untuk informasi selengkapnya, lihat Mengelola tugas tentang masalah.
5. Untuk mengedit Status, Estimasi, atau Prioritas, pilih opsi dari menu tarik-turun masing-masing.

6. Di Label, Anda dapat menambahkan label yang ada, membuat label baru, atau menghapus label.

   a. Untuk menambahkan label yang ada, pilih + Tambahkan label dan pilih label dari daftar. Anda dapat memasukkan istilah pencarian di bidang untuk mencari semua label yang berisi istilah itu dalam proyek.

   b. Untuk membuat label baru dan menambahkannya, pilih + Tambahkan label masukkan nama label yang ingin Anda buat di bidang pencarian dan tekan enter.

   c. Untuk menghapus label, pilih ikon X di sebelah label yang ingin Anda hapus. Jika Anda menghapus label dari semua masalah, label akan muncul di bagian Label yang tidak digunakan di bagian Label dari pengaturan masalah. Label yang tidak digunakan muncul di akhir daftar label saat menggunakan filter atau menambahkan label ke masalah. Anda dapat menemukan ikhtisar semua label (bekas dan tidak terpakai) dan masalah yang ada di pengaturan masalah.

7. Untuk menetapkan masalah, pilih + Tambahkan penerima tugas di bagian Penerima Tugas, lalu cari dan pilih penerima tugas dari daftar. Anda dapat memilih + Tambahkan saya untuk menambahkan diri Anda dengan cepat sebagai penerima tugas.

8. Di Lampiran, Anda dapat menambahkan, mengunduh, atau menghapus lampiran. Untuk informasi selengkapnya, lihat Bekerja dengan lampiran.

9. Untuk menautkan permintaan tarik, pilih Permintaan tarik tautan, lalu pilih permintaan tarik dari daftar atau masukkan URL atau IDnya. Untuk memutuskan tautan permintaan tarik, pilih ikon batalkan tautan.

   > **Tip**
   >
   > Setelah menambahkan tautan ke permintaan tarik ke masalah, Anda dapat dengan cepat menavigasi ke sana dengan memilih ID-nya dalam daftar permintaan tarik tertaut. Anda dapat menggunakan URL permintaan tarik untuk menautkan permintaan tarik yang ada di proyek yang berbeda dari papan masalah, tetapi hanya pengguna yang merupakan anggota proyek tersebut yang dapat melihat atau menavigasi ke permintaan tarik tersebut.

10. (Opsional) Tambahkan dan atur bidang kustom yang ada, buat bidang kustom baru, atau hapus bidang khusus. Masalah dapat memiliki beberapa bidang khusus.

a. Untuk menambahkan bidang kustom yang ada, pilih bidang kustom dari daftar. Anda dapat memasukkan istilah pencarian di bidang untuk mencari semua bidang kustom yang berisi istilah itu dalam proyek.

b. Untuk membuat bidang kustom baru dan menambahkannya, masukkan nama bidang khusus yang ingin Anda buat di bidang pencarian dan tekan enter. Kemudian pilih jenis bidang khusus yang ingin Anda buat dan tetapkan nilainya.

c. Untuk menghapus bidang kustom, pilih ikon X di sebelah bidang kustom yang ingin Anda hapus. Jika Anda menghapus bidang khusus dari semua masalah, bidang kustom akan dihapus dan Anda tidak akan lagi melihatnya saat memfilter.

Bekerja dengan lampiran

Anda dapat menambahkan lampiran ke masalah CodeCatalyst untuk membuat file terkait mudah diakses. Gunakan prosedur berikut untuk mengelola lampiran untuk masalah.

Ukuran lampiran yang ditambahkan ke masalah diperhitungkan dalam kuota penyimpanan ruang Anda. Untuk informasi tentang melihat dan mengelola lampiran untuk proyek Anda, lihatMelihat dan mengelola lampiran.

> **Important**
>
> Lampiran pada masalah tidak dipindai atau dianalisis oleh Amazon. CodeCatalyst Setiap pengguna dapat menambahkan lampiran ke masalah yang berpotensi mengandung kode atau konten berbahaya. Pastikan pengguna mengetahui praktik terbaik dalam mengelola lampiran dan menjaga terhadap kode, konten, atau virus berbahaya.

Untuk menambah, mengunduh, atau menghapus lampiran

1. Pilih masalah yang ingin Anda kelola lampirannya. Untuk bantuan dalam menemukan masalah Anda, lihatMenemukan dan melihat masalah.
2. Untuk menambahkan lampiran, pilih Unggah file. Arahkan ke file di file explorer sistem operasi Anda dan pilih. Pilih Buka untuk menambahkannya sebagai lampiran. Untuk informasi kuota, seperti ukuran lampiran maksimum, lihatKuota untuk masalah di CodeCatalyst.

   Perhatikan batasan berikut untuk nama file lampiran dan jenis konten:

- Karakter berikut tidak diizinkan dalam nama file:
  - Karakter kontrol: `0x00-0x1f` dan `0x80-0x9f`
  - Karakter yang dicadangkan: `/?,<,>,\,:,*,|`, dan `"`
  - Nama file cadangan Unix: dan `. ..`
  - Periode dan spasi tertinggal
  - Nama file yang dicadangkan Windows: `CON, PRN, AUX, NUL, COM1, COM2, COM3, COM4, COM5, COM6, COM7, COM8, COM9, LPT1, LPT2, LPT3, LPT4, LPT5, LPT6, LPT7, LPT8, and LPT9`
- Jenis konten lampiran harus mematuhi pola tipe media berikut:

  ```
  media-type = type "/" [tree "."] subtype ["+" suffix]* [";" parameter];
  ```

  Misalnya, `text/html; charset=UTF-8`.

3. Untuk mengunduh lampiran, pilih menu elips di sebelah lampiran yang ingin Anda unduh dan pilih Unduh.
4. Untuk menyalin URL lampiran, pilih menu elips di sebelah lampiran yang ingin Anda salin URL dan pilih Salin URL.
5. Untuk menghapus lampiran, pilih menu elips di sebelah lampiran yang ingin Anda hapus dan pilih Hapus.

Melihat dan mengelola lampiran

Anda dapat melihat tabel dengan setiap lampiran yang ditambahkan ke masalah dalam proyek Anda dalam pengaturan masalah. Tabel ini mencakup detail setiap lampiran, termasuk informasi seperti jenis konten, saat ditambahkan, masalah yang ditambahkan dan statusnya, dan ukuran file.

Tabel ini dapat digunakan untuk dengan mudah mengidentifikasi lampiran besar pada masalah yang diselesaikan atau diarsipkan untuk menghapusnya untuk mengosongkan penyimpanan ruang.

> **Important**
>
> Lampiran pada masalah tidak dipindai atau dianalisis oleh Amazon. CodeCatalyst Setiap pengguna dapat menambahkan lampiran ke masalah yang berpotensi mengandung kode

atau konten berbahaya. Pastikan pengguna mengetahui praktik terbaik dalam mengelola lampiran dan menjaga terhadap kode, konten, atau virus berbahaya.

Untuk melihat dan mengelola semua lampiran masalah dalam proyek

1. Di panel navigasi, pilih Masalah.
2. Pilih ikon elipsis dan pilih Pengaturan.
3. Pilih tab Lampiran.

Mengelola tugas tentang masalah

Tugas dapat ditambahkan ke masalah untuk memecah, mengatur, dan melacak pekerjaan masalah itu lebih lanjut.

Untuk mengelola tugas pada suatu masalah

1. Pilih masalah yang ingin Anda kelola tugasnya. Untuk bantuan dalam menemukan masalah Anda, lihatMenemukan dan melihat masalah.
2. Di Tugas, Anda dapat melihat dan mengelola tugas untuk masalah tersebut.
    1. Untuk menambahkan tugas, masukkan nama tugas di bidang teks dan tekan enter.
    2. Untuk menandai tugas sebagai selesai, pilih kotak centang tugas.
    3. Untuk melihat atau memperbarui detail tugas, pilih dari daftar.
    4. Untuk menyusun ulang tugas, pilih dan seret tugas dari sisi kiri kotak centang.
    5. Untuk menghapus tugas, pilih menu elips tugas dan pilih Hapus.

Menandai masalah sebagai diblokir atau tidak diblokir

Jika ada sesuatu yang mencegah Anda mengerjakan suatu masalah, Anda mungkin ingin menandainya sebagai diblokir. Misalnya, masalah Anda mungkin diblokir jika bergantung pada perubahan pada bagian lain dari basis kode Anda yang belum digabungkan.

Saat Anda menandai masalah sebagai diblokir, CodeCatalyst tambahkan label merah Diblokir ke masalah, membuatnya sangat terlihat di backlog atau arsip Anda, atau di papan Anda.

Anda dapat membuka blokir masalah saat keadaan di luar diselesaikan.

Untuk menandai masalah sebagai diblokir

1. Buka masalah yang ingin Anda tandai sebagai diblokir. Untuk bantuan dalam menemukan masalah Anda, lihatMenemukan dan melihat masalah.
2. Pilih Tindakan, lalu pilih Tandai sebagai diblokir.

Untuk membuka blokir masalah

1. Buka masalah yang ingin Anda buka blokir. Untuk bantuan dalam menemukan masalah Anda, lihatMenemukan dan melihat masalah.
2. Pilih Tindakan, lalu pilih Tandai sebagai tidak diblokir.

Menambahkan, mengedit, atau menghapus komentar

Anda dapat meninggalkan komentar tentang suatu masalah. Dalam komentar Anda dapat menandai anggota ruang lain, proyek lain di ruang, masalah terkait, dan kode.

Untuk menambahkan komentar ke masalah

1. Arahkan ke proyek Anda.
2. Di bilah navigasi pilih Masalah.
3. Pilih masalah di mana Anda ingin menambahkan komentar. Untuk bantuan dalam menemukan masalah Anda, lihatMenemukan dan melihat masalah.
4. Masukkan komentar di kolom Komentar. Anda dapat menggunakan Markdown untuk menambahkan pemformatan.
5. Pilih Kirim.

Untuk mengedit komentar

Anda dapat mengedit komentar yang Anda buat tentang masalah. Anda hanya dapat mengedit komentar yang Anda tulis.

1. Arahkan ke proyek Anda.
2. Di bilah navigasi pilih Masalah.
3. Pilih masalah di mana Anda ingin mengedit komentar. Untuk bantuan dalam menemukan masalah Anda, lihatMenemukan dan melihat masalah.

4. Untuk mengedit komentar, temukan komentar yang ingin Anda edit.

> **Tip**
>
> Anda dapat mengurutkan komentar berdasarkan yang terlama atau terbaru terlebih dahulu. Komentar dimuat 10 sekaligus.

5. Pilih ikon elipsis, lalu pilih Edit.
6. Edit komentar. Anda dapat menggunakan Markdown untuk menambahkan pemformatan.
7. Pilih Simpan. Komentar sekarang diperbarui.

Untuk menghapus komentar

Anda dapat menghapus komentar yang Anda buat tentang masalah. Anda hanya dapat menghapus komentar yang Anda tulis. Ketika komentar dihapus, nama pengguna Anda akan ditampilkan, tetapi dengan kata-kata Komentar ini telah dihapus sebagai pengganti teks komentar asli.

1. Arahkan ke proyek Anda.
2. Di bilah navigasi pilih Masalah.
3. Pilih masalah di mana Anda ingin menghapus komentar. Untuk bantuan dalam menemukan masalah Anda, lihatMenemukan dan melihat masalah.
4. Pilih ikon elipsis, pilih Hapus, lalu pilih Konfirmasi.

Menggunakan sebutan dalam komentar

Anda dapat menyebutkan anggota ruang, proyek lain di ruang, masalah terkait, dan kode dalam komentar. Melakukannya membuat tautan cepat ke pengguna atau sumber daya yang Anda sebutkan.

Ke @mention dalam komentar

1. Arahkan ke proyek Anda.
2. Di bilah navigasi pilih Masalah.
3. Pilih masalah yang ingin Anda edit untuk melihat detail masalah. Untuk bantuan dalam menemukan masalah Anda, lihatMenemukan dan melihat masalah.
4. Pilih kotak teks Tambahkan komentar.

5. Ketik @*user_name* untuk menyebutkan pengguna lain.
6. Ketik @*project_name* untuk menyebutkan proyek.
7. Ketik @*issue_name* atau @*issue_number* menyebutkan masalah lain.
8. Ketik @*file_name* untuk menyebutkan file atau kode tertentu dalam repositori sumber.

> **Note**
>
> Daftar 5 item teratas (pengguna, repositori sumber, proyek, dll) yang berisi istilah yang cocok dengan yang `@mention` akan Anda isi saat Anda mengetik.

9. Pilih item yang diinginkan yang ingin Anda sebutkan. Jalur yang menunjukkan di mana item berada akan terisi di kotak teks komentar.
10. Selesaikan komentar Anda dan pilih Kirim.

Menemukan dan melihat masalah

Bagian berikut menjelaskan cara efektif mencari dan melihat masalah dalam sebuah CodeCatalyst proyek.

Mencari masalah

Anda dapat menemukan masalah dengan mencari parameter tertentu. Untuk informasi selengkapnya tentang menyempurnakan pencarian Anda, lihatCari kode, masalah, proyek, dan pengguna di CodeCatalyst.

Untuk mencari masalah

1. Arahkan ke proyek Anda.
2. Gunakan bilah pencarian untuk mencari masalah atau informasi yang terkait dengan masalah. Anda dapat menggunakan parameter kueri untuk mempersempit pencarian Anda. Untuk informasi selengkapnya, lihat Cari kode, masalah, proyek, dan pengguna di CodeCatalyst.

Masalah penyortiran

Secara default, masalah di CodeCatalyst diurutkan berdasarkan urutan Manual. Urutan manual menampilkan masalah dalam urutan mereka dipindahkan oleh pengguna. Anda dapat menarik dan

melepas masalah saat diurutkan dalam Manual untuk mengubah urutannya. Opsi penyortiran ini sangat membantu saat merawat masalah backlog dan memprioritaskan masalah.

Tabel berikut menunjukkan bagaimana masalah dapat diurutkan dalam tampilan kisi dan papan.

Opsi penyortiran tampilan kisi	Opsi penyortiran tampilan papan
Pesanan manual	Pesanan manual
Terakhir diperbarui	Terakhir diperbarui
Prioritas	Prioritas
Estimasi	Estimasi
Judul	Judul
ID	
Status	
Diblokir	
Bidang kustom	

Gunakan prosedur berikut untuk mengubah cara masalah diurutkan.

Untuk mengurutkan masalah

1. Arahkan ke proyek Anda.
2. Di panel navigasi, pilih Masalah. Tampilan default adalah Dewan.
3. (Opsional) Pilih Masalah aktif untuk membuka menu tarik-turun pengalih tampilan masalah untuk menavigasi ke tampilan masalah yang berbeda.
4. Untuk mengurutkan tampilan grid, ada dua opsi:

    a. Pilih header bidang yang ingin Anda urutkan berdasarkan. Memilih header akan berputar antara urutan naik dan turun.

    b. Pilih menu dropdown Sort by dan pilih parameter untuk diurutkan berdasarkan. Masalah akan diurutkan dalam urutan menaik.

5. Untuk mengurutkan tampilan papan, pilih menu tarik-turun Urutkan berdasarkan dan pilih parameter untuk diurutkan berdasarkan. Masalah akan diurutkan dalam urutan menaik.

Masalah pengelompokan

Pengelompokan digunakan untuk mengatur masalah di papan dengan beberapa parameter, seperti penerima tugas, label, dan prioritas.

Untuk mengelompokkan masalah

1. Arahkan ke proyek Anda.
2. Di panel navigasi, pilih Masalah. Tampilan default adalah Dewan.
3. (Opsional) Pilih Masalah aktif untuk membuka menu tarik-turun pengalih tampilan masalah untuk menavigasi ke tampilan masalah yang berbeda.
4. Pilih Grup.
5. Di Grup menurut, pilih parameter untuk dikelompokkan berdasarkan:
   - Jika Anda memilih Penerima Tugas atau Prioritas, pilih urutan Grup.
   - Jika Anda memilih Label, pilih label dan kemudian pilih Urutan grup.
6. (Opsional) Pilih sakelar Tampilkan grup kosong untuk menampilkan atau menyembunyikan grup yang tidak memiliki masalah yang saat ini ditetapkan ke grup tersebut.
7. Tampilan diperbarui saat Anda membuat pilihan. Masalah hanya muncul di grup yang cocok dengan parameter yang dikonfigurasi.

Masalah penyaringan

Gunakan pemfilteran untuk menemukan masalah yang berisi nama, prioritas, label, bidang khusus, atau penerima tugas tertentu.

Untuk memfilter masalah

1. Arahkan ke proyek Anda.
2. Di panel navigasi, pilih Masalah.
3. (Opsional) Pilih Masalah aktif untuk membuka menu tarik-turun pengalih tampilan masalah untuk menavigasi ke tampilan masalah yang berbeda.

> **Note**
>
> Untuk memfilter berdasarkan string dalam nama atau deskripsi masalah, masukkan string ke dalam bilah pencarian masalah.

4. Pilih Filter, lalu pilih + Tambahkan filter.
5. Pilih parameter yang akan disaring. Anda dapat memilih beberapa filter dan parameter. Anda dapat mengonfigurasi filter untuk menampilkan masalah yang cocok dengan setiap filter atau filter individual dengan memilih dan atau atau. Tampilan akan diperbarui untuk menampilkan masalah yang cocok dengan filter.

Memajukan masalah

Setiap masalah memiliki siklus hidup. Pada tahun CodeCatalyst, masalah biasanya dimulai sebagai draf di backlog. Ketika pekerjaan untuk masalah itu akan dimulai, itu dipindahkan ke kategori status lain dan bergerak melalui berbagai status hingga selesai, dan kemudian diarsipkan. Anda dapat memindahkan atau memajukan masalah melalui siklus hidupnya dengan cara berikut:

- Anda dapat memindahkan masalah antara backlog dan papan.
- Anda dapat memindahkan masalah yang sedang berlangsung melalui berbagai tahap penyelesaian.
- Anda dapat mengarsipkan masalah yang telah selesai.

Memindahkan masalah di antara backlog dan papan

Anda dapat memindahkan masalah dari backlog ke papan setelah Anda mulai mengerjakan masalah tersebut. Anda juga dapat memindahkan masalah kembali ke backlog jika pekerjaan ditunda.

Untuk memindahkan masalah antara backlog dan papan

1. Arahkan ke proyek Anda.
2. Di panel navigasi, pilih Masalah. Tampilan default adalah Dewan.
3. Untuk memindahkan masalah dari papan ke backlog:

    a. Pilih masalah yang ingin Anda pindahkan. Untuk bantuan menemukan masalah Anda, lihatMenemukan dan melihat masalah.

b. Pilih Backlog dari menu Status dropdown.

4. Untuk memindahkan masalah dari backlog ke papan:

   a. Untuk menavigasi ke backlog, pilih Board dan pilih Backlog.

   b. Pilih masalah yang ingin Anda pindahkan. Untuk bantuan menemukan masalah Anda, lihatMenemukan dan melihat masalah.

   c. Pilih Tambahkan ke papan, atau pilih Status selain Backlog.

Kemajuan masalah melalui tahapan siklus hidup di papan tulis

Anda dapat memindahkan masalah di dalam papan melalui status yang berbeda hingga selesai.

Untuk memindahkan masalah di dalam papan

1. Di panel navigasi, pilih Masalah. Tampilan default adalah Dewan.
2. Lakukan salah satu hal berikut ini:
   - Seret dan lepas masalah ke status lain.
   - Pilih masalah, lalu pilih status dari menu tarik-turun Status.
   - Pilih masalah, lalu pilih Pindah ke: status ***`berikutnya`***.

Untuk informasi tentang pengarsipan masalah, lihatMengarsipkan masalah.

Memindahkan masalah antar kelompok

Anda dapat mengelompokkan masalah di Semua masalah dan tampilan Papan berdasarkan berbagai parameter. Jika masalah dikelompokkan, Anda dapat memindahkan masalah dari satu grup ke grup lainnya. Memindahkan masalah dari satu grup ke grup lainnya akan secara otomatis mengedit bidang tempat masalah dikelompokkan agar sesuai dengan grup target.

Sebagai contoh skenario, asumsikan ada perusahaan yang menggunakan yang memiliki masalah CodeCatalyst yang ditugaskan untuk dua orang, Wang Xiulan dan Saanvi Sarkar. Dewan dikelompokkan berdasarkan`Assignee`, dan ada dua kelompok, satu untuk setiap penerima tugas. Memindahkan masalah dari kelompok Wang Xiulan ke grup Saanvi Sarkar akan memperbarui penerima tugas masalah ke Saanvi Sarkar.

Mengarsipkan masalah

> **Note**
>
> Masalah tidak dihapus dalam proyek, mereka diarsipkan. Untuk menghapus masalah, Anda harus menghapus proyek.

Anda dapat mengarsipkan masalah ketika tidak lagi diperlukan dalam proyek Anda. Saat Anda mengarsipkan masalah, CodeCatalyst hapus masalah dari semua tampilan yang menyaring masalah yang diarsipkan. Masalah yang diarsipkan dapat dilihat di tampilan default Masalah yang diarsipkan, di mana masalah tersebut dapat dibatalkan jika diperlukan.

Anda mengarsipkan masalah jika:

- Anda telah menyelesaikan masalah dan tidak lagi membutuhkannya di kolom Selesai.
- Anda tidak punya rencana untuk mengerjakannya.
- Anda membuatnya dalam kesalahan.
- Anda telah mencapai jumlah maksimum masalah aktif.

Untuk mengarsipkan masalah

1. Buka masalah yang ingin Anda arsipkan. Untuk bantuan dalam menemukan masalah Anda, lihatMenemukan dan melihat masalah.
2. Pilih Tindakan, lalu pilih Pindahkan ke arsip.
3. (Opsional) Untuk mengarsipkan beberapa masalah dengan cepat dengan status Selesai, pilih elipsis vertikal di bagian atas status Selesai apa pun di papan tulis dan pilih Masalah arsip.

Untuk membatalkan arsip masalah

1. Buka masalah yang ingin Anda hapus arsipnya. Anda dapat melihat daftar masalah yang diarsipkan dengan membuka tampilan Masalah yang diarsipkan dari menu tarik-turun pengalih tampilan masalah. Untuk bantuan dalam menemukan masalah Anda, lihatMenemukan dan melihat masalah.
2. Pilih Batalkan Arsip.

Masalah ekspor

Anda dapat mengekspor masalah dalam tampilan Anda saat ini ke file.xlsx. Untuk mengekspor masalah, lakukan langkah-langkah berikut.

Untuk mengekspor masalah

1. Arahkan ke proyek Anda.
2. Di bilah navigasi pilih Masalah.
3. Pilih Masalah aktif untuk membuka menu tarik-turun pengalih tampilan masalah dan arahkan ke tampilan yang berisi masalah yang ingin Anda ekspor. Hanya masalah yang ditampilkan dalam tampilan yang akan diekspor.
4. Pilih menu elips dan pilih Ekspor ke Excel.
5. Unduhan file.xlsx. Secara default, itu berjudul nama proyek dan tanggal ekspor selesai.

Mengatur pekerjaan dengan backlog, label, dan papan

Tidak semua tim bekerja dengan cara yang sama. Anda dapat mengonfigurasi cara masalah muncul dan dapat ditetapkan di Amazon CodeCatalyst untuk membantu Anda memahami dengan tepat apa yang sedang dikerjakan dan status pekerjaan itu. Anda dapat memilih metode estimasi apa yang memungkinkan masalah sehingga pengguna Anda semua menggunakan metode estimasi yang sama. Anda dapat membuat label dan status cusotm yang juga dapat digunakan untuk memfilter tampilan pekerjaan. Bergantung pada cara kerja tim Anda, Anda dapat mengonfigurasi apakah akan mengizinkan beberapa penerima tugas untuk suatu masalah, atau hanya mengizinkan masalah ditetapkan ke satu pengguna. Anda juga dapat membuat tampilan masalah khusus sehingga pekerjaan ditampilkan dengan cara yang menunjukkan informasi yang paling relevan kepada Anda atau tim Anda.

Mengkategorikan pekerjaan dengan label

Anda dapat menyesuaikan label untuk masalah. Ini termasuk mengedit label dan mengubah warna. Label dapat membantu Anda mengkategorikan dan mengatur pekerjaan Anda. Misalnya, Anda dapat membuat label untuk aspek tertentu dari perangkat lunak Anda, atau untuk grup atau tim yang berbeda.

Topik

- Membuat label

- Mengedit label
- Menghapus label

Membuat label

Di CodeCatalyst, Anda membuat label dengan menambahkannya saat Anda membuat masalah baru atau saat Anda mengedit masalah yang ada. Lihat informasi yang lebih lengkap di Membuat masalah di CodeCatalyst dan Mengedit dan berkolaborasi dalam masalah di CodeCatalyst.

Mengedit label

Gunakan prosedur berikut untuk mengubah nama atau warna label yang ada.

Untuk mengedit label

1. Di panel navigasi, pilih Masalah.
2. Pilih Masalah aktif untuk membuka menu tarik-turun pengalih tampilan masalah dan pilih Pengaturan.
3. Pada ubin Label adalah daftar label yang digunakan dalam proyek. Pilih ikon edit di sebelah label yang ingin Anda edit. Lakukan salah satu atau beberapa hal berikut:
   a. Edit nama label.
   b. Untuk mengubah warna, pilih roda warna. Gunakan picker untuk memilih warna baru.
4. Untuk menyimpan perubahan yang Anda buat pada label, pilih ikon tanda centang.
5. Label yang diubah sekarang terlihat di daftar label yang tersedia. Anda juga dapat melihat berapa banyak masalah yang menggunakan label itu.

> **Note**
>
> Anda dapat memilih nomor yang ditampilkan di sebelah setiap label untuk menavigasi ke halaman Semua masalah dan melihat semua masalah yang berisi label tersebut.

Menghapus label

Saat ini Anda tidak dapat menghapus label masalah di CodeCatalyst. Jika Anda menghapus label dari semua masalah, label akan muncul di bagian Label yang tidak digunakan di bagian Label dari

pengaturan masalah. Label yang tidak digunakan muncul di akhir daftar label saat menggunakan filter atau menambahkan label ke masalah. Anda dapat menemukan ikhtisar semua label (bekas dan tidak terpakai) dan masalah yang ada di pengaturan masalah.

Mengatur pekerjaan dengan bidang khusus

Anda dapat membuat bidang khusus untuk membantu mengatur dan melihat pekerjaan untuk proyek Anda. Bidang khusus ditambahkan ke daftar filter yang tersedia di Filter sehingga Anda dapat memfilter masalah berdasarkan bidang khusus. Bidang kustom adalah pasangan nama dan nilai. Anda memfilter berdasarkan nama bidang kustom, dan kemudian nilai bidang kustom itu.

Masalah dapat memiliki beberapa bidang khusus.

Membuat bidang kustom

Di CodeCatalyst, Anda membuat bidang kustom dengan menambahkannya saat Anda membuat masalah atau saat Anda mengedit masalah yang ada. Untuk informasi selengkapnya, lihat Membuat masalah di CodeCatalyst dan Mengedit dan berkolaborasi dalam masalah di CodeCatalyst.

Menghapus bidang kustom

Untuk menghapus bidang kustom, Anda harus menghapus bidang kustom dari setiap masalah yang ditambahkan. Ketika bidang kustom dihapus, Anda tidak akan lagi melihat bidang kustom di Filter. Anda dapat menggunakan filter untuk melihat semua masalah dengan bidang khusus, dan menghapusnya dengan mengedit masalah. Untuk informasi selengkapnya, lihat Menemukan dan melihat masalah dan Mengedit masalah

Melacak pekerjaan dengan status khusus

Anda dapat menambahkan status khusus di papan Anda. Setiap status kustom harus termasuk dalam salah satu kategori berikut: Draf, Tidak dimulai, Dimulai, atau Selesai. Kategori status digunakan untuk membantu mengatur status dan mengisi tampilan default. Untuk informasi selengkapnya tentang status dan kategori status, lihat Kategori status dan status dan untuk informasi selengkapnya tentang tampilan, lihatMenemukan dan melihat masalah.

Untuk membuat status

1. Di panel navigasi, pilih Masalah.
2. Pilih Masalah aktif untuk membuka menu tarik-turun pengalih tampilan masalah dan pilih Pengaturan.

3. Di Status, pilih ikon plus di samping kategori yang Anda inginkan statusnya.
4. Beri nama status, lalu pilih ikon tanda centang.

> **Note**
>
> Pilih ikon X untuk membatalkan penambahan status.

Status kustom sekarang terlihat di papan Anda dan ditampilkan sebagai opsi saat membuat masalah.

Untuk mengedit status

1. Di panel navigasi, pilih Masalah.
2. Pilih Masalah aktif untuk membuka menu tarik-turun pengalih tampilan masalah dan pilih Pengaturan.
3. Di Status, pilih ikon edit di samping status yang ingin Anda edit atau ubah.
4. Edit status, lalu pilih ikon tanda centang.

Status yang diedit sekarang terlihat di papan Anda.

Untuk memindahkan status

1. Di panel navigasi, pilih Masalah.
2. Pilih ikon elipsis dan pilih Pengaturan.
3. Di Status, pilih status yang ingin Anda pindahkan.
4. Seret dan jatuhkan status di tempat yang Anda inginkan.

> **Note**
>
> Anda hanya dapat memindahkan status dalam kategori yang ditunjuk.

Status sekarang disusun ulang di papan Anda.

Untuk menonaktifkan status

1. Di panel navigasi, pilih Masalah.
2. Pilih Masalah aktif untuk membuka menu tarik-turun pengalih tampilan masalah dan pilih Pengaturan.
3. Di Status, pilih status yang ingin Anda nonaktifkan.
4. Pada status yang ingin Anda nonaktifkan, pilih sakelar pada status. Statusnya sekarang berwarna abu-abu.

> **Note**
>
> Status yang dinonaktifkan muncul di papan tulis sampai semua masalah dipindahkan darinya. Masalah tidak dapat ditambahkan ke status yang dinonaktifkan.

5. Untuk mengaktifkan kembali status yang dinonaktifkan, pilih sakelar pada status. Statusnya tidak lagi berwarna abu-abu.

> **Note**
>
> Setidaknya harus ada satu status aktif di setiap kategori. Jika hanya ada satu status dalam kategori, Anda tidak dapat menonaktifkannya.

Mengkonfigurasi estimasi upaya masalah

Ikuti langkah-langkah ini untuk mengonfigurasi pengaturan estimasi upaya untuk masalah di CodeCatalyst.

Untuk mengonfigurasi estimasi upaya untuk masalah

1. Di panel navigasi, pilih Masalah.
2. Pilih Masalah aktif untuk membuka menu tarik-turun pengalih tampilan masalah dan pilih Pengaturan.
3. Di Estimasi di bagian Pengaturan dasar, pilih bagaimana nilai estimasi akan ditampilkan. Jenis perkiraan yang tersedia adalah ukuran T-shirt, pengurutan Fibonacci, atau perkiraan Sembunyikan. Ketika jenis estimasi diperbarui, tidak ada data yang akan hilang dan nilai estimasi

semua masalah akan dikonversi secara otomatis. Pemetaan konversi ditunjukkan pada tabel berikut.

Ukuran kaos	Urutan Fibonacci
XS	1
XS	2
D	3
M	5
L	8
XL	13

Mengaktifkan atau menonaktifkan beberapa penerima tugas

Ikuti langkah-langkah ini untuk mengonfigurasi pengaturan untuk beberapa penerima tugas untuk masalah di. CodeCatalyst

Untuk mengaktifkan atau menonaktifkan beberapa penerima tugas

1. Di panel navigasi, pilih Masalah.
2. Pilih Masalah aktif untuk membuka menu tarik-turun pengalih tampilan masalah dan pilih Pengaturan.
3. Di Penerima tugas di bagian Pengaturan dasar, alihkan indikator untuk mengaktifkan beberapa penerima tugas yang ditetapkan ke masalah yang sama. Masalah dapat memiliki hingga 10 penerima tugas. Jika Anda tidak mengaktifkan opsi ini, Anda hanya dapat menetapkan satu penerima tugas untuk suatu masalah.

Membuat tampilan masalah

Anda dapat membuat tampilan untuk melihat masalah yang sesuai dengan kumpulan filter tertentu dengan cepat. Ini dapat membantu Anda menghemat waktu dan dengan cepat melihat masalah yang sebelumnya telah Anda filter, dikelompokkan, atau diurutkan berdasarkan.

Untuk membuat tampilan masalah

1. Di panel navigasi, pilih Masalah.
2. (Opsional) Tergantung pada kasus penggunaan Anda, Anda mungkin ingin membuat tampilan dari tampilan yang ada. Untuk menavigasi ke tampilan yang berbeda, pilih Masalah aktif untuk membuka menu tarik-turun pengalih tampilan masalah dan pilih tampilan.
3. (Opsional) Konfigurasikan filter, pengelompokan, dan pengurutan sebelum Anda membuat tampilan. Anda dapat menambahkan ini saat membuat tampilan, tetapi jika Anda melakukannya sebelumnya, Anda dapat melihat pratinjau apa yang ditampilkan dalam tampilan sebelum membuatnya.
4. Buka menu tarik-turun pengalih tampilan masalah dari bilah header. Untuk membuat tampilan papan di mana masalah dilihat di kolom berdasarkan status, pilih + di kolom Papan. Untuk membuat tampilan kisi di mana masalah dilihat dalam daftar, pilih + di kolom Grid. Anda dapat mengubah jenis tampilan sebelum dibuat jika Anda berubah pikiran.
5. Dalam kotak dialog Buat tampilan, masukkan Nama untuk tampilan.
6. Filter, masalah Grup menurut, dan Urutkan masalah menurut bidang diisi berdasarkan pengaturan tampilan saat ini. Perbarui jika perlu.
7. Pilih Buat tampilan untuk membuat tampilan dan beralih ke sana.

Kuota untuk masalah di CodeCatalyst

Tabel berikut menjelaskan kuota dan batasan untuk masalah di Amazon CodeCatalyst. Untuk informasi lebih lanjut tentang kuota di Amazon CodeCatalyst, lihat.. Kuota untuk CodeCatalyst

Sumber Daya	Kuota bawaan
Masalah aktif	Maksimal 1.000 per proyek.
Ukuran lampiran	Maksimum 500MB per lampiran. Total penyimpanan lampiran maksimum dipengaruhi oleh batas penyimpanan keseluruh an untuk ruang Anda. Untuk informasi selengkapnya, silakan lihat Harga .
Jumlah total masalah (aktif dan diarsipkan)	Maksimal 100.000 per proyek.

Sumber Daya	Kuota bawaan
Tampilan tersimpan	Maksimal 50 tampilan masalah tersimpan per proyek.
Jumlah permintaan tarik yang dapat Anda tautkan ke masalah	Maksimal 50 permintaan tarik per masalah.
Status (per proyek)	Maksimal 50 per proyek.
Status (per terbitan)	Maksimal 50 per edisi.
Label (per proyek)	Maksimal 200 per proyek.
Label (per edisi)	Maksimal 50 per edisi.
Bidang kustom (per masalah)	Maksimal 50 per edisi.
Penerima Tugas	Maksimal 10 per edisi.
Komentar	Maksimal 1.000 per edisi.
Tugas	Maksimal 100 per edisi.

Konfigurasikan identitas, izin, dan akses di CodeCatalyst

Saat masuk ke Amazon CodeCatalyst untuk pertama kalinya, Anda membuat AWS Builder ID. AWS ID Builder tidak ada di AWS Identity and Access Management. Nama pengguna yang Anda pilih selama login pertama menjadi ID pengguna unik untuk identitas Anda.

Masuk CodeCatalyst, Anda dapat masuk untuk pertama kalinya dengan salah satu dari dua cara:

- Sebagai bagian dari menciptakan ruang.
- Sebagai bagian dari menerima undangan ke proyek atau ruang di CodeCatalyst.

Peran atau peran yang terkait dengan identitas Anda menentukan tindakan yang dapat Anda lakukan CodeCatalyst. Peran proyek, seperti administrator Proyek dan Kontributor, khusus untuk proyek, sehingga Anda dapat memiliki satu peran dalam satu proyek dan peran yang berbeda dalam proyek lain. Jika Anda membuat spasi, CodeCatalyst secara otomatis menetapkan peran administrator Space. Saat pengguna menerima undangan ke proyek, CodeCatalyst tambahkan identitas tersebut ke ruang dan berikan peran akses terbatas kepada mereka. Ketika Anda mengundang pengguna ke proyek, Anda memilih peran yang Anda ingin mereka miliki dalam proyek, yang menentukan tindakan apa yang dapat dan tidak dapat mereka ambil dalam proyek. Sebagian besar pengguna yang mengerjakan proyek hanya memerlukan peran Kontributor untuk melakukan tugas mereka. Untuk informasi selengkapnya, lihat Memberikan akses dengan peran pengguna.

Selain peran proyek, pengguna dalam proyek memerlukan token akses pribadi (PAT) untuk mengakses repositori sumber untuk proyek saat menggunakan klien Git atau lingkungan pengembangan terintegrasi (IDE). Anggota proyek dapat menggunakan PAT ini dengan aplikasi pihak ketiga sebagai kata sandi khusus aplikasi yang terkait dengan identitas mereka CodeCatalyst . Misalnya, ketika Anda mengkloning repositori sumber ke komputer lokal, Anda harus memberikan PAT serta nama pengguna Anda CodeCatalyst .

Anda dapat mengonfigurasi akses antara CodeCatalyst dan AWS sumber daya dengan menggunakan peran layanan untuk melakukan tindakan seperti mengakses AWS CloudFormation tumpukan dan sumber daya saat Anda menerapkan tindakan dalam alur kerja. Anda harus mengonfigurasi akses antara CodeCatalyst dan AWS sumber daya untuk tindakan alur kerja yang disertakan dengan templat proyek untuk dijalankan.

Topik

- Berikan akses repositori pengguna dengan token akses pribadi
- Mengakses GitHub sumber daya dengan koneksi pribadi
- Konfigurasikan ID AWS Builder Anda untuk masuk dengan otentikasi multi-faktor (MFA)
- Keamanan di Amazon CodeCatalyst
- Memantau peristiwa dan panggilan API menggunakan logging
- Kuota untuk identitas, izin, dan akses di CodeCatalyst
- Pemecahan Masalah

Memberikan akses dengan peran pengguna

Di Amazon CodeCatalyst, Anda dapat menetapkan peran kepada pengguna di tingkat proyek dan tingkat ruang. Dalam sebuah proyek, peran menentukan apa yang pengguna diizinkan untuk melakukan dalam proyek dengan sumber daya untuk proyek itu. Pengguna mendapatkan keanggotaan di ruang ketika mereka bergabung dengan sebuah proyek. Anda dapat menambah atau menghapus pengguna sebagai administrator spasi. Peran administrator Space memiliki izin terluas dari peran apa pun. CodeCatalyst Sebagai praktik terbaik, tetapkan pengguna izin tersempit yang diperlukan untuk melakukan pekerjaan mereka.

Anda dapat menetapkan peran kepada pengguna di ruang. Anda juga dapat menetapkan peran kepada pengguna dalam proyek tempat mereka menjadi anggota. Setiap pengguna hanya dapat memiliki satu peran dalam proyek atau ruang, tetapi pengguna dapat memiliki peran yang berbeda di setiap proyek dan ruang. Misalnya, pengguna mungkin memiliki peran administrator Project dalam satu proyek dan peran Kontributor dalam proyek lain.

Topik

- Memahami peran pengguna untuk ruang dan proyek
- Melihat izin yang tersedia untuk setiap peran
- Melihat dan mengubah peran pengguna

Memahami peran pengguna untuk ruang dan proyek

Ada tiga peran yang tersedia untuk sebuah ruang:

- Administrator ruang

- Pengguna daya
- Akses terbatas

Pengguna yang menerima undangan ke proyek memiliki peran akses terbatas yang secara otomatis ditetapkan kepada mereka di ruang yang berisi proyek.

Ada empat peran yang tersedia untuk anggota dalam sebuah proyek:

- Administrator proyek
- Kontributor
- Pengulas
- Baca saja

Saat Anda menambahkan pengguna ke proyek, CodeCatalyst secara otomatis memberi mereka peran akses terbatas. Jika Anda menghapus pengguna dari semua proyek, CodeCatalyst secara otomatis menghapus peran Akses terbatas dari pengguna tersebut.

Peran administrator ruang

Peran administrator Space adalah peran yang paling kuat dalam CodeCatalyst. Hanya tetapkan peran administrator Space kepada pengguna yang perlu mengelola setiap aspek spasi, karena peran ini memiliki semua izin. CodeCatalyst Pengguna dengan peran administrator Space adalah satu-satunya pengguna yang dapat menambah atau menghapus pengguna lain dari peran administrator Space dan menghapus ruang.

Saat Anda membuat spasi, CodeCatalyst secara otomatis memberi Anda peran administrator Space. Sebagai praktik terbaik, kami menyarankan Anda menambahkan peran ini ke setidaknya satu pengguna lain yang dapat bertindak dalam peran ini jika pembuat ruang asli tidak tersedia.

Peran pengguna daya

Peran pengguna Power adalah peran paling kuat kedua dalam CodeCatalyst ruang, tetapi tidak memiliki akses ke proyek di ruang angkasa. Ini dirancang untuk pengguna yang harus dapat membuat proyek di ruang dan membantu mengelola pengguna dan sumber daya untuk ruang tersebut. Tetapkan peran pengguna Power kepada pengguna yang merupakan pemimpin tim atau manajer yang membutuhkan kemampuan untuk membuat proyek dan mengelola pengguna di ruang sebagai bagian dari pekerjaan mereka.

Peran akses terbatas

Peran akses akses terbatas adalah peran yang akan dimiliki sebagian besar pengguna di CodeCatalyst spasi. Ini adalah peran yang secara otomatis ditetapkan kepada pengguna ketika mereka menerima undangan ke proyek dalam ruang. Ini memberikan izin terbatas yang mereka butuhkan untuk bekerja dalam ruang yang berisi proyek itu. Tetapkan peran akses terbatas ke pengguna yang Anda undang langsung ke ruang kecuali pekerjaan mereka mengharuskan mereka mengelola beberapa aspek ruang.

Peran administrator proyek

Peran administrator Proyek adalah peran yang paling kuat dalam sebuah CodeCatalyst proyek. Hanya tetapkan peran ini kepada pengguna yang perlu mengelola setiap aspek proyek, termasuk mengedit pengaturan proyek, mengelola izin proyek, dan menghapus proyek.

Peran proyek tidak memiliki izin apa pun di tingkat ruang. Oleh karena itu, pengguna dengan peran administrator Proyek tidak dapat membuat proyek tambahan. Hanya pengguna dengan administrator Space atau peran pengguna Power yang dapat membuat proyek.

> **Note**
>
> Peran administrator Space memiliki semua izin di CodeCatalyst.

Peran kontributor

Peran kontributor ditujukan untuk sebagian besar anggota dalam suatu CodeCatalyst proyek. Tetapkan peran ini kepada pengguna yang harus dapat bekerja dengan kode, alur kerja, masalah, dan tindakan dalam proyek.

Peran pengulas

Peran Peninjau ditujukan untuk pengguna yang harus dapat berinteraksi dengan sumber daya dalam proyek, seperti permintaan tarik dan masalah, tetapi tidak membuat dan menggabungkan kode, membuat alur kerja, atau memulai atau menghentikan alur kerja yang berjalan dalam proyek. CodeCatalyst Tetapkan peran Reviewer kepada pengguna yang harus dapat menyetujui dan mengomentari permintaan tarik, membuat, memperbarui, menyelesaikan, dan mengomentari masalah, serta melihat kode dan alur kerja dalam proyek.

Baca saja peran

Peran Read only ditujukan untuk pengguna yang perlu melihat sumber daya dan status sumber daya tetapi tidak berinteraksi dengan mereka atau berkontribusi langsung ke proyek. Pengguna dengan peran ini tidak dapat membuat sumber daya CodeCatalyst, tetapi mereka dapat melihatnya dan menyalinnya, seperti mengkloning repositori dan mengunduh lampiran ke masalah ke komputer lokal. Tetapkan peran Baca saja untuk pengguna yang perlu melihat sumber daya dan status proyek, tetapi tidak berinteraksi langsung dengannya.

Melihat izin yang tersedia untuk setiap peran

Tabel berikut menunjukkan izin yang tersedia untuk setiap CodeCatalyst peran. Gunakan tautan untuk melompat ke set izin yang sesuai.

- Space permissions
- Extensions permissions
- Project permissions
- Source repository permissions
- Dev Environment permissions
- Package repository and package permissions
- Workflow permissions
- Issues permissions
- Custom blueprint permissions
- Notifications permissions
- Search permissions

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Izin ruang							
Buat ruang							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Edit detail penagihan ruang							
Siapkan dan aktifkan sistem masuk tunggal							
Hapus sistem masuk tunggal							
Aktifkan fitur AI generatif untuk suatu ruang							
Nonaktifkan fitur AI generatif untuk suatu ruang							
Hapus spasi							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Menambahkan pengguna lain ke peran administrator Space							
Hapus pengguna lain dari peran administrator Space							
Buat tim							
Hapus tim							
Perbarui tim							
Nonaktifkan sumber daya mesin untuk ruang							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Aktifkan sumber daya mesin untuk ruang							
Buat proyek							
Kaitkan koneksi akun AWS dengan spasi							
Memperbarui koneksi akun AWS							
Putus koneksi akun AWS dari ruang							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Hapus koneksi akun AWS dan hapus dari ruang							
Undang orang lain ke ruang							
Buat koneksi VPC							
Edit koneksi VPC							
Hapus koneksi VPC							
Lihat log aktivitas di ruang							
Lihat koneksi akun AWS							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Lihat insiden untuk CodeCatalyst							
Lihat ruang							
Lihat tim							
Lihat koneksi VPC							

Izin ekstensi	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Instal ekstensi							
Perbarui ekstensi							
Hapus ekstensi							
Connect GitHub akun							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Putuskan sambungan akun GitHub							
Connect situs Jira							
Putuskan sambungan situs Jira							
Lihat detail konfigurasi untuk ekstensi yang diinstal							
Lihat ekstensi							

Izin proyek	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Edit pengaturan proyek							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Nonaktifkan sumber daya mesin untuk proyek							
Aktifkan sumber daya mesin untuk proyek							
Hapus proyek							
Mengundang pengguna ke proyek							
Ubah peran pengguna dalam proyek							
Menghapus pengguna dari proyek							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Menambahkan tim ke proyek							
Hapus tim dari proyek							
Ubah peran proyek tim							
Lihat proyek							
Lihat aktivitas proyek							
Lihat tim dalam proyek							
Lihat cetak biru							

Izin repositori sumber	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Buat repositori							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Tautkan repositori							
Putuskan tautan repositori							
Hapus repositori							
Edit pengaturan repositori							
Lihat repositori							
Lihat pengaturan repositori							
Repositori klon							
Buat cabang							
Buat aturan cabang							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Ubah cabang default							
Hapus cabang							
Menggabungkan cabang							
Perbarui aturan cabang							
Lihat cabang							
Lihat aturan cabang							
Buat folder							
Hapus folder							
Edit folder							
Lihat folder							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Buat file							
Menghapus file							
Mengedit file							
Lihat file							
Buat dan dorong komit							
Lihat komit							
Buat permintaan tarik							
Buat aturan persetujuan untuk permintaan tarik							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Ganti persyaratan penggabungan untuk permintaan tarik							
Perbarui permintaan tarik							
Perbarui aturan persetujuan untuk permintaan tarik							
Tampilkan permintaan pull							
Lihat aturan persetujuan untuk permintaan tarik							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Tutup permintaan tarik							
Menyetujui permintaan tarik							
Komentari permintaan tarik							
Berinteraksi dengan Amazon Q dalam komentar tentang permintaan tarik							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Buat revisi untuk permintaan tarik yang dibuat oleh Amazon Q							
Tautkan masalah ke permintaan tarik							
Putuskan tautan masalah dari permintaan tarik							

Izin Lingkungan Pengembang	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Buat Lingkungan Dev Anda sendiri							
Hentikan Lingkungan Dev Anda sendiri							
Stop Dev Environments yang dibuat oleh pengguna lain							
Lanjutkan Lingkungan Dev Anda sendiri							
Lihat Lingkungan Dev Anda sendiri							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Lihat Lingkungan Dev yang dibuat oleh pengguna lain							
Edit Lingkungan Pengembang Anda sendiri							
Edit Lingkungan Pengembang yang dibuat oleh pengguna lain							
Hapus Lingkungan Dev Anda sendiri							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Hapus Lingkungan Pengembang yang dibuat oleh pengguna lain							
Buat devfile untuk Lingkungan Dev							
Edit devfile untuk Lingkungan Dev							
Hapus devfile untuk Lingkungan Dev							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Lihat devfile untuk Lingkungan Dev							
Repositori paket dan izin paket	**Peran administrator ruang**	**Peran pengguna daya**	**Peran akses terbatas**	**Peran administrator proyek**	**Peran kontributor**	**Peran pengulas**	**Baca peran saja**
Buat repositori paket							
Lihat repositori paket							
Edit repositori paket							
Hapus repositori paket							
Buat repositori paket gateway							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Lihat repositori paket gateway							
Hapus repositori paket gateway							
Tambahkan repositori paket upstream							
Edit urutan pencarian repositori hulu							
Hapus repositori paket upstream							
Connect ke repositori paket							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Baca paket dari repositori paket							
Publikasikan paket ke repositori paket							
Membaca dan menyimpan paket dari repositori upstream							
Lihat paket							
Lihat versi paket							
Lihat aset versi paket							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Daftar dependensi versi paket							
Memperbarui status versi paket							
Perbarui konfigurasi asal paket							
Hapus versi paket							

Izin alur kerja	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Buat alur kerja							
Perbarui alur kerja							
Hapus alur kerja							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Mulai alur kerja							
Hentikan alur kerja							
Buat rahasia alur kerja							
Perbarui rahasia alur kerja							
Hapus rahasia alur kerja							
Ciptakan lingkungan							
Hapus lingkungan							
Buat armada							
Perbarui armada							
Hapus armada							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Mengelola sumber daya alur kerja di akun lain							
Kaitkan koneksi VPC dengan lingkungan							
Putus koneksi VPC dengan lingkungan							
Kaitkan lingkungan yang terhubung dengan VPC dengan alur kerja							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Lepaskan lingkungan yang terhubung dengan VPC dengan alur kerja							
Menyetujui alur kerja berjalan							
Melacak komit dalam alur kerja							
Lihat lingkungan							
Lihat log tindakan build							
Lihat armada							
Lihat log tindakan pengujian							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Lihat alur kerja							
Lihat alur kerja berjalan							
Lihat hasil alur kerja							
Lihat rahasia alur kerja							

Masalah izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Buat masalah							
Perbarui masalah							
Lihat masalah							
Arsipkan masalah							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Tetapkan masalah ke Amazon Q							
Berinteraksi dengan Amazon Q dalam komentar tentang suatu masalah							
Batalkan penetapan Amazon Q dari masalah							
Memperbarui masalah yang dibuat oleh pengguna lain							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Lihat komentar tentang masalah							
Buat komentar tentang suatu masalah							
Perbarui komentar tentang suatu masalah							
Buat label							
Memperbarui label							
Lihat label							
Menambahkan label ke masalah							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Hapus label dari masalah							
Buat status kustom untuk masalah							
Memperbarui status kustom							
Melihat status kustom							
Memindahkan status kustom							
Nonaktifkan status kustom							
Tambahkan lampiran ke masalah							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Lihat lampiran masalah							
Menghapus lampiran dari masalah							
Tautkan permintaan tarik ke masalah							
Putuskan tautan permintaan tarik dari masalah							
Tautkan proyek Jira							
Putuskan tautan proyek Jira							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Izin cetak biru kustom	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Buat proyek cetak biru kustom							
Publikasikan cetak biru kustom pratinjau							
Batalkan publikasi cetak biru kustom pratinjau							
Publikasikan cetak biru khusus							
Batalkan publikasi cetak biru khusus							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Tambahkan cetak biru khusus ke katalog cetak biru ruang							
Hapus cetak biru khusus dari katalog cetak biru ruang							
Mengelola izin penerbitan untuk cetak biru kustom							
Mengelola versi katalog untuk cetak biru kustom							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Perbarui cetak biru khusus							
Menghapus versi cetak biru kustom							
Hapus cetak biru kustom							
Menerapkan cetak biru khusus ke proyek							
Lepaskan cetak biru kustom dari proyek							
Perbarui versi cetak biru kustom yang diterapkan							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Edit alias cetak biru kustom							
Lihat cetak biru kustom yang dipublikasikan							

Izin pemberitahuan	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Konfigurasikan saluran notifikasi							
Hapus saluran notifikasi							
Edit setelan notifikasi							
Lihat setelan notifikasi							

Izin	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Secara otomatis menerima pemberitahuan tentang CodeCatalyst insiden							
Mengonfigurasi notifikasi email untuk akun email terkait							

Izin pencarian	Peran administrator ruang	Peran pengguna daya	Peran akses terbatas	Peran administrator proyek	Peran kontributor	Peran pengulas	Baca peran saja
Cari di dalam proyek							
Cari di seluruh ruang							

Melihat dan mengubah peran pengguna

Anda dapat melihat peran yang ditetapkan ke pengguna. Ini membantu Anda memahami tindakan apa yang dapat mereka ambil dalam sebuah proyek. Anda juga dapat mengubah peran mereka jika mereka membutuhkan izin tambahan.

Untuk melihat peran pengguna dalam proyek

1. Arahkan ke proyek tempat Anda ingin melihat peran yang terkait dengan setiap anggota proyek.

   > **Tip**
   >
   > Anda dapat memilih proyek mana yang akan dilihat di bilah navigasi atas.

2. Di panel navigasi, pilih Pengaturan proyek.
3. Pada tab Anggota, peran untuk setiap anggota proyek ditampilkan di Peran.

Untuk mengubah peran pengguna dalam proyek

1. Arahkan ke proyek tempat Anda ingin mengubah peran yang terkait dengan anggota proyek.

   > **Tip**
   >
   > Anda dapat memilih proyek mana yang akan dilihat di bilah navigasi atas.

2. Di panel navigasi, pilih Pengaturan proyek.
3. Pada tab Anggota, di Anggota proyek, pilih pengguna yang perannya ingin Anda ubah. Pilih Tindakan, lalu pilih Edit peran.
4. Di Peran, pilih peran proyek, lalu pilih Konfirmasi.

Melihat dan mengubah peran dalam ruang

Semua pengguna yang menerima undangan ke proyek CodeCatalyst menjadi anggota ruang proyek. Anda dapat melihat daftar anggota ruang. Anda dapat mengubah peran pengguna dari Akses terbatas ke administrator Space untuk mengelola ruang dan sumber dayanya dengan lebih baik. Peran administrator Space adalah satu-satunya peran yang memungkinkan pengguna membuat proyek CodeCatalyst.

> **Warning**
>
> Peran administrator Space adalah peran yang paling kuat dalam CodeCatalyst. Pengguna dengan peran ini dapat melakukan tindakan apa pun CodeCatalyst, termasuk menghapus ruang. Hanya tetapkan peran ini kepada pengguna yang memerlukan tingkat akses ini ke ruang Anda. Untuk informasi selengkapnya, lihat Peran administrator ruang.

Untuk mengubah peran pengguna di ruang

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke ruang.

   > **Tip**
   >
   > Jika Anda memiliki lebih dari satu spasi, Anda dapat memilih ruang mana yang akan dilihat di bilah navigasi atas.

3. Pilih tab Anggota.
4. Pilih pengguna yang perannya ingin Anda ubah, lalu pilih Ubah peran.
5. Di Ubah peran, pilih peran yang ingin Anda tetapkan, lalu pilih Konfirmasi.

Berikan akses repositori pengguna dengan token akses pribadi

Untuk mengakses beberapa CodeCatalyst sumber daya, seperti repositori sumber, pada komputer lokal dengan klien Git atau lingkungan pengembangan terintegrasi (IDE), Anda harus memasukkan kata sandi khusus aplikasi. Anda dapat membuat token akses pribadi (PAT) untuk digunakan untuk tujuan ini. PAT yang Anda buat dikaitkan dengan identitas pengguna Anda di semua ruang dan proyek. CodeCatalyst Anda dapat membuat lebih dari satu PAT untuk CodeCatalyst identitas Anda.

Anda dapat melihat nama dan tanggal kedaluwarsa PAT yang telah Anda buat, dan Anda dapat menghapus yang tidak lagi Anda butuhkan. Anda hanya dapat menyalin rahasia PAT pada saat Anda membuatnya.

> **Note**
>
> Secara default, PAT kedaluwarsa dalam 1 tahun.

Membuat PATs

PAT dikaitkan dengan identitas pengguna Anda di CodeCatalyst. Anda hanya dapat menyalin rahasia PAT pada saat Anda membuatnya.

Membuat PATs (konsol)

Anda dapat menggunakan konsol untuk membuat PAT di CodeCatalyst.

Untuk membuat token akses pribadi (konsol)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di bilah menu atas, pilih lencana profil Anda, lalu pilih Pengaturan saya. Halaman Pengaturan CodeCatalyst saya terbuka.

   > **Tip**
   >
   > Anda juga dapat menemukan profil pengguna Anda dengan membuka halaman anggota untuk proyek atau ruang dan memilih nama Anda dari daftar anggota.

3. Di bawah Token akses pribadi, pilih Buat.

   Halaman Create PAT ditampilkan.
4. Dalam nama PAT, masukkan nama deskriptif untuk PAT Anda.
5. Di Tanggal kedaluwarsa, pertahankan tanggal default atau pilih ikon kalender untuk memilih tanggal kustom. Tanggal kedaluwarsa default menjadi 1 tahun dari tanggal saat ini.
6. Pilih Buat.

   > **Tip**
   >
   > Anda juga dapat membuat token ini ketika Anda memilih Repositori klon untuk repositori sumber.

7. Untuk menyalin rahasia PAT, pilih Salin. Simpan rahasia PAT di mana Anda akan dapat mengambilnya.

> **Important**
>
> Rahasia PAT hanya ditampilkan sekali. Anda tidak dapat mengambilnya setelah Anda menutup jendela. Jika Anda tidak menyimpan rahasia PAT di lokasi yang aman, Anda dapat membuat yang lain.

.

Membuat PATs (CLI)

Anda dapat menggunakan CLI untuk membuat PAT di. CodeCatalyst

Untuk membuat token akses pribadi (AWS CLI)

1. Di terminal atau baris perintah, jalankan create-access-token perintah sebagai berikut.

   ```
   aws codecatalyst create-access-token
   ```

   Jika berhasil, perintah mengembalikan informasi tentang PAT yang dibuat seperti contoh berikut.

   ```
   {
       "secret": "value",
       "name": "marymajor-22222EXAMPLE",
       "expiresTime": "2024-02-04T01:56:04.402000+00:00"
   }
   ```

2.

Anda hanya dapat melihat rahasia PAT sekali—saat Anda membuat PAT. Jika Anda salah menempatkan rahasia PAT atau Anda khawatir bahwa itu tidak disimpan dengan aman, Anda dapat membuat yang lain.

Anda dapat melihat PAT yang terkait dengan akun pengguna Anda dengan menggunakan. AWS CLI Anda hanya dapat melihat informasi tentang PAT, dan bukan nilai rahasia PAT itu sendiri.

> **Note**
>
> Pastikan Anda menggunakan versi terbaru AWS CLI untuk dikerjakan CodeCatalyst. Versi sebelumnya mungkin tidak berisi CodeCatalyst perintah. Anda harus mengkonfigurasi AWS CLI profil Anda sebelum Anda dapat menggunakannya dengan CodeCatalyst. Untuk informasi selengkapnya, lihat Menyiapkan untuk menggunakan AWS CLI dengan CodeCatalyst.

Melihat PATs

Anda dapat melihat PAT di CodeCatalyst. Daftar ini menunjukkan semua PAT yang telah Anda kaitkan dengan identitas pengguna Anda. PAT Anda dikaitkan dengan profil pengguna Anda di semua ruang dan proyek di CodeCatalyst. PAT yang kedaluwarsa tidak ditampilkan karena dihapus setelah kedaluwarsa.

Melihat PAT (konsol)

Anda dapat menggunakan konsol untuk melihat PAT yang terkait dengan identitas pengguna Anda. CodeCatalyst

Untuk melihat token akses pribadi Anda (konsol)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di bilah menu atas, pilih lencana profil Anda, lalu pilih Pengaturan saya. Halaman Pengaturan CodeCatalyst saya terbuka.

   > **Tip**
   >
   > Anda juga dapat menemukan profil pengguna Anda dengan membuka halaman anggota untuk proyek atau ruang dan memilih nama Anda dari daftar anggota.

3. Di bawah Token akses pribadi, lihat nama dan tanggal kedaluwarsa PAT Anda saat ini.

Melihat PAT (CLI)

Anda dapat menggunakan CLI untuk melihat PAT yang terkait dengan identitas pengguna Anda di. CodeCatalyst

Untuk melihat token akses pribadi Anda (AWS CLI)

- Di terminal atau baris perintah, jalankan list-access-tokens perintah sebagai berikut.

```
aws codecatalyst list-access-tokens
```

Jika berhasil, perintah mengembalikan informasi tentang PAT yang terkait dengan akun pengguna Anda seperti contoh berikut.

```
{
    "items": [
        {
            "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
            "name": "marymajor-22222EXAMPLE",
            "expiresTime": "2024-02-04T01:56:04.402000+00:00"
        },
        {
            "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLEbbbbb",
            "name": "marymajor-11111EXAMPLE",
            "expiresTime": "2023-03-12T01:58:40.694000+00:00"
        }
    ]
}
```

Menghapus PATs

Anda dapat menghapus PAT yang terkait dengan identitas pengguna Anda di CodeCatalyst.

Menghapus PATs (konsol)

Anda dapat menggunakan konsol untuk menghapus PAT di CodeCatalyst.

Untuk menghapus token akses pribadi (konsol)

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di bilah menu atas, pilih lencana profil Anda, lalu pilih Pengaturan saya. Halaman Pengaturan CodeCatalyst saya terbuka.

> **Tip**
>
> Anda juga dapat menemukan profil pengguna Anda dengan membuka halaman anggota untuk proyek atau ruang dan memilih nama Anda dari daftar anggota.

3. Di bawah Token akses pribadi, pilih pemilih di sebelah PAT yang ingin Anda hapus, lalu pilih Hapus.

   Pada Hapus PAT:? <name> halaman, untuk mengonfirmasi penghapusan, ketik hapus di bidang teks. Pilih Hapus.

Menghapus PATs (CLI)

Anda dapat menghapus PAT yang terkait dengan identitas pengguna Anda dengan menggunakan AWS CLI. Untuk melakukan ini, Anda harus menyediakan ID untuk PAT, yang dapat Anda lihat dengan menggunakan delete-access-token perintah.

> **Note**
>
> Pastikan Anda menggunakan versi terbaru AWS CLI untuk dikerjakan CodeCatalyst. Versi sebelumnya mungkin tidak berisi CodeCatalyst perintah. Untuk informasi lebih lanjut tentang menggunakan AWS CLI with CodeCatalyst, lihatMenyiapkan untuk menggunakan AWS CLI dengan CodeCatalyst.

Untuk menghapus token akses pribadi (AWS CLI)

- Di terminal atau baris perintah, jalankan delete-access-token perintah, berikan ID untuk PAT yang ingin Anda hapus. Misalnya, jalankan perintah berikut untuk menghapus PAT dengan ID *123EXAMPLE*.

  ```
  aws codecatalyst delete-access-token --id a1b2c3d4-5678-90ab-cdef-EXAMPLEbbbbb
  ```

  Jika berhasil, perintah ini tidak mengembalikan respon.

Mengakses GitHub sumber daya dengan koneksi pribadi

Anda dapat menggunakan koneksi pribadi untuk mengotorisasi dan menghubungkan GitHub sumber daya pihak ketiga Anda. CodeCatalyst Misalnya, gunakan koneksi pribadi CodeCatalyst untuk mengotorisasi akses GitHub akun Anda dan membuat repositori sebagai sumber untuk proyek atau cetak biru Anda. Koneksi dipetakan ke CodeCatalyst identitas Anda dan dapat digunakan untuk terhubung ke satu atau lebih repositori sumber. Koneksi yang Anda buat dikaitkan dengan identitas pengguna Anda di semua ruang dan proyek di CodeCatalyst.

> **Note**
>
> Anda dapat mengelola koneksi pribadi dengan cetak biru di GitHub organisasi tempat Anda memiliki akses untuk melakukannya.

Anda dapat membuat satu koneksi pribadi untuk satu identitas pengguna (CodeCatalyst alias) di semua spasi, per jenis penyedia.

Anda dapat menggunakan koneksi pribadi Anda CodeCatalyst untuk membuat GitHub repositori untuk proyek, memilih repositori GitHub sumber untuk cetak biru, dan mengelola permintaan tarik untuk repositori Anda. CodeCatalyst GitHub

> **Note**
>
> Penggunaan koneksi pribadi untuk mengaitkan cetak biru dengan GitHub repositori tidak sama dengan penggunaan ekstensi untuk menautkan repositori. CodeCatalyst GitHub Untuk informasi selengkapnya tentang ekstensi, lihatTambahkan fungsionalitas ke proyek dengan ekstensi di CodeCatalyst.

Membuat koneksi pribadi

Anda dapat menggunakan konsol untuk membuat koneksi pribadi yang terkait dengan identitas pengguna Anda di CodeCatalyst.

Untuk membuat koneksi pribadi

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.

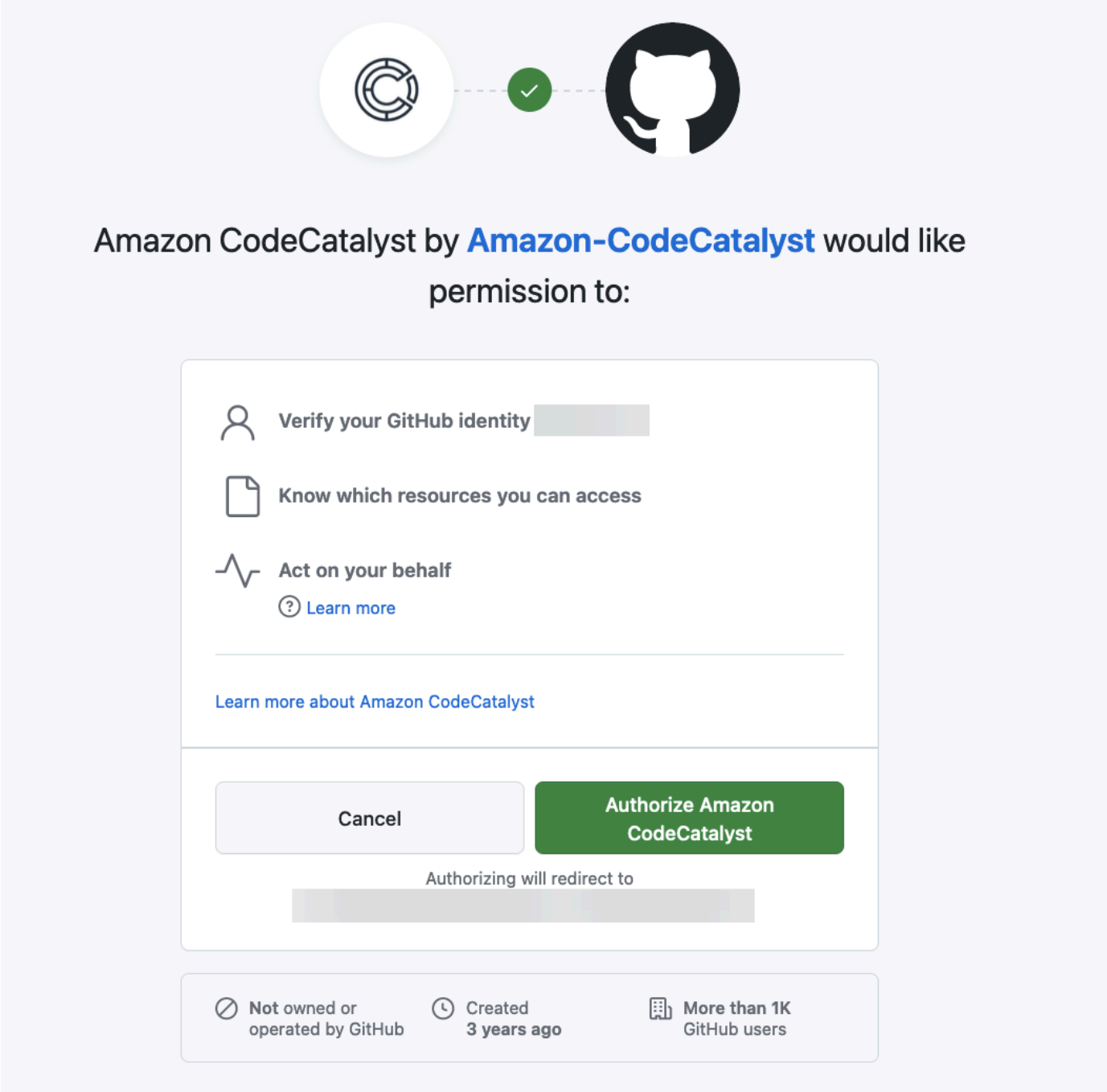
2. Di bilah menu atas, pilih lencana profil Anda, lalu pilih Pengaturan saya. Halaman Pengaturan CodeCatalyst saya terbuka.

   > **Tip**
   >
   > Anda juga dapat menemukan profil pengguna Anda dengan membuka halaman anggota untuk proyek atau ruang dan memilih nama Anda dari daftar anggota.

3. Di bawah Koneksi pribadi, pilih Buat.

   Halaman Buat koneksi ditampilkan.

4. Pilih Buat. Sebuah halaman Buat koneksi ditampilkan.
5. Pada halaman Buat koneksi, di Penyedia, pilih GitHub. Di nama Connection, ketikkan nama untuk koneksi Anda. Pilih Buat.
6. Jika diminta, masuk ke GitHub akun Anda.
7. Pada halaman konfirmasi koneksi, pilih Terima.
8. Pada halaman konfirmasi instalasi, pilih tombol otorisasi untuk mengonfirmasi bahwa Anda ingin menginstal aplikasi konektor.



Menghapus koneksi pribadi

Anda dapat menghapus koneksi pribadi yang terkait dengan identitas pengguna Anda di CodeCatalyst.

> **Note**
>
> Menghapus koneksi pribadi di CodeCatalyst tidak menghapus aplikasi di akun Anda GitHub . Jika Anda membuat koneksi pribadi baru, instalasi aplikasi dapat digunakan. Untuk menghapus instalasi aplikasi GitHub, Anda dapat mencabut aplikasi dan menginstal ulang di lain waktu.

Untuk menghapus koneksi pribadi di CodeCatalyst

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di bilah menu atas, pilih lencana profil Anda, lalu pilih Pengaturan saya. Halaman Pengaturan CodeCatalyst saya terbuka.

   > **Tip**
   >
   > Anda juga dapat menemukan profil pengguna Anda dengan membuka halaman anggota untuk proyek atau ruang dan memilih nama Anda dari daftar anggota.

3. Di bawah Koneksi pribadi, pilih pemilih di sebelah koneksi yang ingin Anda hapus, lalu pilih Hapus.

   Pada koneksi Hapus:? <name> halaman, untuk mengonfirmasi penghapusan, ketik hapus di bidang teks. Pilih Hapus.

1. Masuk ke GitHub dan navigasikan ke pengaturan akun Anda untuk aplikasi yang diinstal. Pilih ikon profil Anda, pilih Pengaturan, lalu pilih Aplikasi.
2. Pada tab Authorized GitHub Apps, dalam daftar aplikasi resmi, lihat aplikasi yang diinstal CodeCatalyst. Untuk mencabut instalasi, pilih Cabut.

Konfigurasikan ID AWS Builder Anda untuk masuk dengan otentikasi multi-faktor (MFA)

Baik Anda membuat profil AWS Builder ID untuk penggunaan pribadi atau penggunaan profesional, kami mendorong untuk mengonfigurasi otentikasi multi-faktor (MFA) sebagai lapisan keamanan lainnya. Kami secara khusus merekomendasikan mengonfigurasi MFA jika Anda anggota ruang dan

berkolaborasi dengan orang lain dalam proyek. Karena lebih dari satu orang dapat memiliki akses ke proyek, ada lebih banyak peluang untuk pelanggaran keamanan.

Saat Anda mengaktifkan MFA, Anda harus masuk ke Amazon CodeCatalyst dengan email dan kata sandi Anda. Bagian masuk ini adalah faktor pertama, di mana Anda menggunakan sesuatu yang Anda ketahui. Anda kemudian masuk dengan kode atau kunci keamanan. Ini adalah faktor kedua, yang merupakan sesuatu yang Anda miliki. Faktor kedua bisa berupa kode otentikasi yang dihasilkan baik oleh perangkat seluler Anda atau dengan mengetuk atau menekan tombol keamanan yang terhubung ke komputer Anda. Secara keseluruhan, beberapa faktor ini memberikan peningkatan keamanan dengan mencegah akses yang tidak sah.

Cara mendaftarkan perangkat untuk digunakan dengan otentikasi multi-faktor

Gunakan prosedur berikut di Profil saya> Autentikasi multi-faktor untuk mendaftarkan perangkat baru Anda untuk otentikasi multi-faktor (MFA).

> **Note**
>
> Kami menyarankan Anda mengunduh aplikasi autentikator yang sesuai terlebih dahulu ke perangkat Anda sebelum memulai langkah-langkah dalam prosedur ini. Untuk daftar aplikasi yang dapat Anda gunakan untuk perangkat MFA, lihat. Aplikasi Authenticator

Untuk mendaftarkan perangkat Anda untuk digunakan dengan MFA

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di kanan atas, pilih panah di sebelah ikon dengan inisiasi pertama Anda, lalu pilih Profil pengguna. Halaman CodeCatalyst Profil terbuka.
3. Pada halaman profil, pilih Kelola profil dan keamanan. Halaman profil AWS Builder ID terbuka.
4. Di sisi kiri halaman, pilih Keamanan.
5. Pada halaman otentikasi multi-faktor, pilih Daftarkan perangkat.
6. Pada halaman Daftarkan perangkat MFA, pilih salah satu jenis perangkat MFA berikut, dan ikuti petunjuknya:

   - Kunci keamanan atau Autentikator bawaan

1. Pada halaman Daftarkan kunci keamanan pengguna Anda, ikuti instruksi yang diberikan kepada Anda oleh browser atau platform Anda.

> **Note**
>
> Pengalaman ini bervariasi berdasarkan sistem operasi dan browser Anda, jadi ikuti instruksi yang ditampilkan oleh browser atau platform Anda. Setelah perangkat Anda berhasil didaftarkan, Anda akan diberikan opsi untuk mengaitkan nama tampilan yang ramah ke perangkat Anda yang baru terdaftar. Jika Anda ingin mengubah ini, pilih Ganti nama, masukkan nama baru, lalu pilih Simpan.

- Aplikasi Authenticator

1. Pada halaman Mengatur aplikasi autentikator, Anda mungkin melihat informasi konfigurasi untuk perangkat MFA baru, termasuk grafik kode QR. Grafik adalah representasi dari kunci rahasia yang tersedia untuk entri manual pada perangkat yang tidak mendukung kode QR.
2. Menggunakan perangkat MFA fisik, lakukan hal berikut:
   a. Buka aplikasi autentikator MFA yang kompatibel. Untuk daftar aplikasi teruji yang dapat Anda gunakan dengan perangkat MFA, lihat. Aplikasi autentikator yang diuji Jika aplikasi MFA mendukung beberapa perangkat, pilih opsi untuk membuat perangkat MFA baru.
   b. Tentukan apakah aplikasi MFA mendukung kode QR, lalu lakukan salah satu hal berikut di halaman Siapkan aplikasi autentikator:
      i. Pilih Tampilkan kode QR, lalu gunakan aplikasi untuk memindai kode QR. Misalnya, Anda dapat memilih ikon kamera atau memilih opsi yang mirip dengan kode Pindai. Kemudian gunakan kamera perangkat untuk memindai kode.
      ii. Pilih tampilkan kunci rahasia, lalu masukkan kunci rahasia itu ke aplikasi MFA Anda.

> **Important**
>
> Saat Anda mengonfigurasi perangkat MFA untuk AWS Builder ID, simpan salinan kode QR atau kunci rahasia di tempat yang aman. Ini dapat membantu jika Anda kehilangan ponsel atau harus menginstal ulang aplikasi otentikator MFA. Jika salah satu dari hal-hal itu terjadi, Anda dapat dengan cepat mengkonfigurasi ulang aplikasi untuk menggunakan konfigurasi MFA yang sama.

3. Pada halaman Siapkan aplikasi autentikator, di bawah kode Authenticator, masukkan kata sandi satu kali yang saat ini muncul di perangkat MFA fisik.

> **Important**
>
> Kirim permintaan Anda segera setelah membuat kode. Jika Anda membuat kode dan kemudian menunggu terlalu lama untuk mengirimkan permintaan, perangkat MFA berhasil dikaitkan dengan profil AWS Builder ID Anda, tetapi perangkat MFA tidak sinkron. Hal ini terjadi karena kata sandi sekali pakai berbasis waktu (TOTP) kedaluwarsa setelah periode waktu yang singkat. Jika ini terjadi, Anda dapat menyinkronisasi ulang perangkat.

4. Pilih Tugaskan MFA. Perangkat MFA sekarang dapat mulai menghasilkan kata sandi satu kali dan sekarang siap digunakan.

Aplikasi Authenticator

Aplikasi Authenticator adalah one-time password (OTP) — based third party-authenticator. Pengguna dapat menggunakan aplikasi autentikator yang diinstal pada perangkat seluler atau tablet mereka sebagai perangkat MFA resmi. Aplikasi autentikator pihak ketiga harus sesuai dengan RFC 6238, yang merupakan algoritma TOTP (kata sandi satu kali berbasis waktu) berbasis standar yang mampu menghasilkan kode otentikasi enam digit.

Saat diminta untuk MFA, pengguna harus memasukkan kode yang valid dari aplikasi autentikator mereka di dalam kotak input yang disajikan. Setiap perangkat MFA yang ditetapkan ke pengguna harus unik. Dua aplikasi autentikator dapat didaftarkan untuk setiap pengguna tertentu.

Aplikasi autentikator yang diuji

Meskipun aplikasi yang sesuai dengan TOTP akan bekerja dengan IAM Identity Center MFA, tabel berikut mencantumkan aplikasi otentikator pihak ketiga yang terkenal untuk dipilih.

Sistem operasi	Aplikasi autentikator yang diuji
Android	Authy, Duo Seluler, Authenticator, Microsoft LastPass Authenticator, Google Authenticator

Sistem operasi	Aplikasi autentikator yang diuji
iOS	Authy, Duo Seluler, Authenticator, Microsoft LastPass Authenticator, Google Authenticator

Mengubah perangkat MFA Anda

Setelah Anda mendaftarkan perangkat MFA, Anda dapat mengubah namanya atau menghapusnya. Kami menyarankan untuk selalu mengaktifkan setidaknya satu perangkat MFA untuk lapisan keamanan tambahan. Anda dapat memiliki hingga lima perangkat terdaftar. Untuk mengetahui cara menambahkan lebih banyak, lihatCara mendaftarkan perangkat untuk digunakan dengan otentikasi multi-faktor.

Mengganti nama perangkat MFA

Untuk mengganti nama perangkat MFA Anda

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di kanan atas, pilih panah di sebelah ikon dengan inisiasi pertama Anda, lalu pilih Profil pengguna. Halaman CodeCatalyst Profil terbuka.
3. Pada halaman profil, pilih Kelola profil dan keamanan. Halaman profil AWS Builder ID terbuka.
4. Pilih Autentikasi multi-faktor di sisi kiri halaman. Anda akan melihat bahwa Ganti nama berwarna abu-abu ketika Anda tiba di halaman.
5. Pilih perangkat MFA yang ingin Anda ubah. Pilih Ganti Nama. Kemudian modal muncul.
6. Pada prompt yang terbuka, masukkan nama baru di nama perangkat MFA, lalu pilih Ganti nama. Perangkat yang diganti namanya muncul di bawah Perangkat otentikasi multi-faktor (MFA).

Menghapus perangkat MFA

Untuk menghapus perangkat MFA

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di kanan atas, pilih panah di sebelah ikon dengan inisiasi pertama Anda, lalu pilih Profil pengguna. Halaman CodeCatalyst Profil terbuka.
3. Pada halaman profil, pilih Kelola profil dan keamanan. Halaman profil AWS Builder ID terbuka.

4. Pilih Autentikasi multi-faktor di sisi kiri halaman. Anda akan melihat bahwa Delete berwarna abu-abu ketika Anda tiba di halaman.
5. Pilih perangkat MFA yang ingin Anda ubah. Pilih Hapus. Muncul modal yang mengatakan Hapus perangkat MFA? . Ikuti petunjuk untuk menghapus perangkat Anda.
6. Pilih Hapus. Perangkat yang dihapus tidak lagi muncul di bawah Perangkat otentikasi multi-faktor (MFA).

Keamanan di Amazon CodeCatalyst

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan ruang yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. Model tanggung jawab bersama menjelaskan hal ini sebagai keamanan cloud dan keamanan dalam cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari Program AWS Kepatuhan Program AWS Kepatuhan . Untuk mempelajari tentang program kepatuhan yang berlaku CodeCatalyst, lihat AWS Layanan dalam Lingkup oleh AWS Layanan Program Kepatuhan .
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, yang mencakup kepekaan data Anda, persyaratan perusahaan, serta peraturan perundangan yang berlaku

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan Amazon CodeCatalyst. Ini menunjukkan kepada Anda cara mengonfigurasi CodeCatalyst untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga belajar cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan CodeCatalyst sumber daya Anda.

Konten

- Ketahanan di Amazon CodeCatalyst
- Keamanan Infrastruktur di Amazon CodeCatalyst
- Analisis konfigurasi dan kerentanan di Amazon CodeCatalyst
- Data dan privasi Anda di Amazon CodeCatalyst
- Praktik terbaik untuk tindakan alur kerja di Amazon CodeCatalyst
- Memahami model CodeCatalyst kepercayaan

Perlindungan data di Amazon CodeCatalyst

Keamanan dan Kepatuhan adalah tanggung jawab bersama antara Amazon CodeCatalyst dan pelanggan, sama seperti model tanggung jawab AWS bersama model berlaku untuk penggunaan AWS sumber daya yang digunakan dalam alur kerja. Seperti yang dijelaskan dalam model CodeCatalyst ini, bertanggung jawab untuk melindungi infrastruktur global untuk layanan. Anda harus bertanggung jawab untuk memelihara kendali terhadap konten yang di-hosting pada infrastruktur ini. Model tanggung jawab bersama ini berlaku untuk perlindungan data di CodeCatalyst.

Untuk tujuan perlindungan data, kami menyarankan Anda untuk melindungi kredensyal akun Anda, dan menyiapkan autentikasi multi-faktor saat masuk. Untuk informasi selengkapnya, lihat Konfigurasikan ID AWS Builder Anda untuk masuk dengan otentikasi multi-faktor (MFA).

Jangan masukkan informasi rahasia atau sensitif, seperti alamat email pelanggan Anda, dalam tag atau kolom bentuk bebas seperti bidang Nama. Ini termasuk nama sumber daya dan pengidentifikasi lain yang Anda masukkan CodeCatalyst selain yang terhubungAkun AWS. Misalnya, jangan memasukkan informasi rahasia atau sensitif sebagai bagian dari nama armada ruang, proyek, atau penyebaran. Data apa pun yang Anda masukkan dalam tag, nama, atau bidang bentuk bebas yang digunakan untuk nama dapat digunakan untuk penagihan atau log diagnostik atau dapat disertakan dalam jalur URL. Ini berlaku untuk menggunakan konsol, APIAWS CLI, CodeCatalyst Action Development Kit, atau AWS SDK apa pun.

Jika Anda memberikan URL ke server eksternal, kami sangat menyarankan agar Anda tidak menyertakan informasi kredensyal keamanan apa pun di URL untuk memvalidasi permintaan Anda ke server tersebut.

CodeCatalyst repositori sumber secara otomatis dienkripsi saat istirahat. Tidak diperlukan tindakan pelanggan. CodeCatalyst juga mengenkripsi data repositori dalam perjalanan menggunakan protokol HTTPS.

CodeCatalyst mendukung MFA. Untuk informasi selengkapnya, lihat Konfigurasikan ID AWS Builder Anda untuk masuk dengan otentikasi multi-faktor (MFA).

Enkripsi data

CodeCatalyst menyimpan dan mentransfer data dengan aman dalam layanan. Semua data dienkripsi dalam perjalanan dan saat istirahat. Setiap data yang dibuat atau disimpan oleh layanan, termasuk metadata apa pun untuk layanan, disimpan secara native dalam layanan dan dienkripsi.

> **Note**
>
> Meskipun informasi tentang masalah disimpan dengan aman di dalam layanan, informasi tentang masalah terbuka juga disimpan di cache lokal browser tempat Anda melihat papan masalah, backlog, dan masalah individual. Untuk keamanan optimal, pastikan untuk menghapus cache browser Anda untuk menghapus informasi ini.

Jika Anda menggunakan sumber daya yang ditautkan CodeCatalyst, seperti koneksi akun ke Akun AWS atau repositori tertaut di GitHub, data dalam perjalanan dari CodeCatalyst sumber daya tertaut tersebut dienkripsi, tetapi penanganan data dalam sumber daya tertaut tersebut dikelola oleh layanan tertaut tersebut. Untuk informasi selengkapnya, lihat dokumentasi untuk layanan tertaut danPraktik terbaik untuk tindakan alur kerja di Amazon CodeCatalyst.

Manajemen kunci

CodeCatalyst tidak mendukung manajemen kunci.

Privasi lalu lintas antar jaringan

Saat Anda membuat spasi CodeCatalyst, Anda memilih Wilayah AWS tempat data dan sumber daya akan disimpan untuk ruang itu. Data proyek dan metadata tidak pernah meninggalkan itu. Wilayah AWS Namun, untuk mendukung navigasi di dalam CodeCatalyst, satu set terbatas ruang, proyek, dan metadata pengguna direplikasi Wilayah AWS di semua partisi. Itu tidak akan direplikasi ke Wilayah AWS luar partisi itu. Misalnya, jika Anda memilih US West (Oregon) sebagai Wilayah AWS saat Anda membuat ruang, data Anda tidak akan direplikasi ke Wilayah di Wilayah China atau. AWS GovCloud (US) Untuk informasi selengkapnya, lihat Mengelola Wilayah AWS, Infrastruktur AWS Global, dan titik akhir AWS layanan.

Data yang direplikasi di Wilayah AWS dalam partisi meliputi:

- Nilai hash terenkripsi yang mewakili nama spasi untuk memastikan keunikan nama spasi. Nilai ini tidak dapat dibaca manusia dan tidak mengekspos nama spasi yang sebenarnya
- ID unik dari ruang
- Metadata untuk ruang yang membantu navigasi lintas ruang
- Di Wilayah AWS mana ruang itu berada
- ID unik dari semua proyek di ruang angkasa
- ID peran yang menunjukkan peran pengguna dalam spasi atau proyek
- Saat mendaftar CodeCatalyst, data dan metadata tentang proses pendaftaran, termasuk:
  - ID unik dari ID AWS Builder
  - Nama tampilan untuk pengguna di ID AWS Builder
  - Alias pengguna di ID AWS Builder
  - Alamat email yang digunakan saat pengguna mendaftar ID AWS Builder
  - Kemajuan proses pendaftaran
  - Jika membuat spasi sebagai bagian dari proses pendaftaran, Akun AWS ID yang digunakan sebagai akun penagihan untuk ruang

Nama-nama luar angkasa unik di seberang CodeCatalyst. Pastikan untuk tidak memasukkan data sensitif dalam nama spasi.

Saat bekerja dengan sumber daya tertaut dan akun yang terhubung seperti koneksi ke Akun AWS atau GitHub repositori, kami sarankan untuk mengonfigurasi lokasi sumber dan tujuan Anda dengan tingkat keamanan tertinggi yang didukung masing-masing. CodeCatalyst mengamankan koneksi antaraAkun AWS,Wilayah AWS, dan Availability Zones dengan menggunakan Transport Layer Security (TLS) 1.2.

Identity and Access Management dan Amazon CodeCatalyst

Di Amazon CodeCatalyst, Anda membuat dan menggunakan AWS Builder ID untuk masuk dan mengakses spasi dan proyek Anda. ID AWS Builder bukan identitas di AWS Identity and Access Management (IAM) dan tidak ada di file Akun AWS. Namun, CodeCatalyst apakah terintegrasi dengan IAM saat memverifikasi ruang untuk tujuan penagihan, dan ketika terhubung ke Akun AWS untuk membuat dan menggunakan sumber daya di dalamnya. Akun AWS

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. Administrator IAM mengontrol siapa yang

dapat diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk menggunakan sumber daya. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Saat Anda membuat spasi di Amazon CodeCatalyst, Anda harus menghubungkan Akun AWS sebagai akun penagihan untuk ruang Anda. Anda harus memiliki izin administrator di Akun AWS untuk memverifikasi CodeCatalyst ruang, atau memiliki izin. Anda juga memiliki opsi untuk menambahkan peran IAM untuk ruang Anda yang CodeCatalyst dapat digunakan untuk membuat dan mengakses sumber daya yang terhubung Akun AWS. Ini disebut peran layanan. Anda dapat memilih untuk membuat koneksi ke lebih dari satu Akun AWS dan membuat peran layanan untuk CodeCatalyst masing-masing akun tersebut.

> **Note**
>
> Penagihan CodeCatalyst berlangsung di akun yang Akun AWS ditunjuk sebagai akun penagihan. Namun, jika Anda membuat peran CodeCatalyst layanan di dalamnya Akun AWS atau dalam koneksi lainnya Akun AWS, sumber daya yang dibuat dan digunakan oleh peran CodeCatalyst layanan akan ditagih dalam koneksi Akun AWS tersebut. Untuk informasi selengkapnya, lihat Mengelola penagihan di Panduan CodeCatalyst Administrator Amazon.

Topik

Kebijakan berbasis identitas di IAM

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke identitas. Identitas itu bisa berupa pengguna, sekelompok pengguna, atau peran. Kebijakan ini

mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan secara spesifik apakah tindakan dan sumber daya diizinkan atau ditolak, serta kondisi yang menjadi dasar dikabulkan atau ditolaknya tindakan tersebut. Anda tidak dapat menentukan secara spesifik prinsipal dalam sebuah kebijakan berbasis identitas karena prinsipal berlaku bagi pengguna atau peran yang melekat kepadanya. Untuk mempelajari semua elemen yang dapat Anda gunakan dalam kebijakan JSON, lihat Referensi elemen kebijakan JSON IAM dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk CodeCatalyst

Untuk melihat contoh kebijakan CodeCatalyst berbasis identitas, lihat. Contoh kebijakan berbasis identitas untuk koneksi CodeCatalyst

Tindakan kebijakan di IAM

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, prinsipal mana yang dapat melakukan tindakan apa pada sumber daya apa, dan dalam kondisi apa.

Elemen `Action` dari kebijakan JSON menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam sebuah kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan operasi AWS API terkait. Ada beberapa pengecualian, misalnya tindakan hanya izin yang tidak memiliki operasi API yang cocok. Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam suatu kebijakan. Tindakan tambahan ini disebut tindakan dependen.

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan tersebut dengan koma.

```
"Action": [
      "prefix:action1",
      "prefix:action2"
         ]
```

Sumber daya kebijakan di IAM

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, prinsipal mana yang dapat melakukan tindakan apa pada sumber daya apa, dan dalam kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek yang menjadi target penerapan tindakan. Pernyataan harus menyertakan elemen `Resource` atau `NotResource`. Praktik terbaiknya, tentukan sumber daya menggunakan Amazon Resource Name (ARN). Anda dapat melakukan ini untuk tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, misalnya operasi pencantuman, gunakan wildcard (*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

```
"Resource": "*"
```

Kunci kondisi kebijakan di IAM

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, prinsipal mana yang dapat melakukan tindakan apa pada sumber daya apa, dan dalam kondisi apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan kondisi yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan operator kondisi, misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa elemen `Condition` dalam sebuah pernyataan, atau beberapa kunci dalam elemen `Condition` tunggal, maka AWS akan mengevaluasinya menggunakan operasi `AND` logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS akan mengevaluasi kondisi tersebut menggunakan operasi `OR` logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Untuk informasi selengkapnya, lihat Elemen kebijakan IAM: variabel dan tag dalam Panduan Pengguna IAM.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua kunci kondisi global AWS , lihat Kunci konteks kondisi global AWS dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk koneksi CodeCatalyst

Di CodeCatalyst, Akun AWS diperlukan untuk mengelola penagihan untuk ruang dan untuk mengakses sumber daya dalam alur kerja proyek. Koneksi akun digunakan untuk mengotorisasi penambahan Akun AWS spasi. Kebijakan berbasis identitas digunakan dalam koneksi. Akun AWS

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau memodifikasi CodeCatalyst sumber daya. Mereka juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS API. Administrator IAM harus membuat kebijakan IAM yang memberikan izin kepada para pengguna dan peran untuk melakukan tindakan di sumber daya yang mereka perlukan. Administrator kemudian harus melampirkan kebijakan tersebut untuk pengguna yang membutuhkannya.

Contoh berikut kebijakan IAM memberikan izin untuk tindakan yang terkait dengan koneksi akun. Gunakan mereka untuk membatasi akses untuk menghubungkan akun ke CodeCatalyst.

Contoh 1: Izinkan pengguna untuk menerima permintaan koneksi dalam satu Wilayah AWS

Kebijakan izin berikut hanya memungkinkan pengguna untuk melihat dan menerima permintaan untuk koneksi antara CodeCatalyst dan Akun AWS. Selain itu, kebijakan menggunakan kondisi untuk hanya mengizinkan tindakan di Wilayah us-west-2 dan bukan dari yang lain. Wilayah AWS Untuk melihat dan menyetujui permintaan, pengguna masuk ke akun AWS Management Console dengan akun yang sama seperti yang ditentukan dalam permintaan.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecatalyst:AcceptConnection",
        "codecatalyst:GetPendingConnection"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestedRegion": "us-west-2"
        }
      }
    }
  ]
```

```
}
```

Contoh 2: Izinkan mengelola koneksi di konsol untuk satu Wilayah AWS

Kebijakan izin berikut memungkinkan pengguna mengelola koneksi antara CodeCatalyst dan Akun AWS di satu Wilayah. Kebijakan menggunakan kondisi untuk hanya mengizinkan tindakan di Wilayah us-west-2 dan bukan dari yang lain. Wilayah AWS Setelah Anda membuat koneksi, Anda dapat membuat CodeCatalystWorkflowDevelopmentRole-***spaceName***peran dengan memilih opsi di AWS Management Console. Dalam kebijakan contoh, kondisi untuk `iam:PassRole` tindakan mencakup prinsip layanan untuk. CodeCatalyst Hanya peran dengan akses itu yang akan dibuat di AWS Management Console.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codecatalyst:*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestedRegion": "us-west-2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateRole",
                "iam:CreatePolicy",
                "iam:AttachRolePolicy",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
```

```
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "codecatalyst.amazonaws.com",
                        "codecatalyst-runner.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

Contoh 3: Tolak mengelola koneksi

Kebijakan izin berikut menyangkal kemampuan pengguna untuk mengelola koneksi antara CodeCatalyst dan. Akun AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "codecatalyst:*"
            ],
            "Resource": "*"
        }
    ]
}
```

Menggunakan tag untuk mengontrol akses ke sumber daya koneksi akun

Tag dapat dilampirkan ke sumber daya atau diteruskan dalam permintaan ke layanan yang mendukung penandaan. Sumber daya dalam kebijakan dapat memiliki tag, dan beberapa tindakan dalam kebijakan dapat menyertakan tag. Tombol kondisi penandaan termasuk kunci `aws:ResourceTag` kondisi `aws:RequestTag` dan. Saat membuat kebijakan IAM, Anda dapat menggunakan kunci syarat tanda berikut:

- Pengguna mana yang dapat melakukan tindakan pada sumber daya koneksi, berdasarkan tag yang sudah dimilikinya.

- Tanda apa yang dapat diteruskan dalam permintaan tindakan.
- Apakah kunci tanda tertentu dapat digunakan dalam permintaan.

Contoh berikut menunjukkan cara menentukan kondisi tag dalam kebijakan untuk pengguna koneksi CodeCatalyst akun. Untuk informasi lebih lanjut tentang kunci syarat ini, lihat Kunci kondisi kebijakan di IAM.

Contoh 1: Izinkan tindakan berdasarkan tag dalam permintaan

Kebijakan berikut memberikan izin kepada pengguna untuk menyetujui koneksi akun.

Untuk melakukan itu, memungkinkan tindakan `AcceptConnection` dan `TagResource` jika permintaan menentukan tag bernama `Project` dengan nilai `ProjectA`. (Kunci syarat `aws:RequestTag` digunakan untuk mengontrol tanda yang dapat dilewatkan dalam permintaan IAM.) Syarat `aws:TagKeys` memastikan kunci tanda peka huruf besar dan kecil.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecatalyst:AcceptConnection",
        "codecatalyst:TagResource"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/Project": "ProjectA"
        },
        "ForAllValues:StringEquals": {
          "aws:TagKeys": ["Project"]
        }
      }
    }
  ]
}
```

Contoh 2: Izinkan tindakan berdasarkan tag sumber daya

Kebijakan berikut memberi pengguna izin untuk melakukan tindakan pada, dan mendapatkan informasi tentang, sumber daya koneksi akun.

Untuk melakukan itu, ini memungkinkan tindakan tertentu jika koneksi memiliki tag bernama `Project` dengan nilai`ProjectA`. (Kunci syarat `aws:ResourceTag` digunakan untuk mengontrol tanda yang dapat diteruskan dalam permintaan IAM.)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecatalyst:GetConnection",
        "codecatalyst:DeleteConnection",
        "codecatalyst:AssociateIamRoleToConnection",
        "codecatalyst:DisassociateIamRoleFromConnection",
        "codecatalyst:ListIamRolesForConnection",
        "codecatalyst:PutBillingAuthorization"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Project": "ProjectA"
        }
      }
    }
  ]
}
```

CodeCatalyst referensi izin

Bagian ini menyediakan referensi izin untuk tindakan yang digunakan dengan sumber daya koneksi akun untuk Akun AWS yang terhubung. CodeCatalyst Bagian berikut menjelaskan tindakan khusus izin yang terkait dengan menghubungkan akun.

Izin yang diperlukan untuk koneksi akun

Izin berikut diperlukan untuk bekerja dengan koneksi akun.

CodeCatalyst izin untuk koneksi akun	Izin yang diperlukan	Sumber daya
AcceptConnection	Diperlukan untuk menerima permintaan untuk menghubungkan akun ini ke CodeCatalyst spasi. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	Hanya mendukung wildcard (*) di `Resource` elemen kebijakan.
AssociateIamRoleToConnection	Diperlukan untuk mengaitkan peran IAM ke koneksi akun. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/connections/` *`connection_ID`*
DeleteConnection	Diperlukan untuk menghapus koneksi akun. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/connections/` *`connection_ID`*
DisassociateIamRoleFromConnection	Diperlukan untuk memisahkan peran IAM dari koneksi akun. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/connections/` *`connection_ID`*
GetBillingAuthorization	Diperlukan untuk menjelaskan otorisasi penagihan untuk koneksi akun. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/connections/` *`connection_ID`*
GetConnection	Diperlukan untuk mendapatkan koneksi akun. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/connections/` *`connection_ID`*
GetPendingConnection	Diperlukan untuk mendapatkan permintaan yang tertunda untuk menghubungkan akun	Hanya mendukung wildcard (*) di `Resource` elemen kebijakan.

CodeCatalyst izin untuk koneksi akun	Izin yang diperlukan	Sumber daya
	ini ke CodeCatalyst spasi. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	
ListConnections	Diperlukan untuk membuat daftar koneksi akun yang tidak tertunda. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	Hanya mendukung wildcard (*) di `Resource` elemen kebijakan.
ListIamRolesForConnection	Diperlukan untuk mencantumkan peran IAM yang terkait dengan koneksi akun. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecataly st:region: account_ID :/connect ions/ connection_ID`
ListTagsForResource	Diperlukan untuk mencantumkan tag yang terkait dengan koneksi akun. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecataly st:region: account_ID :/connect ions/ connection_ID`
PutBillingAuthorization	Diperlukan untuk membuat atau memperbarui otorisasi penagihan untuk koneksi akun. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecataly st:region: account_ID :/connect ions/ connection_ID`
RejectConnection	Diperlukan untuk menolak permintaan untuk menghubungkan akun ini ke CodeCatalyst spasi. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	Hanya mendukung wildcard (*) di `Resource` elemen kebijakan.

CodeCatalyst izin untuk koneksi akun	Izin yang diperlukan	Sumber daya
TagResource	Diperlukan untuk membuat atau mengedit tag yang terkait dengan koneksi akun. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/connections/` *`connection_ID`*
UntagResource	Diperlukan untuk menghapus tag yang terkait dengan koneksi akun. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/connections/` *`connection_ID`*

Izin yang diperlukan untuk aplikasi IAM Identity Center

Izin berikut diperlukan untuk bekerja dengan aplikasi IAM Identity Center.

CodeCatalyst izin untuk aplikasi IAM Identity Center	Izin yang diperlukan	Sumber daya
AssociateIdentityCenterApplicationToSpace	Diperlukan untuk mengaitkan aplikasi Pusat Identitas IAM dengan CodeCatalyst spasi. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/identity-center-applications/` *`identity-center-application_ID`*
AssociateIdentityToIdentityCenterApplication	Diperlukan untuk mengaitkan identitas dengan aplikasi Pusat Identitas IAM untuk suatu CodeCatalyst ruang. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/identity-center-applications/` *`identity-center-application_ID`*

CodeCatalyst izin untuk aplikasi IAM Identity Center	Izin yang diperlukan	Sumber daya
BatchAssociateIdentitiesToIdentityCenterApplication	Diperlukan untuk mengaitkan beberapa identitas dengan aplikasi Pusat Identitas IAM untuk suatu CodeCatalyst spasi. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/identity-center-applications/` *`identity-center-application_ID`*
BatchDisassociateIdentitiesFromIdentityCenterApplication	Diperlukan untuk memisahkan beberapa identitas dari aplikasi Pusat Identitas IAM untuk suatu spasi. CodeCatalyst Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/identity-center-applications/` *`identity-center-application_ID`*
CreateIdentityCenterApplication	Diperlukan untuk membuat aplikasi IAM Identity Center. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/identity-center-applications/` *`identity-center-application_ID`*
CreateSpaceAdminRoleAssignment	Diperlukan untuk membuat tugas peran administrator untuk CodeCatalyst ruang tertentu dan aplikasi Pusat Identitas IAM. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/identity-center-applications/` *`identity-center-application_ID`*

CodeCatalyst izin untuk aplikasi IAM Identity Center	Izin yang diperlukan	Sumber daya
DeleteIdentityCenterApplication	Diperlukan untuk menghapus aplikasi Pusat Identitas IAM. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *account_ID* `:/identity-center-applications/` *identity-center-application_ID*
DisassociateIdentityCenterApplicationFromSpace	Diperlukan untuk memisahkan aplikasi Pusat Identitas IAM dari sebuah CodeCatalyst spasi. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *account_ID* `:/identity-center-applications/` *identity-center-application_ID*
DisassociateIdentityFromIdentityCenterApplication	Diperlukan untuk memisahkan identitas dari aplikasi Pusat Identitas IAM untuk suatu CodeCatalyst ruang. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *account_ID* `:/identity-center-applications/` *identity-center-application_ID*
GetIdentityCenterApplication	Diperlukan untuk mendapatkan informasi tentang aplikasi IAM Identity Center. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *account_ID* `:/identity-center-applications/` *identity-center-application_ID*
ListIdentityCenterApplications	Diperlukan untuk melihat daftar semua aplikasi Pusat Identitas IAM di akun. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	Hanya mendukung wildcard (*) di `Resource` elemen kebijakan.

CodeCatalyst izin untuk aplikasi IAM Identity Center	Izin yang diperlukan	Sumber daya
ListIdentityCenterApplicationsForSpace	Diperlukan untuk melihat daftar aplikasi IAM Identity Center berdasarkan CodeCatalyst spasi. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/identity-center-applications/` *`identity-center-application_ID`*
ListSpacesForIdentityCenterApplication	Diperlukan untuk melihat daftar CodeCatalyst spasi oleh aplikasi IAM Identity Center. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/identity-center-applications/` *`identity-center-application_ID`*
SynchronizeIdentityCenterApplication	Diperlukan untuk menyinkronkan aplikasi IAM Identity Center dengan backing identity store. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/identity-center-applications/` *`identity-center-application_ID`*
UpdateIdentityCenterApplication	Diperlukan untuk memperbarui aplikasi Pusat Identitas IAM. Ini adalah izin kebijakan IAM saja, bukan tindakan API.	`arn:aws:codecatalyst:region:` *`account_ID`* `:/identity-center-applications/` *`identity-center-application_ID`*

Menggunakan peran terkait layanan untuk CodeCatalyst

Amazon CodeCatalyst menggunakan peran AWS Identity and Access Management terkait layanan (IAM). Peran terkait layanan adalah jenis unik peran IAM yang ditautkan langsung ke. CodeCatalyst Peran terkait layanan telah ditentukan sebelumnya oleh CodeCatalyst dan mencakup semua izin yang diperlukan layanan untuk memanggil AWS layanan lain atas nama Anda.

Peran terkait layanan membuat pengaturan CodeCatalyst lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. CodeCatalyst mendefinisikan izin peran terkait layanan, dan kecuali ditentukan lain, hanya CodeCatalyst dapat mengambil perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, serta bahwa kebijakan izin tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran tertaut layanan hanya setelah menghapus sumber daya terkait terlebih dahulu. Ini melindungi CodeCatalyst sumber daya Anda karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran terkait layanan, silakan lihat layanan AWS yang bisa digunakan dengan IAM dan carilah layanan yang memiliki opsi Ya di kolom Peran terkait layanan. Pilih Ya dengan sebuah tautan untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Izin peran terkait layanan untuk CodeCatalyst

CodeCatalyst menggunakan peran terkait layanan bernama AmazonCodeCatalystServiceRoleForIdentityCenterApplicationSynchronization— Memungkinkan Amazon akses CodeCatalyst hanya-baca ke profil instans aplikasi dan pengguna dan grup direktori terkait atas nama Anda.

Peran tertaut layanan AmazonCodeCatalystServiceRoleForIdentityCenterApplicationSynchronization memercayai layanan berikut untuk mengambil peran tersebut:

- `codecatalyst.amazonaws.com`

Kebijakan izin peran bernama AmazonCodeCatalystServiceRoleForIdentityCenterApplicationSynchronizationPolicy memungkinkan CodeCatalyst untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- Aksi: `View application instance profiles and associated directory users and groups` untuk `CodeCatalyst spaces that support identity federation and SSO users and groups`

Anda harus mengonfigurasi izin agar pengguna, grup, atau peran Anda membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, lihat Izin peran tertaut layanan dalam Panduan Pengguna IAM.

Membuat peran terkait layanan untuk CodeCatalyst

Anda tidak perlu membuat peran tertaut layanan secara manual. Saat Anda membuat spasi diAWS Management Console, APIAWS CLI, atau AWS API, CodeCatalyst buat peran terkait layanan untuk Anda.

> **Important**
>
> Peran tertaut layanan ini dapat muncul di akun Anda jika Anda menyelesaikan tindakan di layanan lain yang menggunakan fitur yang disupport oleh peran ini. Juga, jika Anda menggunakan CodeCatalyst layanan sebelum 17 November 2023, ketika mulai mendukung peran terkait layanan, maka CodeCatalyst buat AmazonCodeCatalystServiceRoleForIdentityCenterApplicationSynchronization peran tersebut di akun Anda. Untuk mempelajari lebih lanjut, lihat Peran baru muncul di saya Akun AWS.

Jika Anda menghapus peran terkait layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Saat Anda membuat ruang, CodeCatalyst buat peran terkait layanan untuk Anda lagi.

Anda juga dapat menggunakan konsol IAM untuk membuat peran terkait layanan dengan Lihat profil instance aplikasi dan kasus penggunaan pengguna dan grup direktori terkait. Di AWS CLI atau API AWS, buat peran terkait layanan dengan nama layanan `codecatalyst.amazonaws.com`. Untuk informasi selengkapnya, lihat Membuat peran tertaut layanan dalam Panduan Pengguna IAM. Jika Anda menghapus peran tertaut layanan ini, Anda dapat mengulang proses yang sama untuk membuat peran tersebut lagi.

Mengedit peran terkait layanan untuk CodeCatalyst

CodeCatalyst tidak memungkinkan Anda untuk mengedit peran AmazonCodeCatalystServiceRoleForIdentityCenterApplicationSynchronization terkait layanan. Setelah membuat peran tertaut layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat mengedit deskripsi peran menggunakan IAM. Untuk informasi selengkapnya, lihat Mengedit peran tertaut layanan dalam Panduan Pengguna IAM.

Menghapus peran terkait layanan untuk CodeCatalyst

Anda tidak perlu menghapus peran AmazonCodeCatalystServiceRoleForIdentityCenterApplicationSynchronization secara manual. Saat

Anda menghapus spasi diAWS Management Console, APIAWS CLI, atau AWS API, CodeCatalyst membersihkan sumber daya dan menghapus peran terkait layanan untuk Anda.

Anda juga dapat menggunakan konsol IAM, AWS CLI, atau API AWS untuk menghapus peran tertaut layanan secara manual. Untuk melakukannya, Anda harus membersihkan sumber daya untuk peran tertaut layanan terlebih dahulu, lalu Anda dapat menghapusnya secara manual.

> **Note**
>
> Jika CodeCatalyst layanan menggunakan peran saat Anda mencoba menghapus sumber daya, maka penghapusan mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk menghapus CodeCatalyst sumber daya yang digunakan oleh AmazonCodeCatalystServiceRoleForIdentityCenterApplicationSynchronization

- Hapus spasi.

Untuk menghapus peran terkait layanan secara manual menggunakan IAM

Gunakan konsol IAM, AWS CLI, atau API AWS untuk menghapus peran tertaut layanan AmazonCodeCatalystServiceRoleForIdentityCenterApplicationSynchronization. Untuk informasi selengkapnya, lihat Menghapus peran tertaut layanan dalam Panduan Pengguna IAM.

Wilayah yang Didukung untuk CodeCatalyst peran terkait layanan

CodeCatalyst mendukung penggunaan peran terkait layanan di semua Wilayah tempat layanan tersedia. Untuk informasi selengkapnya, lihat AWSWilayah dan titik akhir.

CodeCatalyst tidak mendukung penggunaan peran terkait layanan di setiap Wilayah tempat layanan tersedia. Anda dapat menggunakan AmazonCodeCatalystServiceRoleForIdentityCenterApplicationSynchronization peran di Wilayah berikut.

Nama Wilayah	Identitas wilayah	Support di CodeCatalyst
US East (Northern Virginia)	us-east-1	Tidak
Timur AS (Ohio)	us-east-2	Tidak

Nama Wilayah	Identitas wilayah	Support di CodeCatalyst
US West (Northern California)	us-west-1	Tidak
AS Barat (Oregon)	us-west-2	Ya
Afrika (Cape Town)	af-south-1	Tidak
Asia Pasifik (Hong Kong)	ap-east-1	Tidak
Asia Pasifik (Jakarta)	ap-southeast-3	Tidak
Asia Pasifik (Mumbai)	ap-south-1	Tidak
Asia Pacific (Osaka)	ap-northeast-3	Tidak
Asia Pasifik (Seoul)	ap-northeast-2	Tidak
Asia Pasifik (Singapura)	ap-southeast-1	Tidak
Asia Pasifik (Sydney)	ap-southeast-2	Tidak
Asia Pasifik (Tokyo)	ap-northeast-1	Tidak
Kanada (Pusat)	ca-central-1	Tidak
Eropa (Frankfurt)	eu-central-1	Tidak
Eropa (Irlandia)	eu-west-1	Ya
Eropa (London)	eu-west-2	Tidak
Eropa (Milan)	eu-south-1	Tidak
Eropa (Paris)	eu-west-3	Tidak
Eropa (Stockholm)	eu-north-1	Tidak
Timur Tengah (Bahrain)	me-south-1	Tidak
Timur Tengah (UAE)	me-central-1	Tidak

Nama Wilayah	Identitas wilayah	Support di CodeCatalyst
Amerika Selatan (Sao Paulo)	sa-east-1	Tidak
AWS GovCloud (AS-Timur)	us-gov-east-1	Tidak
AWS GovCloud (AS-Barat)	us-gov-west-1	Tidak

AWSkebijakan terkelola untuk Amazon CodeCatalyst

Kebijakan terkelola AWS adalah kebijakan mandiri yang dibuat dan dikelola AWS. Kebijakan terkelola AWS dirancang untuk memberikan izin bagi banyak kasus penggunaan umum agar Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan yang dikelola AWS mungkin tidak memberikan izin hak akses paling rendah untuk kasus penggunaan spesifik Anda karena kebijakan ini tersedia untuk digunakan semua pelanggan AWS. Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan yang dikelola AWS. Jika AWS memperbarui izin yang ditentukan dalam sebuah kebijakan yang dikelola AWS, maka pembaruan tersebut akan memengaruhi semua identitas prinsipal (pengguna, grup, dan peran) yang terkait dengan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan yang dikelola AWS saat sebuah Layanan AWS baru diluncurkan atau operasi API baru tersedia untuk layanan yang sudah ada.

Untuk informasi selengkapnya, lihat Kebijakan yang dikelola AWS dalam Panduan Pengguna IAM.

Kebijakan terkelola AWS: AmazonCodeCatalystSupportAccess

Ini adalah kebijakan yang memberikan izin untuk semua administrator ruang dan anggota ruang untuk menggunakan paket dukungan premium Bisnis atau Perusahaan yang terkait dengan akun

penagihan ruang. Izin ini memungkinkan administrator dan anggota ruang untuk menggunakan paket dukungan premium untuk sumber daya yang mereka miliki izin dalam kebijakan izin. CodeCatalyst

Detail izin

Kebijakan ini mencakup izin berikut.

- `support`— Memberikan izin untuk memungkinkan pengguna mencari, membuat, dan menyelesaikan kasus AWS Support. Juga memberikan izin untuk menjelaskan komunikasi, tingkat keparahan, lampiran, dan detail kasus dukungan terkait.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "support:DescribeAttachment",
        "support:DescribeCaseAttributes",
        "support:DescribeCases",
        "support:DescribeCommunications",
        "support:DescribeIssueTypes",
        "support:DescribeServices",
        "support:DescribeSeverityLevels",
        "support:DescribeSupportLevel",
        "support:SearchForCases",
        "support:AddAttachmentsToSet",
        "support:AddCommunicationToCase",
        "support:CreateCase",
        "support:InitiateCallForCase",
        "support:InitiateChatForCase",
        "support:PutCaseAttributes",
        "support:RateCaseCommunication",
        "support:ResolveCase"
      ],
      "Resource": "*"
    }
  ]
```

```
}
```

Kebijakan terkelola AWS: AmazonCodeCatalystFullAccess

Ini adalah kebijakan yang memberikan izin untuk mengelola CodeCatalyst ruang dan akun yang terhubung di halaman Amazon CodeCatalyst Spaces di halaman. AWS Management Console Aplikasi ini digunakan untuk mengkonfigurasi Akun AWS yang terhubung ke ruang Anda di CodeCatalyst.

Detail izin

Kebijakan ini mencakup izin berikut.

- `codecatalyst`— Memberikan izin penuh ke halaman Amazon CodeCatalyst Spaces di. AWS Management Console

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CodeCatalystResourceAccess"
            "Effect": "Allow",
            "Action": [
                "codecatalyst:*",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CodeCatalystAssociateIAMRole"
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
```

```
                    "iam:PassedToService": [
                        "codecatalyst.amazonaws.com",
                        "codecatalyst-runner.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

Kebijakan terkelola AWS: AmazonCodeCatalystReadOnlyAccess

Ini adalah kebijakan yang memberikan izin untuk melihat dan mencantumkan informasi untuk spasi dan akun yang terhubung di halaman Amazon CodeCatalyst Spaces di halaman. AWS Management Console Aplikasi ini digunakan untuk mengkonfigurasi Akun AWS yang terhubung ke ruang Anda di CodeCatalyst.

Detail izin

Kebijakan ini mencakup izin berikut.

- `codecatalyst`— Memberikan izin hanya-baca ke halaman Amazon CodeCatalyst Spaces di halaman. AWS Management Console

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codecatalyst:Get*",
                "codecatalyst:List*",
            ],
            "Resource": "*"
        }
    ]
}
```

Kebijakan terkelola AWS: AmazonCodeCatalystServiceRoleForIdentityCenterApplicationSynchronizationPolicy

Anda tidak dapat melampirkanAmazonCodeCatalystServiceRoleForIdentityCenterApplicationSynchronizationPolicy; ke entitas IAM Anda. Kebijakan ini dilampirkan pada peran terkait layanan yang memungkinkan CodeCatalyst untuk melakukan tindakan atas nama Anda. Untuk informasi selengkapnya, lihat Menggunakan peran terkait layanan untuk CodeCatalyst.

Kebijakan ini memungkinkan pelanggan untuk melihat profil instance aplikasi dan pengguna serta grup direktori terkait saat mengelola spasi CodeCatalyst. Pelanggan akan melihat sumber daya ini saat mengelola ruang yang mendukung federasi identitas dan pengguna dan grup SSO.

Detail izin

Kebijakan ini mencakup izin berikut.

- `sso`— Memberikan izin untuk memungkinkan pengguna melihat profil instance aplikasi yang dikelola di Pusat Identitas IAM untuk ruang terkait di. CodeCatalyst

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid":
 "AmazonCodeCatalystServiceRoleForIdentityCenterApplicationSynchronizationPolicy",
   "Effect": "Allow",
   "Action": [
    "sso:ListInstances",
    "sso:ListApplications",
    "sso:ListApplicationAssignments",
    "sso:DescribeInstance",
    "sso:DescribeApplication"
   ],
   "Resource": "*"
  }
```

```
 ]
}
```

CodeCatalyst pembaruan kebijakan AWS terkelola

Lihat detail tentang pembaruan kebijakan AWS terkelola CodeCatalyst sejak layanan ini mulai melacak perubahan ini. Untuk peringatan otomatis tentang perubahan pada halaman ini, berlangganan umpan RSS di halaman Riwayat CodeCatalyst dokumen.

Perubahan	Deskripsi	Tanggal
AmazonCodeCatalyst ServiceRoleForIdentityCenterApplicationSynchronization Policy – Kebijakan baru	CodeCatalyst menambahkan kebijakan. Memberikan izin untuk memungkinkan CodeCatalyst pengguna melihat profil instance aplikasi dan pengguna dan grup direktori terkait.	17 November 2023
AmazonCodeCatalyst SupportAccess – Kebijakan baru	CodeCatalyst menambahkan kebijakan. Memberikan izin untuk memungkinkan CodeCatalyst pengguna mencari, membuat, dan menyelesaikan kasus dukungan, serta melihat komunikasi dan detail terkait.	20 April 2023
AmazonCodeCatalyst FullAccess – Kebijakan baru	CodeCatalyst menambahkan kebijakan. Memberikan akses penuh ke CodeCatalyst.	20 April 2023

Perubahan	Deskripsi	Tanggal
AmazonCodeCatalyst ReadOnlyAccess – Kebijakan baru	CodeCatalyst menambahkan kebijakan. Memberikan akses hanya-bac a ke. CodeCatalyst	20 April 2023
CodeCatalyst mulai melacak perubahan	CodeCatalyst mulai melacak perubahan untuk kebijakan yang AWS dikelola.	20 April 2023

Berikan akses ke AWS sumber daya proyek dengan peran IAM

CodeCatalyst dapat mengakses AWS sumber daya dengan menghubungkan Anda Akun AWS ke CodeCatalyst ruang. Anda kemudian dapat membuat peran layanan berikut dan mengaitkannya saat Anda menghubungkan akun Anda.

Untuk informasi selengkapnya tentang elemen yang Anda gunakan dalam kebijakan JSON, lihat Referensi Elemen Kebijakan IAM JSON di Panduan Pengguna IAM.

- Untuk mengakses sumber daya dalam CodeCatalyst proyek dan alur kerja Anda, Anda harus terlebih dahulu memberikan izin CodeCatalyst untuk mengakses sumber daya tersebut atas nama Anda. Akun AWS Untuk melakukannya, Anda harus membuat peran layanan dalam koneksi Akun AWS yang CodeCatalyst dapat diasumsikan atas nama pengguna dan proyek di ruang tersebut. Anda dapat memilih untuk membuat dan menggunakan peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan, atau Anda dapat membuat peran layanan yang disesuaikan dan mengonfigurasi kebijakan dan peran IAM ini secara manual. Sebagai praktik terbaik, tetapkan peran ini paling sedikit izin yang diperlukan.

  > **Note**
  >
  > Untuk peran layanan yang disesuaikan, prinsipal CodeCatalyst layanan diperlukan. Untuk informasi lebih lanjut tentang prinsip CodeCatalyst layanan dan model kepercayaan, lihatMemahami model CodeCatalyst kepercayaan.

- Untuk mengelola dukungan ruang melalui koneksi Akun AWS, Anda dapat memilih untuk membuat dan menggunakan peran AWSRoleForCodeCatalystSupportlayanan yang memungkinkan CodeCatalyst pengguna mengakses dukungan. Untuk informasi selengkapnya tentang dukungan untuk CodeCatalyst ruang, lihatAWS Supportuntuk Amazon CodeCatalyst.

Memahami peran CodeCatalystWorkflowDevelopmentRole-*spaceName*layanan

Anda dapat menambahkan peran IAM untuk ruang Anda yang CodeCatalyst dapat digunakan untuk membuat dan mengakses sumber daya yang terhubung Akun AWS. Ini disebut peran layanan. Cara termudah untuk membuat peran layanan adalah dengan menambahkan satu ketika Anda membuat ruang dan memilih CodeCatalystWorkflowDevelopmentRole-*spaceName*opsi untuk peran itu. Ini tidak hanya menciptakan peran layanan dengan `AdministratorAccess` terlampir, tetapi juga menciptakan kebijakan kepercayaan yang memungkinkan CodeCatalyst untuk mengambil peran atas nama pengguna dalam proyek di ruang. Peran layanan dicakup ke ruang, bukan untuk proyek individu. Untuk membuat peran ini, lihat Membuat CodeCatalystWorkflowDevelopmentRole-*spaceName*peran untuk akun dan ruang Anda. Anda hanya dapat membuat satu peran untuk setiap ruang di setiap akun.

> **Note**
>
> Peran ini hanya direkomendasikan untuk digunakan dengan akun pengembangan dan menggunakan kebijakan `AdministratorAccess` AWS terkelola, memberikan akses penuh untuk membuat kebijakan dan sumber daya baru dalam hal ini Akun AWS.

Kebijakan yang melekat pada CodeCatalystWorkflowDevelopmentRole-*spaceName*peran dirancang untuk bekerja dengan proyek yang dibuat dengan cetak biru di ruang angkasa. Hal ini memungkinkan pengguna dalam proyek-proyek untuk mengembangkan, membangun, menguji, dan menyebarkan kode menggunakan sumber daya di terhubung Akun AWS. Untuk informasi selengkapnya, lihat Membuat peran untuk AWS layanan.

Kebijakan yang melekat pada CodeCatalystWorkflowDevelopmentRole-*spaceName*peran tersebut adalah kebijakan yang `AdministratorAccess` dikelola AWS. Ini adalah kebijakan yang memberikan akses penuh ke semua AWS tindakan dan sumber daya. Untuk melihat dokumen kebijakan JSON di konsol IAM, lihat. AdministratorAccess

Kebijakan kepercayaan berikut memungkinkan CodeCatalyst untuk mengambil CodeCatalystWorkflowDevelopmentRole-***spaceName***peran. Untuk informasi lebih lanjut tentang model CodeCatalyst kepercayaan, lihatMemahami model CodeCatalyst kepercayaan.

```
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
             "Principal": {
                "Service": [
                    "codecatalyst-runner.amazonaws.com",
                    "codecatalyst.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:codecatalyst:::space/spaceId/project/*"
                }
            }
        }
    ]
```

Membuat CodeCatalystWorkflowDevelopmentRole-***spaceName***peran untuk akun dan ruang Anda

Ikuti langkah-langkah ini untuk membuat `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran yang akan digunakan untuk alur kerja di ruang Anda. Untuk setiap akun yang ingin memiliki peran IAM untuk digunakan dalam proyek, ke ruang Anda, Anda harus menambahkan peran seperti peran pengembang.

Sebelum Anda mulai, Anda harus memiliki hak administratif untuk Anda Akun AWS atau dapat bekerja dengan administrator Anda. Untuk informasi selengkapnya tentang bagaimana Akun AWS dan peran IAM digunakan CodeCatalyst, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS.

Untuk membuat dan menambahkan CodeCatalyst CodeCatalystWorkflowDevelopmentRole-*spaceName*

1. Sebelum Anda mulai di CodeCatalyst konsol, buka AWS Management Console, dan kemudian pastikan Anda masuk dengan yang sama Akun AWS untuk ruang Anda.
2. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
3. Arahkan ke CodeCatalyst ruang Anda. Pilih Pengaturan, lalu pilih Akun AWS.
4. Pilih tautan untuk Akun AWS tempat Anda ingin membuat peran. Halaman Akun AWS detail ditampilkan.
5. Pilih Kelola peran dari AWS Management Console.

   Peran Tambahkan IAM ke halaman CodeCatalyst ruang Amazon terbuka di. AWS Management Console Ini adalah halaman CodeCatalyst spasi Amazon. Anda mungkin perlu masuk untuk mengakses halaman.
6. Pilih Buat peran administrator CodeCatalyst pengembangan di IAM. Opsi ini membuat peran layanan yang berisi kebijakan izin dan kebijakan kepercayaan untuk peran pengembangan. Peran akan memiliki nama`CodeCatalystWorkflowDevelopmentRole-`*`spaceName`*. Untuk informasi selengkapnya tentang kebijakan peran dan peran, lihatMemahami peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan.

   > **Note**
   >
   > Peran ini hanya disarankan untuk digunakan dengan akun pengembang dan menggunakan kebijakan `AdministratorAccess` AWS terkelola, memberikan akses penuh untuk membuat kebijakan dan sumber daya baru dalam hal ini Akun AWS.
7. Pilih Buat peran pengembangan.
8. Pada halaman koneksi, di bawah peran IAM yang tersedia untuk CodeCatalyst, lihat `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran dalam daftar peran IAM yang ditambahkan ke akun Anda.
9. Untuk kembali ke ruang Anda, pilih Buka Amazon CodeCatalyst.

Memahami peran AWSRoleForCodeCatalystSupportlayanan

Anda dapat menambahkan peran IAM untuk ruang yang dapat digunakan CodeCatalyst pengguna dalam ruang untuk membuat dan mengakses kasus dukungan. Ini disebut peran layanan untuk

dukungan.Cara paling sederhana untuk membuat peran layanan untuk dukungan adalah dengan menambahkan satu ketika Anda membuat ruang dan memilih `AWSRoleForCodeCatalystSupport` opsi untuk peran itu. Ini tidak hanya menciptakan kebijakan dan peran, tetapi juga menciptakan kebijakan kepercayaan yang memungkinkan CodeCatalyst untuk mengambil peran atas nama pengguna dalam proyek di ruang angkasa. Peran layanan dicakup ke ruang, bukan untuk proyek individu. Untuk membuat peran ini, lihat Membuat AWSRoleForCodeCatalystSupportperan untuk akun dan ruang Anda.

Kebijakan yang dilampirkan pada `AWSRoleForCodeCatalystSupport` peran adalah kebijakan terkelola yang menyediakan akses ke izin dukungan. Untuk informasi selengkapnya, lihat Kebijakan terkelola AWS: AmazonCodeCatalystSupportAccess.

Peran kepercayaan untuk kebijakan memungkinkan CodeCatalyst untuk mengambil peran.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "codecatalyst.amazonaws.com",
                    "codecatalyst-runner.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Membuat AWSRoleForCodeCatalystSupportperan untuk akun dan ruang Anda

Ikuti langkah-langkah ini untuk membuat `AWSRoleForCodeCatalystSupport` peran yang akan digunakan untuk kasus dukungan di ruang Anda. Peran harus ditambahkan ke akun penagihan yang ditunjuk untuk ruang tersebut.

Sebelum Anda mulai, Anda harus memiliki hak administratif untuk Anda Akun AWS atau dapat bekerja dengan administrator Anda. Untuk informasi selengkapnya tentang bagaimana Akun AWS dan peran IAM digunakan CodeCatalyst, lihatMemungkinkan akses ke AWS sumber daya yang terhubung Akun AWS.

Untuk membuat dan menambahkan CodeCatalyst AWSRoleForCodeCatalystSupport

1. Sebelum Anda mulai di CodeCatalyst konsol, buka AWS Management Console, dan kemudian pastikan Anda masuk dengan yang sama Akun AWS untuk ruang Anda.
2. Arahkan ke CodeCatalyst ruang Anda. Pilih Pengaturan, lalu pilih Akun AWS.
3. Pilih tautan untuk Akun AWS tempat Anda ingin membuat peran. Halaman Akun AWS detail ditampilkan.
4. Pilih Kelola peran dari AWS Management Console.

   Peran Tambahkan IAM ke halaman CodeCatalyst ruang Amazon terbuka di. AWS Management Console Ini adalah halaman Amazon CodeCatalyst Spaces. Anda mungkin perlu masuk untuk mengakses halaman.
5. Di bawah detail CodeCatalyst spasi, pilih peran Tambah CodeCatalyst Dukungan. Opsi ini membuat peran layanan yang berisi kebijakan izin dan kebijakan kepercayaan untuk peran pengembangan pratinjau. Peran akan memiliki nama AWSRoleForCodeCatalystSupportdengan pengenal unik ditambahkan. Untuk informasi selengkapnya tentang kebijakan peran dan peran, lihatMemahami peran AWSRoleForCodeCatalystSupportlayanan.
6. Pada halaman Add role for CodeCatalyst Support, biarkan default dipilih, lalu pilih Create role.
7. Di bawah peran IAM yang tersedia CodeCatalyst, lihat `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran dalam daftar peran IAM yang ditambahkan ke akun Anda.
8. Untuk kembali ke ruang Anda, pilih Buka Amazon CodeCatalyst.

Mengkonfigurasi peran IAM untuk tindakan alur kerja di CodeCatalyst

Bagian ini merinci peran dan kebijakan IAM yang dapat Anda buat untuk digunakan dengan CodeCatalyst akun Anda. Untuk instruksi untuk membuat contoh peran, lihatMembuat peran secara manual untuk tindakan alur kerja. Setelah Anda membuat peran IAM, salin peran ARN untuk menambahkan peran IAM ke koneksi akun Anda dan kaitkan dengan lingkungan proyek Anda. Untuk mempelajari selengkapnya, lihat Menambahkan peran IAM ke koneksi akun.

CodeCatalyst membangun peran untuk akses Amazon S3

Untuk tindakan pembuatan CodeCatalyst alur kerja, Anda dapat menggunakan peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan default, atau Anda dapat membuat peran IAM bernama CodeCatalystBuildRolefor S3Access. Peran ini menggunakan kebijakan dengan

izin terbatas yang CodeCatalyst perlu menjalankan tugas pada AWS CloudFormation sumber daya di Anda. Akun AWS

Peran ini memberikan izin untuk melakukan hal berikut:

- Menulis ke ember Amazon S3.
- Support membangun sumber daya dengan AWS CloudFormation. Ini membutuhkan akses Amazon S3.

Peran ini menggunakan kebijakan berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [{
    "Action": [
        "s3:PutObject",
        "iam:PassRole"
    ],
    "Resource": "resource_ARN",
    "Effect": "Allow"
}]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

CodeCatalyst membangun peran untuk AWS CloudFormation

Untuk tindakan pembuatan CodeCatalyst alur kerja, Anda dapat menggunakan peran CodeCatalystWorkflowDevelopmentRole-*spaceName*layanan default, atau Anda dapat membuat peran IAM dengan izin yang diperlukan. Peran ini menggunakan kebijakan dengan izin terbatas yang CodeCatalyst perlu menjalankan tugas pada AWS CloudFormation sumber daya di Anda. Akun AWS

Peran ini memberikan izin untuk melakukan hal berikut:

- Support membangun sumber daya dengan AWS CloudFormation. Ini diperlukan bersama dengan peran CodeCatalyst build untuk akses Amazon S3 dan peran CodeCatalyst penerapan untuk. AWS CloudFormation

Kebijakan AWS terkelola berikut harus dilampirkan pada peran ini:

- AWSCloudFormationFullAccess
- IAM FullAccess
- AmazonS3 FullAccess
- AmazonAPI GatewayAdministrator
- AWSLambdaFullAccess

CodeCatalyst membangun peran untuk CDK

Untuk CodeCatalyst alur kerja yang menjalankan tindakan build CDK,
seperti aplikasi web tiga tingkat modern, Anda dapat menggunakan peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan default, atau Anda dapat membuat peran IAM dengan izin yang diperlukan. Peran ini menggunakan kebijakan dengan izin terbatas yang CodeCatalyst perlu bootstrap dan menjalankan perintah build CDK untuk sumber daya di Anda. AWS CloudFormation Akun AWS

Peran ini memberikan izin untuk melakukan hal berikut:

- Menulis ke ember Amazon S3.
- Support membangun konstruksi CDK dan tumpukan AWS CloudFormation sumber daya. Ini memerlukan akses ke Amazon S3 untuk penyimpanan artefak, Amazon ECR untuk dukungan repositori gambar, dan SSM untuk tata kelola sistem dan pemantauan untuk instans virtual.

Peran ini menggunakan kebijakan berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "cloudformation:*",
                "ecr:*",
                "ssm:*",
                "s3:*",
                "iam:PassRole",
                "iam:GetRole",
                "iam:CreateRole",
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

CodeCatalyst menyebarkan peran untuk AWS CloudFormation

Untuk tindakan penerapan CodeCatalyst alur kerja yang digunakan AWS CloudFormation, Anda dapat menggunakan peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan default, atau Anda dapat menggunakan kebijakan dengan izin cakupan yang CodeCatalyst diperlukan untuk menjalankan tugas pada sumber daya di Anda. AWS CloudFormation Akun AWS

Peran ini memberikan izin untuk melakukan hal berikut:

- Izinkan CodeCatalyst untuk memanggil fungsi Λ untuk melakukan penyebaran biru/hijau melalui. AWS CloudFormation
- Izinkan CodeCatalyst untuk membuat dan memperbarui tumpukan dan set perubahan di. AWS CloudFormation

Peran ini menggunakan kebijakan berikut:

```
{"Action": [
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack",
        "cloudformation:Describe*",
        "cloudformation:UpdateStack",
        "cloudformation:CreateChangeSet",
        "cloudformation:DeleteChangeSet",
        "cloudformation:ExecuteChangeSet",
        "cloudformation:SetStackPolicy",
        "cloudformation:ValidateTemplate",
        "cloudformation:List*",
        "iam:PassRole"
    ],
    "Resource": "resource_ARN",
    "Effect": "Allow"
}
```

Note

Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.

```
"Resource": "*"
```

CodeCatalyst menyebarkan peran untuk Amazon EC2

CodeCatalyst tindakan penerapan alur kerja menggunakan peran IAM dengan izin yang diperlukan. Peran ini menggunakan kebijakan dengan izin terbatas yang CodeCatalyst perlu menjalankan tugas di sumber daya Amazon EC2 di Anda. Akun AWS Kebijakan default untuk CodeCatalystWorkflowDevelopmentRole-***spaceName***peran tidak menyertakan izin untuk Amazon EC2 atau Amazon EC2 Auto Scaling.

Peran ini memberikan izin untuk melakukan hal berikut:

- Buat penerapan Amazon EC2.
- Baca tag pada instans atau identifikasi instans Amazon EC2 dengan nama grup Auto Scaling.

- Baca, buat, perbarui, dan hapus grup Auto Scaling Amazon EC2, kait siklus hidup, dan kebijakan penskalaan.
- Publikasikan informasi ke topik Amazon SNS.
- Ambil informasi tentang CloudWatch alarm.
- Baca dan perbarui Elastic Load Balancing.

Peran ini menggunakan kebijakan berikut:

```
{
"Version": "2012-10-17",
"Statement": [
{
"Effect": "Allow",
"Action": [
 "autoscaling:CompleteLifecycleAction",
"autoscaling:DeleteLifecycleHook",
"autoscaling:DescribeAutoScalingGroups",
"autoscaling:DescribeLifecycleHooks",
"autoscaling:PutLifecycleHook",
"autoscaling:RecordLifecycleActionHeartbeat",
"autoscaling:CreateAutoScalingGroup",
"autoscaling:UpdateAutoScalingGroup",
"autoscaling:EnableMetricsCollection",
"autoscaling:DescribePolicies",
"autoscaling:DescribeScheduledActions",
"autoscaling:DescribeNotificationConfigurations",
"autoscaling:SuspendProcesses",
"autoscaling:ResumeProcesses",
"autoscaling:AttachLoadBalancers",
"autoscaling:AttachLoadBalancerTargetGroups",
"autoscaling:PutScalingPolicy",
"autoscaling:PutScheduledUpdateGroupAction",
"autoscaling:PutNotificationConfiguration",
"autoscaling:PutWarmPool",
"autoscaling:DescribeScalingActivities",
"autoscaling:DeleteAutoScalingGroup",
"ec2:DescribeInstances",
"ec2:DescribeInstanceStatus",
"ec2:TerminateInstances",
"tag:GetResources",
"sns:Publish",
```

```
"cloudwatch:DescribeAlarms",
"cloudwatch:PutMetricAlarm",
"elasticloadbalancing:DescribeLoadBalancers",
"elasticloadbalancing:DescribeInstanceHealth",
"elasticloadbalancing:RegisterInstancesWithLoadBalancer",
"elasticloadbalancing:DeregisterInstancesFromLoadBalancer",
"elasticloadbalancing:DescribeTargetGroups",
"elasticloadbalancing:DescribeTargetHealth",
"elasticloadbalancing:RegisterTargets",
"elasticloadbalancing:DeregisterTargets"
],
"Resource": "resource_ARN"
      }
   ]
}
```

Note

Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.

```
"Resource": "*"
```

CodeCatalyst menyebarkan peran untuk Amazon ECS

Untuk tindakan CodeCatalyst alur kerja, Anda dapat membuat peran IAM dengan izin yang diperlukan. Anda dapat menggunakan peran CodeCatalystWorkflowDevelopmentRole-*spaceName*layanan default, atau Anda dapat membuat peran IAM untuk tindakan CodeCatalyst penerapan yang akan digunakan untuk penerapan Lambda. Peran ini menggunakan kebijakan dengan izin terbatas yang CodeCatalyst perlu menjalankan tugas di sumber daya Amazon ECS di Anda. Akun AWS

Peran ini memberikan izin untuk melakukan hal berikut:

- Memulai penerapan Amazon ECS yang bergulir atas nama CodeCatalyst pengguna, di akun yang ditentukan dalam koneksi. CodeCatalyst
- Baca, perbarui, dan hapus set tugas Amazon ECS.
- Perbarui grup target Elastic Load Balancing, pendengar, dan aturan.

- Memanggil fungsi Lambda.
- Akses file revisi di bucket Amazon S3.
- Ambil informasi tentang CloudWatch alarm.
- Publikasikan informasi ke topik Amazon SNS.

Peran ini menggunakan kebijakan berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [{
    "Action":[
      "ecs:DescribeServices",
      "ecs:CreateTaskSet",
      "ecs:DeleteTaskSet",
      "ecs:ListClusters",
      "ecs:RegisterTaskDefinition",
      "ecs:UpdateServicePrimaryTaskSet",
      "ecs:UpdateService",
      "elasticloadbalancing:DescribeTargetGroups",
      "elasticloadbalancing:DescribeListeners",
      "elasticloadbalancing:ModifyListener",
      "elasticloadbalancing:DescribeRules",
      "elasticloadbalancing:ModifyRule",
      "lambda:InvokeFunction",
      "lambda:ListFunctions",
      "cloudwatch:DescribeAlarms",
      "sns:Publish",
      "sns:ListTopics",
      "s3:GetObject",
      "s3:GetObjectVersion",
      "codedeploy:CreateApplication",
      "codedeploy:CreateDeployment",
      "codedeploy:CreateDeploymentGroup",
      "codedeploy:GetApplication",
      "codedeploy:GetDeployment",
      "codedeploy:GetDeploymentGroup",
      "codedeploy:ListApplications",
      "codedeploy:ListDeploymentGroups",
      "codedeploy:ListDeployments",
      "codedeploy:StopDeployment",
      "codedeploy:GetDeploymentTarget",
```

```
      "codedeploy:ListDeploymentTargets",
      "codedeploy:GetDeploymentConfig",
      "codedeploy:GetApplicationRevision",
      "codedeploy:RegisterApplicationRevision",
      "codedeploy:BatchGetApplicationRevisions",
      "codedeploy:BatchGetDeploymentGroups",
      "codedeploy:BatchGetDeployments",
      "codedeploy:BatchGetApplications",
      "codedeploy:ListApplicationRevisions",
      "codedeploy:ListDeploymentConfigs",
      "codedeploy:ContinueDeployment"
   ],
   "Resource":"*",
   "Effect":"Allow"
},{"Action":[
      "iam:PassRole"
   ],
   "Effect":"Allow",
   "Resource":"*",
   "Condition":{"StringLike":{"iam:PassedToService":[
            "ecs-tasks.amazonaws.com",
            "codedeploy.amazonaws.com"
         ]
      }
   }
}]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

CodeCatalyst menyebarkan peran untuk Lambda

Untuk tindakan CodeCatalyst alur kerja, Anda dapat membuat peran IAM dengan izin yang diperlukan. Anda dapat menggunakan peran

CodeCatalystWorkflowDevelopmentRole-*spaceName*layanan default, atau Anda membuat peran IAM untuk tindakan CodeCatalyst penerapan yang akan digunakan untuk penerapan Lambda. Peran ini menggunakan kebijakan dengan izin cakupan yang CodeCatalyst diperlukan untuk menjalankan tugas pada sumber daya Lambda di sumber daya Lambda. Akun AWS

Peran ini memberikan izin untuk melakukan hal berikut:

- Baca, perbarui, dan panggil fungsi dan alias Lambda.
- Akses file revisi di bucket Amazon S3.
- Ambil informasi tentang alarm CloudWatch Acara.
- Publikasikan informasi ke topik Amazon SNS.

Peran ini menggunakan kebijakan berikut:

```
*{*
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "cloudwatch:DescribeAlarms",
                "lambda:UpdateAlias",
                "lambda:GetAlias",
                "lambda:GetProvisionedConcurrencyConfig",
                "sns:Publish"
            ],
            "Resource": "resource_ARN",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "arn:aws:s3:::/CodeDeploy/",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
```

```
            "Resource": "",
            "Condition": {
                "StringEquals": {
                    "s3:ExistingObjectTag/UseWithCodeDeploy": "true"
                }
            },
            "Effect": "Allow"
        },
        {
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": "arn:aws:lambda:::function:CodeDeployHook_*",
            "Effect": "Allow"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

CodeCatalyst menyebarkan peran untuk Lambda

Untuk tindakan CodeCatalyst alur kerja, Anda dapat menggunakan peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan default, atau Anda dapat membuat peran IAM dengan izin yang diperlukan. Peran ini menggunakan kebijakan dengan izin cakupan yang CodeCatalyst diperlukan untuk menjalankan tugas pada sumber daya Lambda di sumber daya Lambda. Akun AWS

Peran ini memberikan izin untuk melakukan hal berikut:

- Baca, perbarui, dan panggil fungsi dan alias Lambda.
- Akses file revisi di bucket Amazon S3.
- Ambil informasi tentang CloudWatch alarm.

- Publikasikan informasi ke topik Amazon SNS.

Peran ini menggunakan kebijakan berikut:

```
*{*
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "cloudwatch:DescribeAlarms",
                "lambda:UpdateAlias",
                "lambda:GetAlias",
                "lambda:GetProvisionedConcurrencyConfig",
                "sns:Publish"
            ],
            "Resource": "resource_ARN",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "arn:aws:s3:::/CodeDeploy/",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "",
            "Condition": {
                "StringEquals": {
                    "s3:ExistingObjectTag/UseWithCodeDeploy": "true"
                }
            },
            "Effect": "Allow"
        },
        {
            "Action": [
                "lambda:InvokeFunction"
            ],
```

```
            "Resource": "arn:aws:lambda:::function:CodeDeployHook_*",
            "Effect": "Allow"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

CodeCatalyst menyebarkan peran untuk AWS SAM

Untuk tindakan CodeCatalyst alur kerja, Anda dapat menggunakan peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan default, atau Anda dapat membuat peran IAM dengan izin yang diperlukan. Peran ini menggunakan kebijakan dengan izin cakupan yang CodeCatalyst diperlukan untuk menjalankan tugas AWS SAM dan AWS CloudFormation sumber daya di Anda. Akun AWS

Peran ini memberikan izin untuk melakukan hal berikut:

- Izinkan CodeCatalyst untuk memanggil fungsi Lambda untuk melakukan penerapan aplikasi tanpa server dan CLI. AWS SAM
- Izinkan CodeCatalyst untuk membuat dan memperbarui tumpukan dan set perubahan di. AWS CloudFormation

Peran ini menggunakan kebijakan berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
```

```
                "s3:GetObject",
                "iam:PassRole",
                "iam:DeleteRole",
                "iam:GetRole",
                "iam:TagRole",
                "iam:CreateRole",
                "iam:AttachRolePolicy",
                "iam:DetachRolePolicy",
                "cloudformation:*",
                "lambda:*",
                "apigateway:*"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

CodeCatalyst peran baca saja untuk Amazon EC2

Untuk tindakan CodeCatalyst alur kerja, Anda dapat membuat peran IAM dengan izin yang diperlukan. Peran ini menggunakan kebijakan dengan izin terbatas yang CodeCatalyst perlu menjalankan tugas di sumber daya Amazon EC2 di Anda. Akun AWS Peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan tidak menyertakan izin untuk Amazon EC2 atau tindakan yang dijelaskan untuk Amazon. CloudWatch

Peran ini memberikan izin untuk melakukan hal berikut:

- Dapatkan status instans Amazon EC2.
- Dapatkan CloudWatch metrik untuk instans Amazon EC2.

Peran ini menggunakan kebijakan berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe",
            "Resource": "resource_ARN"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe",
            "Resource": "resource_ARN"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe"
            ],
            "Resource": "resource_ARN"
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:Describe",
            "Resource": "resource_ARN"
        }
    ]
}
```

Note

Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.

```
"Resource": "*"
```

CodeCatalyst baca saja peran untuk Amazon ECS

Untuk tindakan CodeCatalyst alur kerja, Anda dapat membuat peran IAM dengan izin yang diperlukan. Peran ini menggunakan kebijakan dengan izin terbatas yang CodeCatalyst perlu menjalankan tugas di sumber daya Amazon ECS di Anda. Akun AWS

Peran ini memberikan izin untuk melakukan hal berikut:

- Baca set tugas Amazon ECS.
- Ambil informasi tentang CloudWatch alarm.

Peran ini menggunakan kebijakan berikut:

```
*{*
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ecs:DescribeServices",
                "cloudwatch:DescribeAlarms"
            ],
            "Resource": "resource_ARN",
            "Effect": "Allow"
        },
       {
            "Action": [
                "elasticloadbalancing:DescribeTargetGroups",
                "elasticloadbalancing:DescribeListeners",
                "elasticloadbalancing:DescribeRules"
            ],
            "Resource": "resource_ARN",
            "Effect": "Allow"
        },
       {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "",
            "Condition": {
                "StringEquals": {
                    "s3:ExistingObjectTag/UseWithCodeDeploy": "true"
```

```
                }
            },
            "Effect": "Allow"
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:iam:::role/ecsTaskExecutionRole",
                "arn:aws:iam:::role/ECSTaskExecution"
            ],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": [
                        "ecs-tasks.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

CodeCatalyst baca saja peran untuk Lambda

Untuk tindakan CodeCatalyst alur kerja, Anda dapat membuat peran IAM dengan izin yang diperlukan. Peran ini menggunakan kebijakan dengan izin cakupan yang CodeCatalyst diperlukan untuk menjalankan tugas pada sumber daya Lambda di sumber daya Lambda. Akun AWS

Peran ini memberikan izin untuk hal-hal berikut:

- Baca fungsi dan alias Lambda.
- Akses file revisi di bucket Amazon S3.
- Ambil informasi tentang CloudWatch alarm.

Peran ini menggunakan kebijakan berikut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "cloudwatch:DescribeAlarms",
                "lambda:GetAlias",
                "lambda:GetProvisionedConcurrencyConfig"
            ],
            "Resource": "resource_ARN",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "arn:aws:s3:::/CodeDeploy/",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "",
            "Condition": {
                "StringEquals": {
                    "s3:ExistingObjectTag/UseWithCodeDeploy": "true"
                }
            },
            "Effect": "Allow"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

Membuat peran secara manual untuk tindakan alur kerja

CodeCatalyst Tindakan alur kerja menggunakan peran IAM yang Anda buat disebut peran build, peran penerapan, dan peran tumpukan.

Ikuti langkah-langkah ini untuk membuat peran ini di IAM.

Untuk membuat peran penerapan

1. Buat kebijakan untuk peran tersebut, sebagai berikut:
   a. Masuk ke AWS.
   b. Buka konsol IAM di https://console.aws.amazon.com/iam/.
   c. Di panel navigasi, pilih Kebijakan.
   d. Pilih Buat kebijakan.
   e. Pilih tab JSON.
   f. Hapus kode yang ada.
   g. Tempel kode berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [{
    "Action": [
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack",
        "cloudformation:Describe*",
        "cloudformation:UpdateStack",
        "cloudformation:CreateChangeSet",
        "cloudformation:DeleteChangeSet",
        "cloudformation:ExecuteChangeSet",
```

```
            "cloudformation:SetStackPolicy",
            "cloudformation:ValidateTemplate",
            "cloudformation:List*",
            "iam:PassRole"
        ],
        "Resource": "*",
        "Effect": "Allow"
}]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

h. Pilih Berikutnya: Tanda.

i. Pilih Berikutnya: Tinjau.

j. Dalam Nama, masukkan:

```
codecatalyst-deploy-policy
```

k. Pilih Buat kebijakan.

Anda sekarang telah membuat kebijakan izin.

2. Buat peran deploy, sebagai berikut:

a. Di panel navigasi, pilih Peran, lalu pilih Buat peran.

b. Pilih Kebijakan kepercayaan khusus.

c. Hapus kebijakan kepercayaan kustom yang ada.

d. Tambahkan kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                 ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

e. Pilih Selanjutnya.

f. Di Kebijakan izin, cari `codecatalyst-deploy-policy` dan pilih kotak centang.

g. Pilih Selanjutnya.

h. Untuk nama Peran, masukkan:

```
codecatalyst-deploy-role
```

i. Untuk deskripsi Peran, masukkan:

```
CodeCatalyst deploy role
```

j. Pilih Buat peran.

Anda sekarang telah membuat peran penerapan dengan kebijakan kepercayaan dan kebijakan izin.

3. Dapatkan peran penyebaran ARN, sebagai berikut:

a. Di panel navigasi, pilih Peran.

b. Di kotak pencarian, masukkan nama peran yang baru saja Anda buat (`codecatalyst-deploy-role`).

c. Pilih peran dari daftar.

Halaman Ringkasan peran muncul.

d. Di bagian atas, salin nilai ARN.

Anda sekarang telah membuat peran penerapan dengan izin yang sesuai, dan memperoleh ARN-nya.

Untuk membuat peran build

1. Buat kebijakan untuk peran tersebut, sebagai berikut:

   a. Masuk ke AWS.

   b. Buka konsol IAM di https://console.aws.amazon.com/iam/.

   c. Di panel navigasi, pilih Kebijakan.

   d. Pilih Buat kebijakan.

   e. Pilih tab JSON.

   f. Hapus kode yang ada.

   g. Tempel kode berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [{
    "Action": [
        "s3:PutObject",
        "iam:PassRole"
    ],
    "Resource": "*",
    "Effect": "Allow"
}]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

   h. Pilih Berikutnya: Tanda.

i. Pilih Berikutnya: Tinjau.

j. Dalam Nama, masukkan:

```
codecatalyst-build-policy
```

k. Pilih Buat kebijakan.

Anda sekarang telah membuat kebijakan izin.

2. Buat peran build, sebagai berikut:

a. Di panel navigasi, pilih Peran, lalu pilih Buat peran.

b. Pilih Kebijakan kepercayaan khusus.

c. Hapus kebijakan kepercayaan kustom yang ada.

d. Tambahkan kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                 ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

e. Pilih Selanjutnya.

f. Di Kebijakan izin, cari `codecatalyst-build-policy` dan pilih kotak centang.

g. Pilih Selanjutnya.

h. Untuk nama Peran, masukkan:

```
codecatalyst-build-role
```

i. Untuk deskripsi Peran, masukkan:

```
CodeCatalyst build role
```

j. Pilih Buat peran.

Anda sekarang telah membuat peran build dengan kebijakan kepercayaan dan kebijakan izin.

3. Dapatkan peran membangun ARN, sebagai berikut:

a. Di panel navigasi, pilih Peran.

b. Di kotak pencarian, masukkan nama peran yang baru saja Anda buat (`codecatalyst-build-role`).

c. Pilih peran dari daftar.

Halaman Ringkasan peran muncul.

d. Di bagian atas, salin nilai ARN.

Anda sekarang telah membuat peran build dengan izin yang sesuai, dan memperoleh ARN-nya.

Untuk membuat peran tumpukan

> **Note**
>
> Anda tidak perlu membuat peran tumpukan, meskipun melakukannya disarankan untuk alasan keamanan. Jika Anda tidak membuat peran tumpukan, Anda harus menambahkan kebijakan izin yang dijelaskan lebih lanjut dalam prosedur ini ke peran penerapan.

1. Masuk untuk AWS menggunakan akun tempat Anda ingin menyebarkan tumpukan Anda.
2. Buka konsol IAM di https://console.aws.amazon.com/iam/.
3. Di panel navigasi, pilih Peran. Lalu pilih Buat peran.
4. Di bagian atas, pilih AWS layanan.
5. Dari daftar layanan, pilih CloudFormation.
6. Pilih Berikutnya: Izin.

7. Di kotak pencarian, tambahkan kebijakan apa pun yang diperlukan untuk mengakses sumber daya di tumpukan Anda. Misalnya, jika tumpukan menyertakan AWS Lambda fungsi, Anda perlu menambahkan kebijakan yang memberikan akses ke Lambda.

   > **Tip**
   >
   > Jika Anda tidak yakin kebijakan mana yang harus ditambahkan, Anda dapat menghilangkannya untuk saat ini. Saat Anda menguji tindakan, jika Anda tidak memiliki izin yang tepat, AWS CloudFormation menghasilkan kesalahan yang menunjukkan izin mana yang perlu Anda tambahkan.

8. Pilih Berikutnya: Tanda.
9. Pilih Berikutnya: Tinjau.
10. Untuk nama Peran, masukkan:

    ```
    codecatalyst-stack-role
    ```

11. Pilih Buat peran.
12. Untuk mendapatkan ARN peran tumpukan, lakukan hal berikut:

    a. Di panel navigasi, pilih Peran.

    b. Di kotak pencarian, masukkan nama peran yang baru saja Anda buat (`codecatalyst-stack-role`).

    c. Pilih peran dari daftar.

    d. Pada halaman Ringkasan, salin nilai ARN Peran.

Menggunakan AWS CloudFormation untuk membuat kebijakan dan peran di IAM

Anda dapat memilih untuk membuat dan menggunakan AWS CloudFormation templat untuk membuat kebijakan dan peran yang Anda perlukan untuk mengakses sumber daya dalam CodeCatalyst proyek dan alur kerja Anda. Akun AWS AWS CloudFormation adalah layanan yang membantu Anda memodelkan dan mengatur AWS sumber daya Anda sehingga Anda dapat menghabiskan lebih sedikit waktu mengelola sumber daya tersebut dan lebih banyak waktu untuk berfokus pada aplikasi Anda yang berjalan AWS. Jika Anda berniat untuk membuat peran dalam beberapa Akun AWS, membuat template dapat membantu Anda melakukan tugas ini lebih cepat.

Contoh template berikut membuat peran dan kebijakan tindakan penerapan.

```
Parameters:
  CodeCatalystAccountId:
    Type: String
    Description: Account ID from the connections page
  ExternalId:
    Type: String
    Description: External ID from the connections page
Resources:
  CrossAccountRole:
    Type: 'AWS::IAM::Role'
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: Allow
            Principal:
              AWS:
                - !Ref CodeCatalystAccountId
            Action:
              - 'sts:AssumeRole'
            Condition:
              StringEquals:
                sts:ExternalId: !Ref ExternalId
      Path: /
      Policies:
        - PolicyName: CodeCatalyst-CloudFormation-action-policy
          PolicyDocument:
            Version: "2012-10-17"
            Statement:
              - Effect: Allow
                Action:
                  - 'cloudformation:CreateStack'
                  - 'cloudformation:DeleteStack'
                  - 'cloudformation:Describe*'
                  - 'cloudformation:UpdateStack'
                  - 'cloudformation:CreateChangeSet'
                  - 'cloudformation:DeleteChangeSet'
                  - 'cloudformation:ExecuteChangeSet'
                  - 'cloudformation:SetStackPolicy'
                  - 'cloudformation:ValidateTemplate'
                  - 'cloudformation:List*'
                  - 'iam:PassRole'
                Resource: '*'
```

Membuat peran secara manual untuk cetak biru aplikasi web

Cetak biru aplikasi CodeCatalyst web menggunakan peran IAM yang Anda buat disebut peran build untuk CDK, peran penerapan, dan peran tumpukan.

Ikuti langkah-langkah ini untuk membuat peran dalam IAM.

Untuk membuat peran build

1. Buat kebijakan untuk peran tersebut, sebagai berikut:

    a. Masuk ke AWS.

    b. Buka konsol IAM di https://console.aws.amazon.com/iam/.

    c. Di panel navigasi, pilih Kebijakan.

    d. Pilih Buat Kebijakan.

    e. Pilih tab JSON.

    f. Hapus kode yang ada.

    g. Tempel kode berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudformation:*",
                "ecr:*",
                "ssm:*",
                "s3:*",
                "iam:PassRole",
                "iam:GetRole",
                "iam:CreateRole",
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

h. Pilih Berikutnya: Tanda.

i. Pilih Berikutnya: Tinjau.

j. Dalam Nama, masukkan:

```
codecatalyst-webapp-build-policy
```

k. Pilih Buat kebijakan.

Anda sekarang telah membuat kebijakan izin.

2. Buat peran build, sebagai berikut:

a. Di panel navigasi, pilih Peran, lalu pilih Buat peran.

b. Pilih Kebijakan kepercayaan khusus.

c. Hapus kebijakan kepercayaan kustom yang ada.

d. Tambahkan kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                 ]
            },
            "Action": "sts:AssumeRole"
        }
```

```
    ]
}
```

e. Pilih Selanjutnya.

f. Lampirkan kebijakan izin ke peran build. Pada halaman Tambahkan izin, di bagian Kebijakan izin, cari `codecatalyst-webapp-build-policy` dan pilih kotak centang.

g. Pilih Selanjutnya.

h. Untuk nama Peran, masukkan:

```
codecatalyst-webapp-build-role
```

i. Untuk deskripsi Peran, masukkan:

```
CodeCatalyst Web app build role
```

j. Pilih Buat peran.

Anda sekarang telah membuat peran build dengan kebijakan kepercayaan dan kebijakan izin.

3. Lampirkan kebijakan izin ke peran build, sebagai berikut:

a. Di panel navigasi, pilih Peran, lalu cari`codecatalyst-webapp-build-role`.

b. Pilih `codecatalyst-webapp-build-role` untuk menampilkan detailnya.

c. Di tab Izin, pilih Tambahkan izin, lalu pilih Lampirkan kebijakan.

d. Cari`codecatalyst-webapp-build-policy`, pilih kotak centang, lalu pilih Lampirkan kebijakan.

Anda sekarang telah melampirkan kebijakan izin ke peran build. Peran build sekarang memiliki dua kebijakan: kebijakan izin dan kebijakan kepercayaan.

4. Dapatkan peran membangun ARN, sebagai berikut:

a. Di panel navigasi, pilih Peran.

b. Di kotak pencarian, masukkan nama peran yang baru saja Anda buat (`codecatalyst-webapp-build-role`).

c. Pilih peran dari daftar.

Halaman Ringkasan peran muncul.

d. Di bagian atas, salin nilai ARN.

Anda sekarang telah membuat peran build dengan izin yang sesuai, dan memperoleh ARN-nya.

Membuat peran secara manual untuk cetak biru SAM

Cetak biru CodeCatalyst SAM menggunakan peran IAM yang Anda buat disebut peran build CloudFormation dan peran penerapan untuk SAM.

Ikuti langkah-langkah ini untuk membuat peran di IAM.

Untuk membuat peran build untuk CloudFormation

1. Buat kebijakan untuk peran tersebut, sebagai berikut:

    a. Masuk ke AWS.

    b. Buka konsol IAM di https://console.aws.amazon.com/iam/.

    c. Di panel navigasi, pilih Kebijakan.

    d. Pilih Buat Kebijakan.

    e. Pilih tab JSON.

    f. Hapus kode yang ada.

    g. Tempel kode berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:*",
                "cloudformation:*"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

h. Pilih Berikutnya: Tanda.

i. Pilih Berikutnya: Tinjau.

j. Dalam Nama, masukkan:

```
codecatalyst-SAM-build-policy
```

k. Pilih Buat kebijakan.

Anda sekarang telah membuat kebijakan izin.

2. Buat peran build, sebagai berikut:

a. Di panel navigasi, pilih Peran, lalu pilih Buat peran.

b. Pilih Kebijakan kepercayaan khusus.

c. Hapus kebijakan kepercayaan kustom yang ada.

d. Tambahkan kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                 ]
            },
            "Action": "sts:AssumeRole"
        }
```

```
    ]
}
```

e. Pilih Selanjutnya.

f. Lampirkan kebijakan izin ke peran build. Pada halaman Tambahkan izin, di bagian Kebijakan izin, cari `codecatalyst-SAM-build-policy` dan pilih kotak centang.

g. Pilih Selanjutnya.

h. Untuk nama Peran, masukkan:

```
codecatalyst-SAM-build-role
```

i. Untuk deskripsi Peran, masukkan:

```
CodeCatalyst SAM build role
```

j. Pilih Buat peran.

Anda sekarang telah membuat peran build dengan kebijakan kepercayaan dan kebijakan izin.

3. Lampirkan kebijakan izin ke peran build, sebagai berikut:

   a. Di panel navigasi, pilih Peran, lalu cari`codecatalyst-SAM-build-role`.

   b. Pilih `codecatalyst-SAM-build-role` untuk menampilkan detailnya.

   c. Di tab Izin, pilih Tambahkan izin, lalu pilih Lampirkan kebijakan.

   d. Cari`codecatalyst-SAM-build-policy`, pilih kotak centang, lalu pilih Lampirkan kebijakan.

   Anda sekarang telah melampirkan kebijakan izin ke peran build. Peran build sekarang memiliki dua kebijakan: kebijakan izin dan kebijakan kepercayaan.

4. Dapatkan peran membangun ARN, sebagai berikut:

   a. Di panel navigasi, pilih Peran.

   b. Di kotak pencarian, masukkan nama peran yang baru saja Anda buat (`codecatalyst-SAM-build-role`).

   c. Pilih peran dari daftar.

   Halaman Ringkasan peran muncul.

   d. Di bagian atas, salin nilai ARN.

Anda sekarang telah membuat peran build dengan izin yang sesuai, dan memperoleh ARN-nya.

Untuk membuat peran deploy untuk SAM

1. Buat kebijakan untuk peran tersebut, sebagai berikut:

   a. Masuk ke AWS.

   b. Buka konsol IAM di https://console.aws.amazon.com/iam/.

   c. Di panel navigasi, pilih Kebijakan.

   d. Pilih Buat Kebijakan.

   e. Pilih tab JSON.

   f. Hapus kode yang ada.

   g. Tempel kode berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "iam:PassRole",
                "iam:DeleteRole",
                "iam:GetRole",
                "iam:TagRole",
                "iam:CreateRole",
                "iam:AttachRolePolicy",
                "iam:DetachRolePolicy",
                "cloudformation:*",
                "lambda:*",
                "apigateway:*"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Pertama kali peran digunakan untuk menjalankan tindakan alur kerja, gunakan wildcard dalam pernyataan kebijakan sumber daya dan kemudian cakup kebijakan dengan nama sumber daya setelah tersedia.
>
> ```
> "Resource": "*"
> ```

h. Pilih Berikutnya: Tanda.

i. Pilih Berikutnya: Tinjau.

j. Dalam Nama, masukkan:

```
codecatalyst-SAM-deploy-policy
```

k. Pilih Buat kebijakan.

Anda sekarang telah membuat kebijakan izin.

2. Buat peran build, sebagai berikut:

a. Di panel navigasi, pilih Peran, lalu pilih Buat peran.

b. Pilih Kebijakan kepercayaan khusus.

c. Hapus kebijakan kepercayaan kustom yang ada.

d. Tambahkan kebijakan kepercayaan khusus berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service":  [
                   "codecatalyst-runner.amazonaws.com",
                   "codecatalyst.amazonaws.com"
                 ]
            },
            "Action": "sts:AssumeRole"
        }
```

```
    ]
}
```

e. Pilih Selanjutnya.

f. Lampirkan kebijakan izin ke peran build. Pada halaman Tambahkan izin, di bagian Kebijakan izin, cari `codecatalyst-SAM-deploy-policy` dan pilih kotak centang.

g. Pilih Selanjutnya.

h. Untuk nama Peran, masukkan:

```
codecatalyst-SAM-deploy-role
```

i. Untuk deskripsi Peran, masukkan:

```
CodeCatalyst SAM deploy role
```

j. Pilih Buat peran.

Anda sekarang telah membuat peran build dengan kebijakan kepercayaan dan kebijakan izin.

3. Lampirkan kebijakan izin ke peran build, sebagai berikut:

a. Di panel navigasi, pilih Peran, lalu cari`codecatalyst-SAM-deploy-role`.

b. Pilih `codecatalyst-SAM-deploy-role` untuk menampilkan detailnya.

c. Di tab Izin, pilih Tambahkan izin, lalu pilih Lampirkan kebijakan.

d. Cari`codecatalyst-SAM-deploy-policy`, pilih kotak centang, lalu pilih Lampirkan kebijakan.

Anda sekarang telah melampirkan kebijakan izin ke peran build. Peran build sekarang memiliki dua kebijakan: kebijakan izin dan kebijakan kepercayaan.

4. Dapatkan peran membangun ARN, sebagai berikut:

a. Di panel navigasi, pilih Peran.

b. Di kotak pencarian, masukkan nama peran yang baru saja Anda buat (`codecatalyst-SAM-deploy-role`).

c. Pilih peran dari daftar.

Halaman Ringkasan peran muncul.

d. Di bagian atas, salin nilai ARN.

Anda sekarang telah membuat peran build dengan izin yang sesuai, dan memperoleh ARN-nya.

Validasi kepatuhan untuk Amazon CodeCatalyst

Untuk mempelajari apakah an Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat Layanan AWS di Lingkup oleh Program Kepatuhan Layanan AWS dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat Program AWS Kepatuhan Program AWS .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat Mengunduh Laporan di AWS Artifact .

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Panduan Memulai Cepat Keamanan dan Kepatuhan — Panduan penerapan ini membahas pertimbangan arsitektur dan memberikan langkah-langkah untuk menerapkan lingkungan dasar AWS yang berfokus pada keamanan dan kepatuhan.
- Arsitektur untuk Keamanan dan Kepatuhan HIPAA di Amazon Web Services — Whitepaper ini menjelaskan bagaimana perusahaan dapat menggunakan AWS untuk membuat aplikasi yang memenuhi syarat HIPAA.

  > **Note**
  >
  > Tidak semua memenuhi Layanan AWS syarat HIPAA. Untuk informasi selengkapnya, lihat Referensi Layanan yang Memenuhi Syarat HIPAA.

- AWS Sumber Daya AWS — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- AWS Panduan Kepatuhan Pelanggan - Memahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan ini merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan untuk kontrol keamanan di berbagai kerangka kerja (termasuk Institut Standar dan Teknologi Nasional (NIST), Dewan Standar Keamanan Industri Kartu Pembayaran (PCI), dan Organisasi Internasional untuk Standardisasi (ISO)).

- Mengevaluasi Sumber Daya dengan Aturan dalam Panduan AWS Config Pengembang — AWS Config Layanan menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- AWS Security Hub— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS. Security Hub menggunakan kontrol keamanan untuk sumber daya AWS Anda serta untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat Referensi kontrol Security Hub.
- Amazon GuardDuty — Ini Layanan AWS mendeteksi potensi ancaman terhadap beban kerja Akun AWS, kontainer, dan data Anda dengan memantau lingkungan Anda untuk aktivitas yang mencurigakan dan berbahaya. GuardDuty dapat membantu Anda mengatasi berbagai persyaratan kepatuhan, seperti PCI DSS, dengan memenuhi persyaratan deteksi intrusi yang diamanatkan oleh kerangka kerja kepatuhan tertentu.
- AWS Audit ManagerIni Layanan AWS membantu Anda terus mengaudit AWS penggunaan Anda untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap peraturan dan standar industri.

Ketahanan di Amazon CodeCatalyst

Infrastruktur AWS global dibangun di sekitar Wilayah AWS dan Availability Zones. Wilayah memberikan beberapa Zona Ketersediaan yang terpisah dan terisolasi secara fisik, yang terkoneksi melalui jaringan latensi rendah, throughput tinggi, dan sangat redundan. Dengan Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan fail over di antara zona tanpa gangguan. Availability Zone memiliki ketersediaan yang lebih baik, toleran terhadap kegagalan, dan dapat diukur skalanya jika dibandingkan dengan satu atau beberapa infrastruktur pusat data tradisional.

Untuk informasi selengkapnya tentang Wilayah AWS dan Availability Zone, lihat Infrastruktur Global AWS. Untuk mempelajari lebih lanjut tentang CodeCatalyst data apa yang direplikasiWilayah AWS, lihatPerlindungan data di Amazon CodeCatalyst.

Keamanan Infrastruktur di Amazon CodeCatalyst

Sebagai layanan terkelola, Amazon CodeCatalyst dilindungi oleh keamanan jaringan AWS global. Untuk informasi tentang layanan AWS keamanan dan cara AWS melindungi infrastruktur, lihat Keamanan AWS Cloud. Untuk mendesain AWS lingkungan Anda menggunakan praktik

terbaik untuk keamanan infrastruktur, lihat Perlindungan Infrastruktur dalam Kerangka Kerja yang AWSDiarsiteksikan dengan Baik Pilar Keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses CodeCatalyst melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Transportasi (TLS). Kami membutuhkan TLS 1.2 dan merekomendasikan TLS 1.3.
- Cipher suite dengan perfect forward secrecy (PFS) seperti DHE (Ephemeral Diffie-Hellman) atau ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Sebagian besar sistem modern seperti Java 7 dan sistem yang lebih baru mendukung mode ini.

Selain itu, permintaan harus ditandatangani menggunakan access key ID dan secret access key yang terkait dengan principal IAM. Atau Anda bisa menggunakan AWS Security Token Service (AWS STS) untuk membuat kredensial keamanan sementara guna menandatangani permintaan.

Analisis konfigurasi dan kerentanan di Amazon CodeCatalyst

Konfigurasi dan kontrol IT merupakan tanggung jawab bersama antara AWS dan Anda, pelanggan kami. Untuk informasi selengkapnya, lihat model tanggung jawab bersama AWS.

Data dan privasi Anda di Amazon CodeCatalyst

Amazon CodeCatalyst menjaga privasi Anda dengan serius, dan keamanan informasi Anda adalah prioritas utama kami. Anda dapat meninjau lebih lanjut tentang cara kami menangani informasi Anda dalam Pemberitahuan AWS Privasi.

Untuk meminta dan melihat data Anda, lihat Meminta data Anda di. Referensi Umum AWS

Menghapus profil AWS Builder ID

Menghapus profil Anda adalah tindakan permanen yang tidak dapat dibalik. Proses penghapusan dimulai segera setelah Anda memilih Hapus. Amazon CodeCatalyst mulai menghapus profil Anda dan semua informasi pribadi terkait. Proses ini bisa memakan waktu hingga 90 hari untuk menyelesaikannya.

Ketika profil Anda dihapus, Anda tidak dapat mengakses atau memulihkan data Anda di Amazon CodeCatalyst. Ini termasuk Token Akses Pribadi, Peran, Keanggotaan Pengguna, dan setiap

CodeCatalyst ruang Amazon di mana Anda adalah satu-satunya anggota. Anda tidak dapat lagi masuk ke Amazon CodeCatalyst.

Untuk informasi tentang cara menghapus profil AWS Builder ID Anda, lihat Menghapus ID AWS Builder Anda diReferensi Umum AWS.

Praktik terbaik untuk tindakan alur kerja di Amazon CodeCatalyst

Ada sejumlah praktik terbaik keamanan yang perlu dipertimbangkan saat Anda mengembangkan alur kerja Anda. CodeCatalyst Berikut ini adalah pedoman umum dan tidak mewakili solusi keamanan yang lengkap. Karena praktik terbaik ini mungkin tidak sesuai atau cukup untuk lingkungan Anda, anggap praktik terbaik tersebut sebagai pertimbangan yang membantu dan bukan sebagai rekomendasi.

Topik

Informasi sensitif

Jangan menyematkan informasi sensitif di YAMAL Anda. Daripada menyematkan kredensil, kunci, atau token di YAMAL Anda, kami sarankan Anda menggunakan rahasia. CodeCatalyst Rahasia menyediakan cara mudah untuk menyimpan dan mereferensikan informasi sensitif dari dalam YAMAL Anda.

Ketentuan lisensi

Pastikan untuk memperhatikan ketentuan lisensi dari tindakan yang Anda pilih untuk digunakan.

Kode tidak tepercaya

Tindakan umumnya mandiri, modul tujuan tunggal yang dapat dibagi di seluruh proyek, ruang, atau komunitas yang lebih luas. Menggunakan kode dari orang lain dapat menjadi keuntungan kenyamanan dan efisiensi yang luar biasa, tetapi juga memperkenalkan vektor ancaman baru. Tinjau bagian berikut untuk memastikan Anda mengikuti praktik terbaik untuk menjaga alur kerja CI/CD Anda tetap aman.

GitHub Tindakan

GitHub Tindakan bersifat open source, dibangun dan dipelihara oleh masyarakat. Kami mengikuti model tanggung jawab bersama dan mempertimbangkan kode sumber GitHub Tindakan sebagai data pelanggan yang menjadi tanggung jawab Anda. GitHub Tindakan dapat diberikan akses ke rahasia, token repositori, kode sumber, tautan akun, dan waktu komputasi Anda. Pastikan Anda yakin dengan kepercayaan dan keamanan GitHub Tindakan yang Anda rencanakan untuk dijalankan.

Panduan yang lebih spesifik dan praktik terbaik keamanan untuk GitHub Tindakan:

- Pengerasan keamanan
- Mencegah permintaan pwn
- Masukan tidak tepercaya
- Bagaimana mempercayai blok bangunan Anda

Memahami model CodeCatalyst kepercayaan

Model CodeCatalyst kepercayaan Amazon memungkinkan CodeCatalyst untuk mengambil peran layanan dalam terhubung Akun AWS. Model menghubungkan peran IAM, prinsip CodeCatalyst layanan, dan ruang. CodeCatalyst Kebijakan kepercayaan menggunakan kunci `aws:SourceArn` kondisi untuk memberikan izin ke CodeCatalyst ruang yang ditentukan dalam kunci kondisi. Untuk informasi selengkapnya tentang kunci kondisi ini, lihat aws: SourceArn di Panduan Pengguna IAM.

Kebijakan kepercayaan adalah dokumen kebijakan JSON di mana Anda menentukan prinsip yang Anda percayai untuk mengambil peran tersebut. Kebijakan kepercayaan adalah kebijakan berbasis sumber daya yang diperlukan yang melekat pada peran dalam IAM. Untuk informasi selengkapnya, lihat Syarat dan konsep di Panduan Pengguna IAM. Untuk detail tentang prinsip layanan, lihat. CodeCatalyst Prinsipal layanan untuk CodeCatalyst

Dalam kebijakan kepercayaan berikut, prinsip layanan yang tercantum dalam `Principal` elemen diberikan izin dari kebijakan berbasis sumber daya, dan `Condition` blok tersebut digunakan untuk membatasi akses ke sumber daya cakupan bawah.

```
"Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Principal": {
                "Service": [
                    "codecatalyst-runner.amazonaws.com",
                    "codecatalyst.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "ArnLike": {
                    "aws:SourceArn": "arn:aws:codecatalyst:::space/spaceId/project/*"
                }
            }
        }
    ]
```

Dalam kebijakan kepercayaan, prinsipal CodeCatalyst layanan diberikan akses melalui kunci `aws:SourceArn` kondisi, yang berisi Nama Sumber Daya Amazon (ARN) untuk ID spasi. CodeCatalyst ARN menggunakan format berikut:

```
arn:aws:codecatalyst:::space/spaceId/project/*
```

> **Important**
>
> Gunakan ID spasi hanya di tombol kondisi, seperti`aws:SourceArn`. Jangan gunakan ID spasi dalam pernyataan kebijakan IAM sebagai ARN sumber daya.

Sebagai praktik terbaik, kurangi izin sebanyak mungkin dalam kebijakan.

- Anda dapat menggunakan wildcard (*) di kunci `aws:SourceArn` kondisi untuk menentukan semua proyek di ruang dengan. `project/*`
- Anda dapat menentukan izin tingkat sumber daya dalam kunci `aws:SourceArn` kondisi untuk proyek tertentu di ruang dengan. `project/projectId`

Prinsipal layanan untuk CodeCatalyst

Anda menggunakan `Principal` elemen dalam kebijakan JSON berbasis sumber daya untuk menentukan prinsipal yang diizinkan atau ditolak akses ke sumber daya. Prinsipal yang dapat Anda sebutkan dalam kebijakan kepercayaan termasuk pengguna, peran, akun, dan layanan. Anda tidak

dapat menggunakan `Principal` elemen dalam kebijakan berbasis identitas; demikian pula, Anda tidak dapat mengidentifikasi grup pengguna sebagai prinsipal dalam kebijakan (seperti kebijakan berbasis sumber daya) karena grup terkait dengan izin, bukan otentikasi, dan prinsipal adalah entitas IAM yang diautentikasi.

Dalam kebijakan kepercayaan, Anda dapat menentukan Layanan AWS `Principal` elemen kebijakan berbasis sumber daya atau dalam kunci kondisi yang mendukung prinsipal. Prinsipal layanan ditentukan oleh layanan. Berikut ini adalah prinsip layanan yang didefinisikan untuk: CodeCatalyst

- codecatalyst.amazonaws.com - Prinsip layanan ini digunakan untuk peran yang akan memberikan akses ke. CodeCatalyst AWS
- codecatalyst-runner.amazonaws.com - Prinsip layanan ini digunakan untuk peran yang akan memberikan akses ke sumber daya dalam penerapan untuk alur kerja. CodeCatalyst AWS CodeCatalyst

Untuk informasi selengkapnya, lihat elemen kebijakan AWS JSON: Principal dalam Panduan Pengguna IAM.

Memantau peristiwa dan panggilan API menggunakan logging

Di Amazon CodeCatalyst, acara manajemen untuk ruang dikumpulkan oleh AWS CloudTrail dan dicatat di jejak untuk akun penagihan untuk ruang tersebut. CloudTrail logging adalah metode utama untuk mengelola logging untuk CodeCatalyst peristiwa, dan metode sekunder adalah melihat peristiwa masuk CodeCatalyst.

Peristiwa di akun dicatat dengan jejak dan ember yang ditunjuk yang disiapkan untuk Akun AWS.

Diagram berikut menunjukkan bagaimana semua peristiwa manajemen untuk ruang masuk CloudTrail untuk akun penagihan, sementara koneksi akun/peristiwa penagihan dan peristiwa AWS sumber daya masuk CloudTrail untuk akun masing-masing.

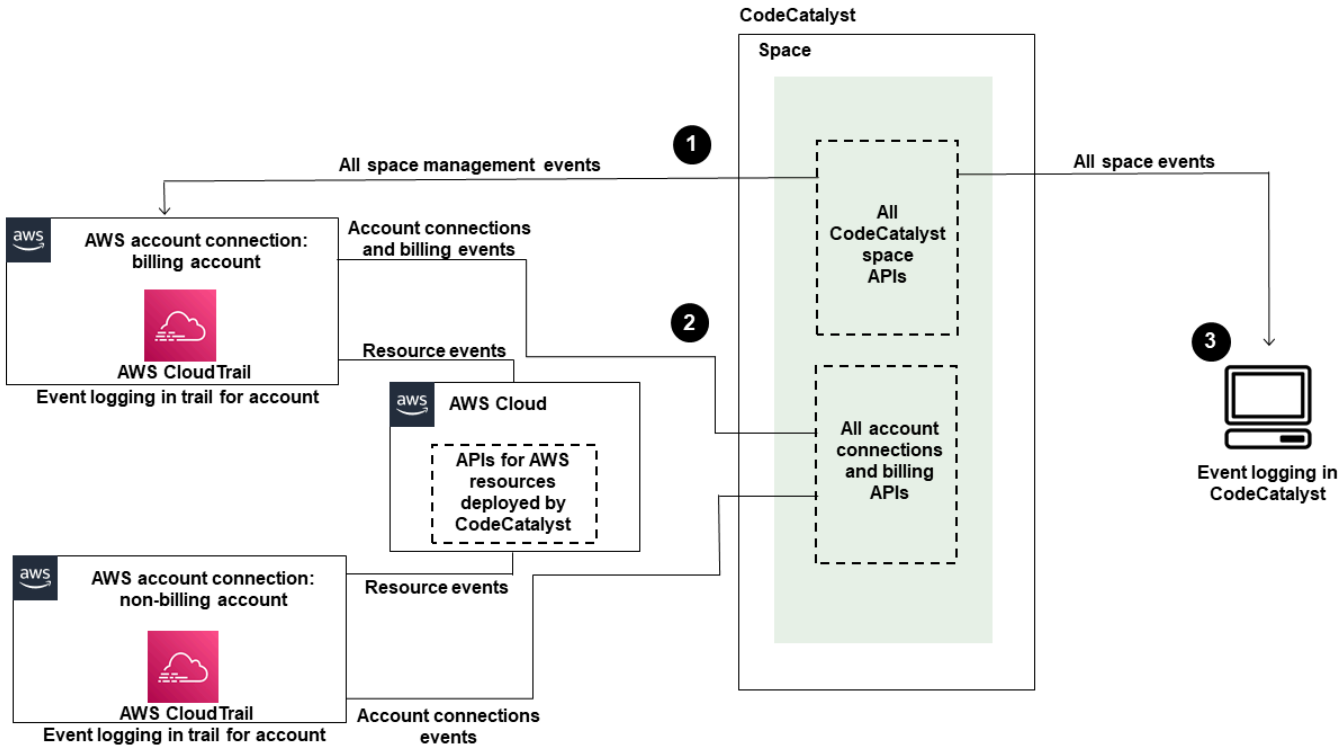


Diagram ini menggambarkan langkah-langkah berikut:

1. Ketika ruang dibuat, sebuah Akun AWS terhubung ke ruang dan ditetapkan sebagai akun penagihan. Jejak yang digunakan adalah jejak yang dibuat CloudTrail untuk akun penagihan, tempat peristiwa luar angkasa dicatat. CloudTrail menangkap panggilan API dan peristiwa terkait yang dibuat oleh atau atas nama CodeCatalyst spasi dan mengirimkan file log ke bucket S3 yang Anda tentukan. Jika akun penagihan berubah ke akun AWS lain, maka peristiwa luar angkasa dicatat di jejak dan bucket untuk akun tersebut. Untuk informasi selengkapnya tentang peristiwa CodeCatalyst manajemen yang dicatat oleh CloudTrail, lihatCodeCatalyst informasi di CloudTrail.
2. Akun lain yang terhubung ke ruang, termasuk akun penagihan, mencatat subset peristiwa untuk koneksi akun dan peristiwa penagihan. CodeCatalyst alur kerja yang menghasilkan peristiwa akun untuk AWS sumber daya yang digunakan untuk akun tersebut juga dicatat di jejak dan keranjang untuk. Akun AWS CloudTrail menangkap panggilan API dan peristiwa terkait yang dibuat oleh atau atas nama CodeCatalyst spasi dan mengirimkan file log ke bucket S3 yang Anda tentukan. Untuk informasi selengkapnya tentang peristiwa CodeCatalyst manajemen yang dicatat oleh CloudTrail, lihatMengakses peristiwa yang dicatat menggunakan pencatatan peristiwa.
3. Anda juga dapat memantau CodeCatalyst tindakan di ruang Anda dalam waktu tertentu di ruang dengan list-event-logsperintah menggunakan AWS CLI. Untuk informasi selengkapnya, lihat

Panduan Referensi Amazon CodeCatalyst API. Anda harus memiliki peran administrator Space untuk memanggil daftar acara untuk CodeCatalyst tindakan di ruang Anda. Untuk informasi selengkapnya, lihat Mengakses peristiwa yang dicatat menggunakan pencatatan peristiwa.

> **Note**
>
> `ListEventLogs`menjamin acara selama 30 hari terakhir di ruang tertentu. Anda juga dapat melihat dan mengambil daftar acara manajemen selama 90 hari terakhir CodeCatalyst di AWS CloudTrail konsol dengan melihat riwayat Acara, atau dengan membuat jejak untuk membuat dan memelihara catatan peristiwa yang berlangsung selama 90 hari terakhir. Untuk informasi selengkapnya, lihat Bekerja dengan riwayat CloudTrail Acara dan Bekerja dengan CloudTrail jalur.

> **Note**
>
> AWS sumber daya yang digunakan ke akun yang terhubung untuk CodeCatalyst alur kerja, tidak dicatat sebagai bagian dari CloudTrail pencatatan ruang. CodeCatalyst Misalnya, CodeCatalyst sumber daya termasuk ruang atau proyek. AWS sumber daya termasuk layanan Amazon ECS atau fungsi Lambda. Anda harus mengonfigurasi CloudTrail logging secara terpisah untuk setiap Akun AWS tempat sumber daya digunakan.

Berikut adalah salah satu aliran yang mungkin untuk pemantauan acara di CodeCatalyst.

Mary Major adalah administrator Space untuk CodeCatalyst ruang dan melihat semua peristiwa manajemen CodeCatalyst untuk sumber daya tingkat ruang dan tingkat proyek di ruang yang masuk. CloudTrail Lihat CodeCatalyst informasi di CloudTrail misalnya peristiwa yang masuk CloudTrail.

Untuk sumber daya yang dibuat di CodeCatalyst, seperti Lingkungan Dev, Mary melihat riwayat Acara di akun penagihan untuk ruang tersebut dan menyelidiki peristiwa di mana Lingkungan Pengembang dibuat oleh anggota proyek. CodeCatalyst Acara ini menyediakan tipe identitas IAM penyimpanan identitas dan kredensional untuk ID AWS Pembangun bagi pengguna yang membuat Lingkungan Pengembang. Untuk sumber daya yang dibuat AWS saat digunakan oleh alur kerja di CodeCatalyst, seperti fungsi Lambda untuk penerapan tanpa server, Akun AWS pemilik dapat melihat riwayat peristiwa untuk jejak yang terkait dengan terpisah Akun AWS (yang juga merupakan akun yang terhubung ke) untuk CodeCatalyst tindakan penerapan alur kerja.

Untuk menyelidiki lebih lanjut, Mary juga dapat melihat peristiwa untuk semua CodeCatalyst API di ruang dengan menggunakan list-event-logsperintah di AWS CLI.

Topik

Memantau panggilan API dengan Akun AWS menggunakan AWS CloudTrail logging

Amazon CodeCatalyst terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau Layanan AWS. CloudTrail menangkap panggilan API yang dibuat atas nama CodeCatalyst dalam terhubung Akun AWS sebagai peristiwa. Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara berkelanjutan ke bucket S3, termasuk acara untuk CodeCatalyst. Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru di CloudTrail konsol dalam Riwayat acara.

CodeCatalyst mendukung pencatatan tindakan berikut sebagai peristiwa dalam file CloudTrail log:

- Acara manajemen untuk CodeCatalyst spasi akan dicatat di Akun AWS yang merupakan akun penagihan yang ditunjuk untuk ruang tersebut. Untuk informasi selengkapnya, lihat CodeCatalyst acara luar angkasa.

  > **Note**
  >
  > Peristiwa data untuk CodeCatalyst spasi dapat diakses dengan menggunakan CLI seperti yang dijelaskan dalam. Mengakses peristiwa yang dicatat menggunakan pencatatan peristiwa

- Peristiwa untuk sumber daya yang digunakan dalam tindakan CodeCatalyst alur kerja yang terjadi dalam koneksi Akun AWS akan dicatat sebagai peristiwa di dalamnya. Akun AWS Untuk informasi selengkapnya, lihat CodeCatalyst koneksi akun dan acara penagihan.

> **Important**
>
> Meskipun beberapa akun dapat dikaitkan dengan spasi, CloudTrail pencatatan untuk acara di CodeCatalyst spasi dan proyek hanya berlaku untuk akun penagihan.

Akun penagihan ruang adalah akun Anda Akun AWS yang dikenakan biaya untuk CodeCatalyst sumber daya di luar tingkat AWS Gratis. Beberapa akun dapat dihubungkan ke spasi, sementara hanya satu akun yang dapat menjadi akun penagihan yang ditunjuk. Akun penagihan atau akun tambahan yang terhubung untuk ruang tersebut dapat memiliki peran IAM yang digunakan untuk menyebarkan AWS sumber daya dan infrastruktur, seperti cluster Amazon ECS atau bucket S3, dari alur kerja. CodeCatalyst Anda dapat menggunakan alur kerja YAMAL untuk mengidentifikasi Akun AWS yang Anda gunakan.

> **Note**
>
> AWS sumber daya yang digunakan ke akun yang terhubung untuk CodeCatalyst alur kerja, tidak dicatat sebagai bagian dari CloudTrail pencatatan ruang. CodeCatalyst Misalnya, CodeCatalyst sumber daya termasuk ruang atau proyek. AWS sumber daya termasuk layanan Amazon ECS atau fungsi Lambda. CloudTrail logging harus dikonfigurasi secara terpisah untuk masing-masing Akun AWS tempat sumber daya digunakan.

CodeCatalyst masuk ke akun yang terhubung mencakup pertimbangan berikut:

- Akses ke CloudTrail acara dikelola dengan IAM di akun yang terhubung dan bukan di CodeCatalyst.
- Koneksi pihak ketiga, seperti menautkan ke GitHub repositori, akan mengakibatkan nama sumber daya pihak ketiga dicatat dalam log. CloudTrail

> **Note**
>
> CloudTrail logging untuk CodeCatalyst acara berada pada tingkat ruang dan tidak mengisolasi peristiwa berdasarkan batas proyek.

Untuk informasi selengkapnya CloudTrail, lihat Panduan AWS CloudTrail Pengguna.

> **Note**
>
> Bagian ini menjelaskan CloudTrail pencatatan untuk semua peristiwa yang dicatat dalam CodeCatalyst ruang masuk dan Akun AWS yang terhubung ke CodeCatalyst. Selain itu, untuk meninjau semua peristiwa yang masuk dalam CodeCatalyst spasi, Anda juga dapat menggunakan AWS CLI dan aws codecatalyst list-event-logs perintah. Untuk informasi selengkapnya, lihat Mengakses peristiwa yang dicatat menggunakan pencatatan peristiwa.

CodeCatalyst acara luar angkasa

Tindakan CodeCatalyst untuk mengelola sumber daya tingkat ruang dan tingkat proyek dicatat di akun penagihan untuk ruang tersebut. Untuk CloudTrail pencatatan CodeCatalyst ruang, peristiwa dicatat dengan pertimbangan berikut.

- CloudTrail peristiwa berlaku di seluruh ruang dan tidak tercakup ke proyek tunggal mana pun.
- Saat Anda menghubungkan Akun AWS ke CodeCatalyst spasi, peristiwa yang dapat dicatat untuk koneksi akun akan masuk ke dalamnya. Akun AWS Setelah Anda mengaktifkan koneksi ini, Anda tidak dapat menonaktifkannya.
- Saat Anda menghubungkan Akun AWS ke CodeCatalyst spasi dan menetapkannya sebagai akun penagihan untuk ruang tersebut, acara akan masuk ke dalamnya. Akun AWS Setelah Anda mengaktifkan koneksi ini, Anda tidak dapat menonaktifkannya.

  Peristiwa untuk sumber daya tingkat ruang dan tingkat proyek hanya dicatat di akun penagihan. Untuk mengubah akun CloudTrail tujuan, perbarui akun penagihan di CodeCatalyst. Pada awal siklus penagihan bulanan berikutnya, perubahan berlaku untuk akun penagihan baru di. CodeCatalyst Setelah itu, akun CloudTrail tujuan diperbarui.

Berikut ini adalah contoh peristiwa AWS yang terkait dengan tindakan CodeCatalyst untuk mengelola sumber daya tingkat ruang dan tingkat proyek. API berikut dirilis melalui SDK dan CLI. Acara akan dicatat dalam yang Akun AWS ditentukan sebagai akun penagihan untuk CodeCatalyst ruang tersebut.

- `CreateDevEnvironment`
- `CreateProject`
- `DeleteDevEnvironment`
- `GetDevEnvironment`

- GetProject
- GetSpace
- GetSubscription
- ListDevEnvironments
- ListDevEnvironmentSessions
- ListEventLogs
- ListProjects
- ListSourceRepositories
- StartDevEnvironment
- StartDevEnvironmentSession
- StopDevEnvironment
- StopDevEnvironmentSession
- UpdateDevEnvironment

CodeCatalyst koneksi akun dan acara penagihan

Berikut ini adalah contoh peristiwa AWS yang terkait dengan tindakan untuk koneksi akun atau penagihan: CodeCatalyst

- AcceptConnection
- AssociateIAMRoletoConnection
- DeleteConnection
- DissassociateIAMRolefromConnection
- GetBillingAuthorization
- GetConnection
- GetPendingConnection
- ListConnections
- ListIAMRolesforConnection
- PutBillingAuthorization
- RejectConnection

CodeCatalyst informasi di CloudTrail

CloudTrail diaktifkan pada Akun AWS saat Anda membuat akun itu. Saat Anda menghubungkannya Akun AWS ke CodeCatalyst spasi, peristiwa untuk ruang yang terjadi di dalamnya masuk CloudTrail log di akun AWS tersebut. Akun AWS Peristiwa yang dapat dicatat dicatat sebagai CloudTrail peristiwa dalam CloudTrail log di akun yang terhubung dan dalam riwayat Acara di CloudTrail konsol, bersama dengan AWS peristiwa lain yang dapat dicatat di akun tersebut. CodeCatalyst

Setiap entri peristiwa atau log berisi informasi tentang siapa yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan berikut ini:

- Apakah permintaan dibuat oleh pengguna dengan AWS Builder ID mereka.
- Apakah permintaan itu dibuat dengan kredenal pengguna root atau AWS Identity and Access Management (IAM).
- Apakah permintaan tersebut dibuat dengan kredensial keamanan sementara untuk satu peran atau pengguna terfederasi.
- Apakah permintaan itu dibuat oleh AWS layanan lain.

Untuk informasi selengkapnya, lihat elemen CloudTrail UserIdentity.

Mengakses acara CloudTrail

Untuk catatan acara yang sedang berlangsung di Anda Akun AWS, termasuk acara untuk CodeCatalyst aktivitas di Akun AWS, buat jejak. Jejak memungkinkan CloudTrail untuk mengirimkan file log ke bucket S3. Secara default, saat Anda membuat jejak di konsol, jejak tersebut berlaku untuk semua Wilayah AWS. Jejak mencatat peristiwa dari semua Wilayah di AWS partisi dan mengirimkan file log ke bucket S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasi AWS layanan lain untuk menganalisis lebih lanjut dan menindaklanjuti data peristiwa yang dikumpulkan dalam CloudTrail log. Untuk informasi selengkapnya, lihat berikut:

- Gambaran umum untuk membuat jejak
- CloudTrail layanan dan integrasi yang didukung
- Mengonfigurasi notifikasi Amazon SNS untuk CloudTrail
- Menerima file CloudTrail log dari beberapa wilayah dan Menerima file CloudTrail log dari beberapa akun

Trail adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai file log ke bucket S3 yang Anda tentukan. CloudTrail file log berisi satu atau lebih entri log. Peristiwa mewakili permintaan tunggal dari sumber manapun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. CloudTrail file log bukanlah jejak tumpukan yang diurutkan dari panggilan API publik, sehingga file tersebut tidak muncul dalam urutan tertentu.

Contoh acara koneksi CodeCatalyst akun di AWS

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan `ListConnections` tindakan. Untuk Akun AWS yang terhubung ke ruang, `ListConnections` digunakan untuk melihat semua koneksi akun CodeCatalyst untuk ini Akun AWS. Acara akan dicatat dalam yang Akun AWS ditentukan`accountId`, dan nilai `arn` akan menjadi Nama Sumber Daya Amazon (ARN) dari peran yang digunakan untuk tindakan tersebut.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AKIAI44QH8DHBEXAMPLE",
        "arn": "role-ARN",
        "accountId": "account-ID",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AKIAI44QH8DHBEXAMPLE",
                "arn": "role-ARN",
                "accountId": "account-ID",
                "userName": "user-name"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-09-06T15:04:31Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-09-06T15:08:43Z",
    "eventSource": "account-ID",
    "eventName": "ListConnections",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.168.0.1",
```

```
    "userAgent": "aws-cli/1.18.147 Python/2.7.18 Linux/5.4.207-126.363.amzn2int.x86_64
 botocore/1.18.6",
    "requestParameters": null,
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 ",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 ",
    "readOnly": true,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-ID",
    "eventCategory": "Management"
}
```

Contoh acara sumber daya CodeCatalyst proyek di AWS

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan `CreateDevEnvironment` tindakan. An Akun AWS yang terhubung ke ruang dan merupakan akun penagihan yang ditunjuk untuk ruang tersebut digunakan untuk acara tingkat proyek di ruang tersebut, seperti membuat Lingkungan Pengembang.

Di bawah`userIdentity`, di `accountId` bidang, ini adalah ID akun Pusat Identitas IAM (`432677196278`) yang menghosting kumpulan identitas untuk semua identitas ID AWS Pembangun. ID akun ini berisi informasi berikut tentang CodeCatalyst pengguna untuk acara tersebut.

- `type`Bidang menunjukkan jenis entitas IAM untuk permintaan. Untuk CodeCatalyst acara untuk ruang dan sumber daya proyek, nilai ini adalah`IdentityCenterUser`. `accountId`Bidang menentukan akun yang memiliki entitas yang digunakan untuk mendapatkan kredensil.
- `userId`Kolom berisi pengenal AWS Builder ID untuk pengguna.
- `identityStoreArn`Bidang berisi peran ARN untuk akun penyimpanan identitas dan pengguna.

`recipientAccountId`Kolom berisi ID akun untuk akun penagihan untuk spasi, dengan nilai contoh di sini 111122223333.

Untuk informasi selengkapnya, lihat elemen CloudTrail UserIdentity.

```
{
 "eventVersion": "1.09",
 "userIdentity": {
  "type": "IdentityCenterUser",
```

```
  "accountId": "432677196278",
  "onBehalfOf": {
   "userId": "user-ID",
   "identityStoreArn": "arn:aws:identitystore::432677196278:identitystore/d-9067642ac7"
  },
  "credentialId": "ABCDefGhiJKLMn11Lmn_1AbCDEFgHijk-AaBCdEFGHIjKLmnOPqrs11abEXAMPLE"
 },
 "eventTime": "2023-05-18T17:10:50Z",
 "eventSource": "codecatalyst.amazonaws.com",
 "eventName": "CreateDevEnvironment",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "192.168.0.1",
 "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101
 Firefox/102.0",
 "requestParameters": {
  "spaceName": "MySpace",
  "projectName": "MyProject",
  "ides": [{
   "runtime": "public.ecr.aws/q6e8p2q0/cloud9-ide-runtime:2.5.1",
   "name": "Cloud9"
  }],
  "instanceType": "dev.standard1.small",
  "inactivityTimeoutMinutes": 15,
  "persistentStorage": {
   "sizeInGiB": 16
  }
 },
 "responseElements": {
  "spaceName": "MySpace",
  "projectName": "MyProject",
  "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111 "
 },
 "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
 "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
 "readOnly": false,
 "eventType": "AwsApiCall",
 "managementEvent": true,
 "recipientAccountId": "111122223333",
 "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
 "eventCategory": "Management"
}
```

> **Note**
>
> Dalam peristiwa tertentu, agen pengguna mungkin tidak diketahui. Dalam hal ini, CodeCatalyst akan memberikan nilai `Unknown` di `userAgent` lapangan dalam CloudTrail acara tersebut.

Menanyakan jejak CodeCatalyst acara Anda

Anda dapat membuat dan mengelola kueri untuk CloudTrail log Anda menggunakan tabel kueri di Amazon Athena. Untuk informasi selengkapnya tentang membuat kueri, lihat Menanyakan AWS CloudTrail log di Panduan Pengguna Amazon Athena.

Mengakses peristiwa yang dicatat menggunakan pencatatan peristiwa

Saat pengguna melakukan tindakan di Amazon CodeCatalyst, tindakan ini dicatat sebagai peristiwa. Anda dapat menggunakan AWS CLI untuk melihat log peristiwa dalam spasi dalam jangka waktu tertentu. Anda dapat melihat peristiwa ini untuk meninjau tindakan yang diambil dalam ruang, termasuk tanggal dan waktu tindakan, nama pengguna yang melakukan tindakan, dan alamat IP tempat pengguna membuat permintaan.

> **Note**
>
> Acara manajemen untuk CodeCatalyst ruang masuk CloudTrail untuk akun penagihan yang terhubung. Untuk informasi selengkapnya tentang peristiwa CodeCatalyst manajemen yang dicatat oleh CloudTrail, lihatCodeCatalyst informasi di CloudTrail.

Untuk melihat log peristiwa untuk suatu ruang, Anda harus menginstal dan mengkonfigurasi AWS CLI dengan profil untuk CodeCatalyst, dan Anda harus memiliki peran administrator Space untuk ruang tersebut. Untuk informasi selengkapnya, lihat Menyiapkan untuk menggunakan AWS CLI dengan CodeCatalyst dan Peran administrator ruang.

> **Note**
>
> Untuk melihat pencatatan peristiwa yang terjadi atas nama yang terhubung Akun AWS, atau untuk melihat pencatatan peristiwa untuk ruang atau sumber daya proyek di akun penagihan yang terhubung, Anda dapat menggunakannya AWS CloudTrail. CodeCatalyst Untuk

informasi selengkapnya, lihat Memantau panggilan API dengan Akun AWS menggunakan AWS CloudTrail logging.

1. Buka terminal atau baris perintah dan jalankan aws codecatalyst list-event-logs perintah, dengan menentukan:

   - Nama spasi dengan **`--space-name`** opsi.
   - Tanggal dan waktu ketika Anda ingin mulai meninjau peristiwa, dalam format stempel waktu universal terkoordinasi (UTC) seperti yang ditentukan dalam RFC 3339, dengan opsi. **`--start-time`**
   - Tanggal dan waktu ketika Anda ingin berhenti meninjau peristiwa, dalam format stempel waktu universal terkoordinasi (UTC) sebagaimana ditentukan dalam RFC 3339, dengan opsi. **`--end-time`**
   - (Opsional) Jumlah maksimum hasil untuk kembali dalam satu respons, dengan **`--max-results`** opsi. Jika jumlah hasil lebih besar dari jumlah yang Anda tentukan, respons akan menyertakan `nextToken` elemen yang dapat Anda gunakan untuk mengembalikan hasil berikutnya.
   - (Opsional) Batasi hasil ke jenis acara tertentu yang ingin Anda kembalikan, dengan **`--event-name`** opsi.

   Contoh ini mengembalikan peristiwa yang dicatat dalam ruang yang dinamai *`ExampleCorp`*dari periode waktu *`2022-11-30`* hingga *`2022-12-01`*, dan maksimal *`2`* peristiwa dikembalikan sebagai respons.

   ```
   aws codecatalyst list-event-logs --space-name ExampleCorp --start-time 2022-11-30
    --end-time 2022-12-01 --event-name list-event-logs --max-results 2
   ```

2. Jika peristiwa terjadi dalam kerangka waktu ini, perintah mengembalikan hasil yang mirip dengan berikut ini:

   ```
   {
       "nextToken": "EXAMPLE",
       "items": [
           {
               "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
               "eventName": "listEventLogs",
   ```

```
            "eventType": "AwsApiCall",
            "eventCategory": "MANAGEMENT",
            "eventSource": "manage",
            "eventTime": "2022-12-01T22:47:24.605000+00:00",
            "operationType": "READONLY",
            "userIdentity": {
                "userType": "USER",
                "principalId": "a1b2c3d4e5-678fgh90-1a2b-3c4d-e5f6-EXAMPLE11111"
                "userName": "MaryMajor"
            },
            "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
            "requestPayload": {
                "contentType": "application/json",
                "data": "{\"spaceName\":\"ExampleCorp\",\"startTime\":
\"2022-12-01T00:00:00Z\",\"endTime\":\"2022-12-10T00:00:00Z\",\"maxResults\":
\"2\"}"
            },
            "sourceIpAddress": "127.0.0.1",
            "userAgent": "aws-cli/2.9.0 Python/3.9.11 Darwin/21.3.0 exe/x86_64
 prompt/off command/codecatalyst.list-event-logs"
        },
        {
            "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
            "eventName": "createProject",
            "eventType": "AwsApiCall",
            "eventCategory": "MANAGEMENT",
            "eventSource": "manage",
            "eventTime": "2022-12-01T09:15:32.068000+00:00",
            "operationType": "MUTATION",
            "userIdentity": {
                "userType": "USER",
                "principalId": "a1b2c3d4e5-678fgh90-1a2b-3c4d-e5f6-EXAMPLE11111",
                "userName": "MaryMajor"
            },
            "requestId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
            "requestPayload": {
                "contentType": "application/json",
                "data": "{\"spaceName\":\"ExampleCorp\",\"name\":\"MyFirstProject
\",\"displayName\":\"MyFirstProject\"}"
            },
            "responsePayload": {
                "contentType": "application/json",
```

```
                "data": "{\"spaceName\":\"ExampleCorp\",\"name\":\"MyFirstProject
\",\"displayName\":\"MyFirstProject\",\"id\":\"a1b2c3d4-5678-90ab-cdef-
EXAMPLE4444\"}"
            },
            "sourceIpAddress": "192.0.2.23",
            "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:102.0)
 Gecko/20100101 Firefox/102.0"
        }
    ]
}
```


3. Jalankan list-event-logs perintah lagi dengan --next-token opsi dan nilai token yang dikembalikan untuk mengambil set peristiwa yang dicatat berikutnya yang cocok dengan permintaan.

Kuota untuk identitas, izin, dan akses di CodeCatalyst

Tabel berikut menjelaskan kuota dan batasan untuk identitas, izin, dan akses di Amazon CodeCatalyst. Untuk informasi selengkapnya tentang kuota di Amazon CodeCatalyst, lihatKuota untuk CodeCatalyst.

Sumber Daya	Informasi
Alias di CodeCatalyst	Setiap kombinasi karakter yang diizinkan antara 3 dan 100 karakter panjangnya dan harus dimulai dengan huruf. Karakter yang valid: A-Z, a-z, dan 0-9. Alias tidak bisa: • mengandung kurang dari 3 karakter • berisi spasi atau salah satu karakter berikut: `? ^ * [ \ ~ :`
Jumlah maksimum undangan yang dikirim oleh pengguna per hari	500
Jumlah maksimum undangan yang dikirim ke alamat email per hari	25
Jumlah maksimum Token Akses Pribadi (PAT) per pengguna	100

Sumber Daya	Informasi
Jumlah maksimum koneksi pribadi untuk setiap identitas pengguna (CodeCatalystalias) di semua ruang, per jenis penyedia	1
Kata sandi di CodeCatalyst	Kombinasi karakter yang diizinkan antara 8 dan 64 karakter panjangnya. Karakter yang valid: A-Z, a-z, dan 0-9. Kata sandi Anda dapat menyertakan karakter nonalfanumerik berikut: `( ~ ! @ # $ % ^ & * _ - + = ` | \ { } [ ] : ; " ' < > , . ? /)`
Nama PAT di CodeCatalyst	Kombinasi karakter yang diizinkan antara 1 dan 100 karakter
Waktu sampai undangan anggota proyek berakhir	Kedaluwarsa setelah 24 jam
Waktu hingga undangan anggota luar angkasa berakhir	Kedaluwarsa setelah 24 jam
Waktu hingga verifikasi alamat email kedaluwarsa	Kedaluwarsa 10 menit setelah mengirim

Pemecahan Masalah

Bagian ini dapat membantu Anda memecahkan masalah umum yang mungkin Anda temui saat mengakses profil Amazon Anda. CodeCatalyst

Masalah mendaftar

Anda mungkin mengalami beberapa masalah saat mendaftar. Kami punya beberapa solusi.

Alamat email saya sudah digunakan

Jika email yang Anda masukkan sudah digunakan dan Anda mengenalinya sebagai milik Anda, maka Anda mungkin sudah memiliki profil dengan kami. Masuk dengan identitas yang ada ini. Jika Anda tidak memiliki email yang ada, maka daftarlah dengan email yang berbeda dan tidak terpakai.

Saya tidak dapat menyelesaikan verifikasi email

Jika Anda belum menerima email verifikasi

1. Periksa folder spam, sampah, dan item yang dihapus.

   > **Note**
   >
   > Email verifikasi ini berasal dari alamat `no-reply@signin.aws` atau `no-reply@login.awsapps.com`. Kami menyarankan Anda mengonfigurasi sistem surat Anda sehingga menerima email dari alamat email pengirim ini dan tidak menanganinya sebagai sampah atau spam.

2. Tunggu 5 menit dan segarkan kotak masuk Anda. Periksa lagi folder spam, sampah, dan item yang dihapus.
3. Jika Anda masih tidak melihat email verifikasi, pilih Kirim ulang kode. Jika Anda sudah keluar dari halaman itu, maka restart alur kerja Anda untuk mendaftar dengan Amazon. CodeCatalyst

Kata sandi saya tidak memenuhi persyaratan minimum

Untuk keamanan Anda, kata sandi Anda harus menyertakan 8-20 karakter, baik huruf besar dan kecil, dan angka.

Masalah saat masuk

Saya lupa kata sandi saya

Ikuti langkah-langkah di Saya lupa kata sandi.

Kata sandi saya tidak berfungsi

Anda harus mengikuti persyaratan ini setiap kali Anda menetapkan atau mengubah kata sandi Anda:

- Password bersifat case-sensitive.
- Kata sandi harus memiliki panjang antara 8 dan 64 karakter dengan huruf besar dan kecil, angka, dan setidaknya satu karakter non-alfanumerik.
- Anda tidak dapat menggunakan kembali tiga kata sandi terakhir .

Saya tidak bisa mengaktifkan MFA

Untuk mengaktifkan MFA, tambahkan satu atau beberapa perangkat MFA ke profil Anda dengan mengikuti langkah-langkah di. Konfigurasikan ID AWS Builder Anda untuk masuk dengan otentikasi multi-faktor (MFA)

Saya tidak dapat menambahkan perangkat MFA

Jika Anda menemukan bahwa Anda tidak dapat menambahkan perangkat MFA lain, itu mungkin telah mencapai batas perangkat MFA yang dapat Anda daftarkan. Anda mungkin perlu menghapus perangkat MFA yang ada sebelum menambahkan yang baru.

Saya tidak dapat menghapus perangkat MFA

Jika Anda bermaksud menonaktifkan MFA, lanjutkan dengan menghapus perangkat MFA Anda dengan mengikuti langkah-langkah di. Menghapus perangkat MFA Namun, jika Anda ingin mengaktifkan MFA, Anda harus menambahkan perangkat MFA lain sebelum mencoba menghapus perangkat MFA yang ada. Untuk informasi selengkapnya tentang menambahkan perangkat MFA lain, lihat. Cara mendaftarkan perangkat untuk digunakan dengan otentikasi multi-faktor

Masalah keluar

Saya tidak dapat menemukan tempat untuk keluar

Di sudut kanan atas halaman, pilih Keluar.

Keluar tidak membuat saya keluar sepenuhnya

Sistem ini dirancang untuk segera keluar, tetapi keluar penuh mungkin memakan waktu hingga satu jam.

Saya mendapatkan kesalahan peran tidak ada untuk alur kerja yang gagal

Masalah: Setelah membuat proyek dari aplikasi web atau cetak biru tanpa server, alur kerja gagal dengan kesalahan berikut:

CLIENT_ERROR: Peran tidak ada

Solusi yang mungkin: Setelah Anda mengonfigurasi peran IAM dengan izin untuk menjalankan alur kerja Anda, dan Anda telah menambahkan peran IAM ke YAMAL alur kerja Anda, alur kerja masih

gagal karena peran IAM mungkin perlu ditambahkan ke koneksi akun Anda. Tambahkan peran IAM ke koneksi akun untuk ruang Anda seperti yang dijelaskan dalamMenambahkan peran IAM ke koneksi akun.

Saya mendapatkan kesalahan peran untuk alur kerja yang gagal

Masalah: Setelah membuat proyek dari aplikasi web atau cetak biru tanpa server, alur kerja gagal dengan kesalahan berikut:

CLIENT_ERROR: Peran tidak diatur dengan benar atau tidak ada

Solusi yang mungkin: Ruang tempat proyek dibuat mungkin perlu menyiapkan Akun AWS koneksi atau mungkin perlu menyelesaikan permintaan koneksi akun. Jika ruang Anda sudah memiliki Akun AWS koneksi aktif, buat dan tambahkan peran IAM dengan izin untuk menjalankan tindakan alur kerja. Tambahkan peran IAM ke koneksi akun Anda seperti yang dijelaskan dalamMenambahkan peran IAM ke koneksi akun.

Solusi yang mungkin: Jika proyek dibuat tanpa menentukan koneksi, maka koneksi akun perlu dikaitkan dengan lingkungan penyebaran. Jika ruang Anda sudah memiliki Akun AWS koneksi aktif dan peran IAM ditambahkan, Anda harus menambahkan koneksi akun dengan peran IAM ke lingkungan penyebaran Anda seperti yang dijelaskan dalam. Menambahkan koneksi akun dan peran IAM ke lingkungan penerapan Anda

Saya perlu memperbarui peran IAM dalam alur kerja proyek

Jika Akun AWS koneksi sudah diatur sepenuhnya, dan peran IAM dibuat dan ditambahkan ke koneksi akun, Anda dapat memperbarui peran IAM dalam alur kerja proyek Anda.

1. Pilih opsi CI/CD dan pilih alur kerja Anda. Pilih tombol YAMAL.
2. Pilih Edit.
3. Di `ActionRoleArn:` lapangan, ganti ARN peran IAM dengan ARN peran IAM yang diperbarui. Pilih Validasi.
4. Pilih Terapkan.

   Alur kerja dimulai secara otomatis jika di cabang jalur utama. Jika tidak, untuk menjalankan kembali alur kerja, pilih Jalankan.

Saya memiliki permintaan peninjauan untuk GitHub akun saya setelah membuat koneksi pribadi

Setelah Anda membuat koneksi pribadi ke GitHub, CodeCatalyst aplikasi diinstal untuk GitHub akun Anda sebagai GitHub aplikasi. Jika ada sumber daya tertentu CodeCatalyst yang memerlukan izin baca atau tulis yang diperbarui, Anda mungkin perlu mengakses GitHub akun untuk memperbarui izin pada aplikasi yang diinstal.

1. Masuk ke GitHub dan navigasikan ke pengaturan akun Anda untuk aplikasi yang diinstal. Pilih ikon profil Anda, pilih Pengaturan, lalu pilih Aplikasi.
2. Pada tab GitHub Aplikasi Terinstal, dalam daftar aplikasi yang diinstal, lihat aplikasi yang diinstal CodeCatalyst. Tautan permintaan Tinjauan akan ditampilkan jika ada izin untuk meninjau.
3. Pilih tautannya, lalu konfirmasikan kredensialnya saat diminta. Masukkan kredensil Anda dan pilih Verifikasi.
4. Terima izin baru, tunjukkan repositori tempat Anda ingin menerapkan izin, lalu pilih Simpan.

Bagaimana cara mengisi formulir dukungan?

Anda dapat pergi ke Amazon CodeCatalyst atau mengisi formulir Support Feedback. Di bagian Permintaan informasi, di bawah Bagaimana kami dapat membantu Anda, sertakan bahwa Anda adalah CodeCatalyst pelanggan Amazon. Berikan detail sebanyak mungkin sehingga kami dapat mengatasi masalah Anda dengan paling efisien.

Tambahkan fungsionalitas ke proyek dengan ekstensi di CodeCatalyst

Amazon CodeCatalyst menyertakan ekstensi yang membantu Anda menambahkan fungsionalitas dan berintegrasi dengan produk di luar CodeCatalyst. Dengan ekstensi dari CodeCatalyst katalog, tim dapat menyesuaikan pengalaman mereka CodeCatalyst.

Topik

Ekstensi pihak ketiga yang tersedia

Anda dapat menambahkan fungsionalitas tertentu ke CodeCatalyst proyek Anda tergantung pada ekstensi yang Anda pilih untuk mengintegrasikan sumber daya.

Mengintegrasikan GitHub repositori di CodeCatalyst

GitHub adalah layanan berbasis cloud yang membantu pengembang menyimpan dan mengelola kode mereka. Ekstensi GitHub repositori memungkinkan Anda menggunakan GitHub repositori tertaut dalam proyek Amazon. CodeCatalyst Anda juga dapat menautkan GitHub repositori saat membuat proyek baru CodeCatalyst . Untuk informasi selengkapnya, lihat Membuat proyek dengan repositori pihak ketiga yang ditautkan.

> **Note**
>
> - Anda tidak dapat menggunakan GitHub repositori kosong atau diarsipkan dengan proyek. CodeCatalyst
> - Ekstensi GitHub repositori tidak kompatibel dengan repositori GitHub Enterprise Server.

Setelah Anda menginstal dan mengkonfigurasi ekstensi GitHub repositori, Anda akan dapat:

- Lihat GitHub repositori Anda dalam daftar repositori sumber di CodeCatalyst
- Simpan dan kelola file definisi alur kerja di repositori Anda GitHub
- Membuat, membaca, memperbarui, dan menghapus file yang disimpan dalam GitHub repositori tertaut dari CodeCatalyst Dev Environments
- Simpan dan indeks file dari GitHub repositori tertaut di CodeCatalyst
- Buat CodeCatalyst proyek dengan repositori akun yang terhubung GitHub
- Buat GitHub repositori dengan kode yang dihasilkan oleh cetak biru saat membuat proyek dengan cetak biru atau menerapkan cetak biru
- CodeCatalyst Alur kerja mulai berjalan secara otomatis saat kode didorong ke GitHub repositori tertaut, atau saat permintaan tarik dibuat, dimodifikasi, atau ditutup di repositori tertaut GitHub
- Gunakan file sumber GitHub repositori tertaut dalam alur kerja CodeCatalyst
- Baca dan jalankan GitHub tindakan dalam CodeCatalyst alur kerja
- Kirim status CodeCatalyst proses alur kerja ke GitHub repositori tertaut, dan blokir penggabungan permintaan GitHub tarik berdasarkan status komit

Mengintegrasikan repositori Bitbucket di CodeCatalyst

Bitbucket adalah layanan berbasis cloud yang membantu pengembang menyimpan dan mengelola kode mereka. Ekstensi repositori Bitbucket memungkinkan Anda menggunakan repositori Bitbucket tertaut dalam proyek Amazon. CodeCatalyst Anda juga dapat menautkan repositori Bitbucket saat membuat proyek baru. CodeCatalyst Untuk informasi selengkapnya, lihat Membuat proyek dengan repositori pihak ketiga yang ditautkan.

> **Note**
>
> - Anda tidak dapat menggunakan repositori Bitbucket kosong atau diarsipkan dengan proyek. CodeCatalyst
> - Ekstensi repositori Bitbucket tidak kompatibel dengan repositori Bitbucket Data Center.

Setelah Anda menginstal dan mengkonfigurasi ekstensi repositori Bitbucket, Anda akan dapat:

- Lihat repositori Bitbucket Anda dalam daftar repositori sumber di CodeCatalyst
- Simpan dan kelola file definisi alur kerja di repositori Bitbucket Anda.
- Membuat, membaca, memperbarui, dan menghapus file yang disimpan dalam repositori Bitbucket tertaut dari Dev Environments CodeCatalyst
- Buat CodeCatalyst proyek dengan repositori akun Bitbucket yang terhubung
- Simpan dan indeks file dari repositori Bitbucket yang ditautkan di CodeCatalyst
- Buat repositori Bitbucket dengan kode yang dihasilkan oleh cetak biru saat membuat proyek dengan cetak biru atau menerapkan cetak biru
- CodeCatalyst Alur kerja mulai berjalan secara otomatis saat kode didorong ke repositori Bitbucket yang ditautkan, atau saat permintaan tarik dibuat, dimodifikasi, atau ditutup di repositori Bitbucket yang ditautkan
- Gunakan file sumber repositori Bitbucket yang ditautkan dalam alur kerja CodeCatalyst
- Kirim status CodeCatalyst proses alur kerja ke repositori Bitbucket yang ditautkan, dan blokir penggabungan permintaan tarik Bitbucket berdasarkan status komit

Mengintegrasikan masalah Jira di CodeCatalyst

Jira adalah aplikasi perangkat lunak yang membantu tim pengembangan tangkas merencanakan, menetapkan, melacak, melaporkan, dan mengelola pekerjaan. Ekstensi Perangkat Lunak Jira memungkinkan Anda menggunakan proyek Jira dalam proyek Amazon CodeCatalyst .

> **Note**
>
> CodeCatalyst hanya kompatibel dengan Jira Software Cloud.

Setelah Anda menginstal dan mengonfigurasi ekstensi Perangkat Lunak Jira untuk CodeCatalyst proyek Amazon, Anda akan dapat:

- Akses proyek Jira dari CodeCatalyst dengan menautkannya ke proyek CodeCatalyst
- Perbarui masalah Jira dengan permintaan CodeCatalyst tarik
- Lihat status dan alur kerja yang berjalan dari permintaan CodeCatalyst tarik tertaut dalam masalah Jira

Konsep ekstensi

Berikut adalah beberapa konsep dan istilah yang perlu diketahui saat bekerja dengan ekstensi di CodeCatalyst.

Ekstensi

Ekstensi adalah add-on yang dapat Anda instal ke CodeCatalyst ruang Anda untuk menambahkan fungsionalitas baru ke proyek Anda dan mengintegrasikan dengan layanan di luar. CodeCatalyst Ekstensi dapat dijelajahi dan diinstal dari CodeCatalyst katalog.

CodeCatalyst katalog

CodeCatalyst Katalog adalah daftar terpusat dari semua ekstensi yang tersedia di CodeCatalyst. Anda dapat menelusuri CodeCatalyst katalog untuk menemukan ekstensi yang dapat meningkatkan pengalaman tim Anda di berbagai bidang CodeCatalyst seperti sumber, alur kerja, dan lainnya.

Menghubungkan dan menghubungkan

Bergantung pada sumber daya pihak ketiga yang ingin Anda gunakan atau kelola, Anda perlu menghubungkan GitHub akun, ruang kerja Bitbucket, atau proyek Jira. Kemudian, Anda perlu menautkan repositori, GitHub repositori Bitbucket, atau proyek Jira ke proyek Anda. CodeCatalyst

- GitHub repositori: Connect GitHub account dan kemudian link GitHub repositori.
- Repositori Bitbucket: Hubungkan ruang kerja Bitbucket dan kemudian tautkan repositori Bitbucket.
- Perangkat Lunak Jira: Hubungkan situs Jira dan kemudian tautkan proyek Jira.

Mulai cepat: Menginstal ekstensi, menghubungkan penyedia, dan menautkan sumber daya di CodeCatalyst

Tutorial ini memberikan panduan dari tiga tugas berikut:

1. Instal GitHub repositori, repositori Bitbucket atau ekstensi Perangkat Lunak Jira. Anda diminta di situs eksternal untuk terhubung dan menyediakan CodeCatalyst akses ke sumber daya pihak ketiga Anda, yang dilakukan sebagai bagian dari langkah berikutnya.

   > **Important**
   >
   > Untuk menginstal GitHub repositori, repositori Bitbucket, atau ekstensi Perangkat Lunak Jira ke CodeCatalyst ruang Anda, Anda harus masuk dengan akun yang memiliki peran administrator Space di ruang tersebut.

2. Hubungkan GitHub akun Anda, ruang kerja Bitbucket, atau situs Jira ke. CodeCatalyst

   > **Important**
   >
   > Untuk menghubungkan GitHub akun, ruang kerja Bitbucket, atau situs Jira ke CodeCatalyst ruang Anda, Anda harus menjadi administrator sumber pihak ketiga dan administrator Space. CodeCatalyst

> **Important**
>
> Setelah Anda menginstal ekstensi repositori, repositori apa pun yang Anda tautkan CodeCatalyst akan memiliki kode mereka diindeks dan disimpan. CodeCatalyst Ini akan membuat kode dapat dicari di. CodeCatalyst Untuk lebih memahami perlindungan data untuk kode Anda saat menggunakan repositori tertaut di CodeCatalyst, lihat Perlindungan data di CodeCatalyst Panduan Pengguna Amazon.

> **Note**
>
> Jika Anda menggunakan koneksi ke GitHub akun, Anda harus membuat koneksi pribadi untuk membuat pemetaan identitas antara CodeCatalyst identitas dan GitHub identitas Anda. Untuk informasi selengkapnya, lihat Koneksi pribadi dan Mengakses GitHub sumber daya dengan koneksi pribadi.

3. Tautkan GitHub repositori, repositori Bitbucket, atau proyek Jira Anda ke proyek Anda. CodeCatalyst

> **Important**
>
> - Meskipun Anda dapat menautkan repositori GitHub atau Bitbucket sebagai Kontributor, Anda hanya dapat memutuskan tautan repositori pihak ketiga sebagai administrator Space atau administrator Project. Untuk informasi selengkapnya, lihat Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.
> - Untuk menautkan proyek Jira Anda ke CodeCatalyst proyek Anda, Anda harus menjadi administrator CodeCatalyst Space atau administrator CodeCatalyst Proyek.

> **Note**
>
> - Repositori GitHub atau Bitbucket hanya dapat ditautkan ke satu CodeCatalyst proyek dalam satu spasi.
> - Anda tidak dapat menggunakan repositori kosong atau diarsipkan GitHub atau Bitbucket dengan proyek. CodeCatalyst

- Anda tidak dapat menautkan repositori GitHub atau Bitbucket yang memiliki nama yang sama dengan repositori dalam proyek. CodeCatalyst
- Ekstensi GitHub repositori tidak kompatibel dengan repositori GitHub Enterprise Server.
- Ekstensi repositori Bitbucket tidak kompatibel dengan repositori Bitbucket Data Center.
- Sebuah CodeCatalyst proyek hanya dapat ditautkan ke satu proyek Jira. Proyek Jira dapat ditautkan ke beberapa CodeCatalyst proyek.

Anda juga dapat menginstal repositori atau ekstensi GitHub repositori Bitbucket, terhubung ke GitHub akun Anda atau ruang kerja Bitbucket, dan menautkan repositori pihak ketiga saat membuat proyek baru. CodeCatalyst Untuk informasi selengkapnya, lihat Membuat proyek dengan repositori pihak ketiga yang ditautkan.

Topik

Langkah 1: Instal ekstensi pihak ketiga dari CodeCatalyst katalog

Langkah pertama untuk menggunakan sumber daya thid-party CodeCatalyst adalah menginstal ekstensi GitHub repositori dari katalog. CodeCatalyst Untuk menginstal ekstensi, lakukan langkah-langkah berikut, pilih ekstensi untuk sumber daya pihak ketiga yang ingin Anda gunakan. GitHub repositori dan repositori Bitbucket memungkinkan Anda untuk menggunakan GitHub atau repositor Bitbucket di. CodeCatalyst Jira Software memungkinkan Anda untuk mengelola masalah Jira di. CodeCatalyst

Untuk menginstal ekstensi dari CodeCatalyst katalog

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.
3. Arahkan ke CodeCatalyst CodeCatalyst katalog dengan memilih ikon Katalog

di menu atas. Anda dapat mencari GitHub repositori, repositori Bitbucket, atau Perangkat Lunak Jira. Anda juga dapat memfilter ekstensi berdasarkan kategori.

4. (Opsional) Untuk melihat detail lebih lanjut tentang ekstensi, seperti izin yang akan dimiliki ekstensi, pilih nama ekstensi.
5. Pilih Instal. Tinjau izin yang diperlukan oleh ekstensi, dan jika Anda ingin melanjutkan, pilih Instal lagi.

Setelah menginstal ekstensi, Anda dibawa ke halaman detail ekstensi. Bergantung pada ekstensi yang Anda instal, Anda dapat melihat dan mengelola penyedia yang terhubung dan sumber daya yang ditautkan.

Langkah 2: Hubungkan penyedia pihak ketiga Anda ke CodeCatalyst ruang Anda

Setelah Anda menginstal repositori, GitHub repositori Bitbucket, atau ekstensi Perangkat Lunak Jira, langkah selanjutnya adalah menghubungkan GitHub akun Anda, ruang kerja Bitbucket, atau situs Jira ke ruang Anda. CodeCatalyst

Untuk menghubungkan GitHub akun Anda, ruang kerja Bitbucket, atau situs Jira ke CodeCatalyst

- Lakukan salah satu hal berikut tergantung pada ekstensi pihak ketiga yang Anda instal:
  - GitHub repositori: Connect ke akun. GitHub
    1. Di tab Connected GitHub Accounts, pilih Connect GitHub account untuk pergi ke situs eksternal untuk GitHub.
    2. Masuk ke GitHub akun Anda menggunakan GitHub kredensi Anda, lalu pilih akun tempat Anda ingin menginstal Amazon. CodeCatalyst

> **Tip**
>
> Jika sebelumnya Anda telah menghubungkan GitHub akun ke ruang tersebut, Anda tidak akan diminta untuk mengotorisasi ulang. Anda malah akan melihat kotak dialog yang menanyakan di mana Anda ingin menginstal ekstensi jika Anda adalah anggota atau kolaborator di lebih dari satu GitHub ruang, atau halaman konfigurasi untuk CodeCatalyst aplikasi Amazon jika Anda hanya memiliki satu GitHub ruang. Konfigurasikan aplikasi untuk akses repositori yang ingin Anda

izinkan, lalu pilih Simpan. Jika tombol Simpan tidak aktif, buat perubahan pada konfigurasi, lalu coba lagi.

3. Pilih apakah Anda ingin mengizinkan CodeCatalyst untuk mengakses semua repositori saat ini dan masa depan, atau pilih GitHub repositori tertentu yang ingin Anda gunakan. CodeCatalyst Opsi default adalah memasukkan semua GitHub repositori di GitHub akun, termasuk repositori future yang akan diakses oleh. CodeCatalyst
4. Tinjau izin yang diberikan CodeCatalyst, lalu pilih Instal.

Setelah menghubungkan GitHub akun Anda CodeCatalyst, Anda akan dibawa ke halaman detail ekstensi GitHub repositori, di mana Anda dapat melihat dan mengelola GitHub akun yang terhubung dan repositori tertaut GitHub .

- Repositori Bitbucket: Connect ke ruang kerja Bitbucket.

1. Di tab Workspaces Bitbucket Terhubung, pilih Connect Bitbucket workspace untuk pergi ke situs eksternal untuk Bitbucket.
2. Masuk ke ruang kerja Bitbucket Anda menggunakan kredensi Bitbucket Anda.
3. Dari menu tarik-turun Otorisasi untuk ruang kerja, pilih ruang kerja Bitbucket yang ingin Anda akses, lalu pilih Grant CodeCatalyst access.

   a. Jika ini pertama kalinya Anda memberikan CodeCatalyst akses ke ruang kerja, pilih Buka pengaturan untuk mengaktifkan mode pengembangan Bitbucket.

   b. Pada halaman Aplikasi terinstal di ruang kerja Anda, pilih kotak centang Aktifkan mode pengembangan untuk mengizinkan penginstalan CodeCatalyst, lalu sambungkan lagi dari. CodeCatalyst

      Setelah Anda mengizinkan instalasi CodeCatalyst, ia memiliki akses ke semua repositori Anda di ruang kerja Bitbucket.

> **Tip**
>
> Jika sebelumnya Anda telah menghubungkan ruang kerja Bitbucket ke ruang tersebut, Anda tidak akan diminta untuk mengotorisasi ulang. Sebagai gantinya, Anda akan melihat dialog yang menanyakan di mana Anda ingin menginstal ekstensi jika Anda anggota atau kolaborator di lebih dari satu ruang kerja Bitbucket, atau halaman konfigurasi untuk CodeCatalyst aplikasi Amazon jika

Anda hanya memiliki satu ruang kerja Bitbucket. Konfigurasikan aplikasi untuk akses ruang kerja yang ingin Anda izinkan, lalu pilih Grant access. Jika tombol akses Grant tidak aktif, buat perubahan pada konfigurasi, lalu coba lagi.

Setelah menghubungkan ruang kerja Bitbucket CodeCatalyst, Anda akan dibawa ke halaman detail ekstensi repositori Bitbucket, tempat Anda dapat melihat dan mengelola ruang kerja Bitbucket yang terhubung dan repositori Bitbucket yang ditautkan.

- Perangkat Lunak Jira: Hubungkan situs Jira.

  1. Di tab Connected Jira sites, pilih Connect Jira site untuk pergi ke situs eksternal untuk Atlassian Marketplace.
  2. Pilih Dapatkan sekarang untuk memulai menginstal CodeCatalyst di situs Jira Anda.

     Note

     Jika sebelumnya Anda menginstal CodeCatalyst ke situs Jira Anda, Anda akan diberi tahu. Pilih Mulai untuk dibawa ke langkah terakhir.

  3. Tergantung pada peran Anda, lakukan salah satu hal berikut:

     1. Jika Anda seorang administrator situs Jira, dari menu dropdown situs, pilih situs Jira untuk menginstal CodeCatalyst aplikasi, dan kemudian pilih Instal aplikasi.

        Note

        Jika Anda memiliki satu situs Jira, langkah ini tidak akan muncul, dan Anda akan secara otomatis diarahkan ke langkah berikutnya.

     2. a. Jika Anda bukan administrator Jira, dari menu dropdown situs, pilih situs Jira untuk menginstal CodeCatalyst aplikasi, lalu pilih Minta aplikasi. Untuk informasi selengkapnya tentang cara menginstal aplikasi Jira, lihat Siapa yang dapat menginstal aplikasi? .

        b. Masukkan alasan Anda perlu menginstal CodeCatalyst ke bidang teks input atau menyimpan teks default, lalu pilih Kirim permintaan.

  4. Tinjau tindakan yang dilakukan CodeCatalyst ketika aplikasi diinstal, dan kemudian pilih Dapatkan sekarang.

5. Setelah aplikasi diinstal, pilih Kembali CodeCatalyst untuk kembali ke CodeCatalyst.

Setelah menghubungkan situs Jira Anda CodeCatalyst, Anda dapat melihat situs yang terhubung di tab Situs Jira Terhubung di halaman detail ekstensi Perangkat Lunak Jira.

Langkah 3: Tautkan sumber daya pihak ketiga Anda ke CodeCatalyst proyek Anda

Langkah ketiga dan terakhir untuk menggunakan repositori GitHub atau Bitbucket Anda atau mengelola masalah Jira CodeCatalyst adalah menautkannya ke CodeCatalyst proyek tempat Anda ingin menggunakannya.

Untuk menautkan GitHub repositori, repositori Bitbucket, atau proyek Jira ke proyek dari halaman detail ekstensi CodeCatalyst

- Lakukan salah satu hal berikut tergantung pada ekstensi pihak ketiga yang Anda instal dan penyedia yang Anda sambungkan:
  - GitHub repositori: Tautkan repositori. GitHub
    1. Di tab GitHub Repositori Tertaut, pilih Repositori tautan GitHub .
    2. Dari dropdown GitHub akun, pilih GitHub akun yang berisi repositori yang ingin Anda tautkan.
    3. Dari dropdown GitHub repositori, pilih repositori yang ingin Anda tautkan ke proyek. CodeCatalyst

       > **Tip**
       >
       > Jika nama repositori berwarna abu-abu, Anda tidak dapat menautkan repositori itu karena telah ditautkan ke proyek lain di ruang tersebut.

    4. (Opsional) Jika Anda tidak melihat GitHub repositori dalam daftar repositori, mungkin tidak dikonfigurasi untuk akses repositori di aplikasi Amazon di. CodeCatalyst GitHub Anda dapat mengonfigurasi GitHub repositori mana yang dapat digunakan CodeCatalyst di akun yang terhubung.

       a. Arahkan ke GitHubakun Anda, pilih Pengaturan, lalu pilih Aplikasi.

b. Di tab GitHub Aplikasi Terinstal, pilih Konfigurasi untuk CodeCatalyst aplikasi Amazon.

c. Lakukan salah satu hal berikut untuk mengonfigurasi akses GitHub repositori yang ingin Anda tautkan: CodeCatalyst

- Untuk menyediakan akses ke semua repositori saat ini dan masa depan, pilih Semua repositori.
- Untuk menyediakan akses ke repositori tertentu, pilih Hanya pilih repositori, pilih menu tarik-turun Pilih repositori, lalu pilih repositori yang ingin Anda izinkan untuk ditautkan. CodeCatalyst

5. Dari menu tarik-turun CodeCatalyst proyek, pilih CodeCatalyst proyek yang ingin Anda tautkan GitHub repositori.
6. Pilih Tautkan.

Jika Anda tidak lagi ingin menggunakan GitHub repositori CodeCatalyst, Anda dapat memutuskan tautannya dari proyek. CodeCatalyst Ketika repositori tidak terhubung, peristiwa dalam repositori itu tidak akan memulai alur kerja berjalan, dan Anda tidak akan dapat menggunakan repositori itu dengan Dev Environments. CodeCatalyst Untuk informasi selengkapnya, lihat Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.

- Repositori Bitbucket: Tautkan repositori Bitbucket.

1. Di tab Repositori Bitbucket Tertaut, pilih repositori Link Bitbucket.
2. Dari dropdown ruang kerja Bitbucket, pilih ruang kerja Bitbucket yang berisi repositori yang ingin Anda tautkan.
3. Dari dropdown repositori Bitbucket, pilih repositori yang ingin Anda tautkan ke proyek. CodeCatalyst

> **Tip**
>
> Jika nama repositori berwarna abu-abu, Anda tidak dapat menautkan repositori itu karena telah ditautkan ke proyek lain di ruang tersebut.

4. Dari menu tarik-turun CodeCatalyst proyek, pilih CodeCatalyst proyek yang ingin Anda tautkan ke repositori Bitbucket.
5. Pilih Tautkan.

Jika Anda tidak lagi ingin menggunakan repositori Bitbucket CodeCatalyst, Anda dapat memutuskan tautannya dari proyek. CodeCatalyst Ketika repositori tidak terhubung, peristiwa dalam repositori itu tidak akan memulai alur kerja berjalan, dan Anda tidak akan dapat menggunakan repositori itu dengan Dev Environments. CodeCatalyst Untuk informasi selengkapnya, lihat Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.

- Perangkat Lunak Jira: Tautkan proyek Jira.

  1. Di tab Proyek Jira Tertaut, pilih Tautkan proyek Jira.
  2. Dari menu dropdown situs Jira, pilih situs Jira yang berisi proyek yang ingin Anda tautkan.
  3. Dari menu tarik-turun proyek Jira, pilih proyek yang ingin Anda tautkan ke proyek. CodeCatalyst
  4. Dari menu tarik-turun CodeCatalyst proyek, pilih CodeCatalyst proyek yang ingin Anda tautkan ke proyek Jira.
  5. Pilih Tautkan.

  Setelah proyek Jira ditautkan ke CodeCatalyst proyek, akses ke CodeCatalyst masalah dinonaktifkan sepenuhnya, dan Masalah di panel CodeCatalyst navigasi akan diganti dengan item masalah Jira yang tertaut ke proyek Jira.

  Jika Anda tidak lagi ingin menggunakan proyek Jira CodeCatalyst, Anda dapat memutuskan tautannya dari proyek Anda CodeCatalyst. Ketika proyek Jira dibatalkan tautannya, masalah Jira tidak akan tersedia di CodeCatalyst proyek, dan CodeCatalyst Masalah akan menjadi penyedia masalah lagi. Untuk informasi selengkapnya, lihat Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.

Anda juga dapat menautkan repositori Bitbucket Anda GitHub ke proyek dari repositori Sumber di Kode. Untuk informasi selengkapnya, lihat Menautkan sumber daya dari penyedia pihak ketiga yang terhubung.

Langkah selanjutnya

Setelah menginstal GitHub repositori atau ekstensi repositori Bitbucket, menghubungkan penyedia sumber daya Anda, dan menautkan repositori pihak ketiga Anda ke CodeCatalyst proyek Anda, Anda dapat menggunakannya dalam alur kerja dan Lingkungan Dev. CodeCatalyst Anda juga dapat

membuat repositori pihak ketiga di GitHub akun yang terhubung atau ruang kerja Bitbucket dengan kode yang dihasilkan dari cetak biru. Untuk informasi selengkapnya, lihat Memulai alur kerja secara otomatis setelah peristiwa repositori pihak ketiga dan Membuat Lingkungan Dev.

Setelah menginstal ekstensi Perangkat Lunak Jira, menghubungkan situs Jira Anda, menautkan proyek Jira Anda ke CodeCatalyst proyek Anda, dan menautkan permintaan tarik, pembaruan dari tercermin dalam proyek Jira CodeCatalyst Anda. Untuk informasi selengkapnya tentang menautkan permintaan tarik ke masalah Jira, lihat. Menautkan masalah Jira untuk menarik permintaan CodeCatalyst Untuk informasi lebih lanjut tentang melihat CodeCatalyst acara di Jira, lihatMelihat CodeCatalyst acara di masalah Jira.

Memasang ekstensi di ruang

Anda dapat menginstal ekstensi untuk CodeCatalyst ruang Anda yang menambahkan fungsionalitas ke proyek di ruang itu. Anda dapat melihat CodeCatalyst katalog dengan memilih ikon Katalog 
Untuk mempelajari lebih lanjut tentang ekstensi dan fungsinya, lihatEkstensi pihak ketiga yang tersedia.

> **Important**
>
> Untuk memasang ekstensi, Anda harus masuk dengan akun yang memiliki peran administrator Space di ruang tersebut.

> **Important**
>
> Setelah Anda menginstal ekstensi repositori, repositori apa pun yang Anda tautkan CodeCatalyst akan memiliki kode mereka diindeks dan disimpan. CodeCatalyst Ini akan membuat kode dapat dicari di. CodeCatalyst Untuk lebih memahami perlindungan data untuk kode Anda saat menggunakan repositori tertaut di CodeCatalyst, lihat Perlindungan data di CodeCatalyst Panduan Pengguna Amazon.

Untuk menginstal ekstensi dari CodeCatalyst katalog

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.

2. Arahkan ke CodeCatalyst ruang Anda.
3. Arahkan ke CodeCatalyst katalog dengan memilih ikon Katalog

   di menu atas. Anda dapat mencari ekstensi atau memfilter ekstensi berdasarkan kategori.
4. (Opsional) Pilih nama ekstensi untuk melihat detail lebih lanjut tentang ekstensi, seperti izin yang akan dimiliki ekstensi.
5. Pilih Instal. Tinjau izin yang diperlukan oleh ekstensi, dan jika Anda ingin melanjutkan, pilih Instal lagi.

Setelah menginstal ekstensi, Anda akan melihat halaman detail untuk ekstensi yang diinstal. Jelajahi tab untuk informasi lebih lanjut tentang ekstensi. Halaman detail juga merupakan tempat Anda akan melakukan konfigurasi ekstensi lebih lanjut jika diperlukan.

Menghapus instalasi ekstensi di ruang

Anda dapat menghapus ekstensi yang sebelumnya dipasang di CodeCatalyst ruang Anda. Menghapus instalasi ekstensi dapat menghapus sumber daya yang terkait dengan ekstensi itu dari CodeCatalyst ruang atau proyek Anda.

> **Important**
>
> Untuk menghapus ekstensi, Anda harus masuk dengan akun yang memiliki peran administrator Space di ruang tersebut.

Untuk menghapus ekstensi dari ruang Anda CodeCatalyst

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.
3. Lakukan salah satu hal berikut untuk melihat daftar ekstensi yang diinstal untuk ruang Anda:

   a. Pilih Pengaturan, lalu pilih Ekstensi yang diinstal.

   b. Pilih ikon Katalog

      di menu atas.

4. Pilih Konfigurasi pada ekstensi yang ingin Anda hapus instalannya.
5. Pilih Copot pemasangan di halaman detail ekstensi.
6. Tinjau informasi di kotak dialog Uninstall extension. Ikuti petunjuknya, lalu pilih Copot pemasangan untuk menghapus instalasi ekstensi.

Menghubungkan GitHub akun, ruang kerja Bitbucket, dan situs Jira CodeCatalyst

Untuk menggunakan repositori GitHub atau Bitbucket atau mengelola proyek Jira di CodeCatalyst, Anda harus terlebih dahulu menghubungkan sumber pihak ketiga Anda ke ruang Anda. CodeCatalyst Untuk mempelajari lebih lanjut tentang ekstensi dan fungsinya, lihatEkstensi pihak ketiga yang tersedia.

> **Important**
>
> Untuk menghubungkan GitHub akun, ruang kerja Bitbucket, atau situs Jira ke CodeCatalyst ruang Anda, Anda harus menjadi administrator sumber pihak ketiga dan administrator Space. CodeCatalyst

> **Note**
>
> Jika Anda menggunakan koneksi ke GitHub akun, Anda harus membuat koneksi pribadi untuk membuat pemetaan identitas antara CodeCatalyst identitas dan GitHub identitas Anda. Untuk informasi selengkapnya, lihat Koneksi pribadi dan Mengakses GitHub sumber daya dengan koneksi pribadi.

Untuk menghubungkan GitHub akun Anda, ruang kerja Bitbucket, atau situs Jira ke CodeCatalyst

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.
3. Lakukan salah satu hal berikut untuk melihat daftar ekstensi yang diinstal untuk ruang Anda:

    a. Pilih Pengaturan, lalu pilih Ekstensi yang diinstal.

b. Pilih ikon Katalog

di menu atas.

4. Pilih Konfigurasi untuk salah satu ekstensi berikut yang ingin Anda konfigurasikan: GitHub repositori, repositori Bitbucket, atau Perangkat Lunak Jira.
5. Lakukan salah satu hal berikut tergantung pada ekstensi pihak ketiga yang Anda pilih untuk dikonfigurasi:

   - GitHub repositori: Connect ke akun. GitHub

     1. Di tab Connected GitHub Accounts, pilih Connect GitHub account untuk pergi ke situs eksternal untuk GitHub.
     2. Masuk ke GitHub akun Anda menggunakan GitHub kredensi Anda, lalu pilih akun tempat Anda ingin menginstal Amazon. CodeCatalyst

        > **Tip**
        >
        > Jika sebelumnya Anda telah menghubungkan GitHub akun ke ruang tersebut, Anda tidak akan diminta untuk mengotorisasi ulang. Anda malah akan melihat kotak dialog yang menanyakan di mana Anda ingin menginstal ekstensi jika Anda adalah anggota atau kolaborator di lebih dari satu GitHub ruang, atau halaman konfigurasi untuk CodeCatalyst aplikasi Amazon jika Anda hanya milik satu GitHub ruang. Konfigurasikan aplikasi untuk akses repositori yang ingin Anda izinkan, lalu pilih Simpan. Jika tombol Simpan tidak aktif, buat perubahan pada konfigurasi, lalu coba lagi.

     3. Pilih apakah Anda ingin mengizinkan CodeCatalyst untuk mengakses semua repositori saat ini dan masa depan, atau pilih GitHub repositori tertentu yang ingin Anda gunakan. CodeCatalyst Opsi default adalah memasukkan semua GitHub repositori di GitHub akun, termasuk repositori future yang akan diakses oleh. CodeCatalyst
     4. Tinjau izin yang diberikan CodeCatalyst, lalu pilih Instal.

   Setelah menghubungkan GitHub akun Anda CodeCatalyst, Anda akan dibawa ke halaman detail ekstensi GitHub repositori, di mana Anda dapat melihat dan mengelola GitHub akun yang terhubung dan repositori tertaut GitHub .

- Repositori Bitbucket: Connect ke ruang kerja Bitbucket.

  1. Di tab Workspaces Bitbucket Terhubung, pilih Connect Bitbucket workspace untuk pergi ke situs eksternal untuk Bitbucket.
  2. Masuk ke ruang kerja Bitbucket Anda menggunakan kredensi Bitbucket Anda.
  3. Dari menu tarik-turun Otorisasi untuk ruang kerja, pilih ruang kerja Bitbucket yang ingin Anda akses, lalu pilih Grant CodeCatalyst access.

     a. Jika ini pertama kalinya Anda memberikan CodeCatalyst akses ke ruang kerja, pilih Buka pengaturan untuk mengaktifkan mode pengembangan Bitbucket.

     b. Pada halaman Aplikasi terinstal di ruang kerja Anda, pilih kotak centang Aktifkan mode pengembangan untuk mengizinkan penginstalan CodeCatalyst, lalu sambungkan lagi dari. CodeCatalyst

        Setelah Anda mengizinkan instalasi CodeCatalyst, ia memiliki akses ke semua repositori Anda di ruang kerja Bitbucket.

  > **Tip**
  >
  > Jika sebelumnya Anda telah menghubungkan ruang kerja Bitbucket ke ruang tersebut, Anda tidak akan diminta untuk mengotorisasi ulang. Sebagai gantinya, Anda akan melihat dialog yang menanyakan di mana Anda ingin menginstal ekstensi jika Anda anggota atau kolaborator di lebih dari satu ruang kerja Bitbucket, atau halaman konfigurasi untuk CodeCatalyst aplikasi Amazon jika Anda hanya memiliki satu ruang kerja Bitbucket. Konfigurasikan aplikasi untuk akses ruang kerja yang ingin Anda izinkan, lalu pilih Grant access. Jika tombol akses Grant tidak aktif, buat perubahan pada konfigurasi, lalu coba lagi.

  Setelah menghubungkan ruang kerja Bitbucket CodeCatalyst, Anda akan dibawa ke halaman detail ekstensi repositori Bitbucket, tempat Anda dapat melihat dan mengelola ruang kerja Bitbucket yang terhubung dan repositori Bitbucket yang ditautkan.

- Perangkat Lunak Jira: Hubungkan situs Jira.

  1. Di tab Connected Jira sites, pilih Connect Jira site untuk pergi ke situs eksternal untuk Atlassian Marketplace.

2. Pilih Dapatkan sekarang untuk memulai menginstal CodeCatalyst di situs Jira Anda.

> **Note**
>
> Jika sebelumnya Anda menginstal CodeCatalyst ke situs Jira Anda, Anda akan diberi tahu. Pilih Mulai untuk dibawa ke langkah terakhir.

3. Tergantung pada peran Anda, lakukan salah satu hal berikut:

   1. Jika Anda seorang administrator situs Jira, dari menu dropdown situs, pilih situs Jira untuk menginstal CodeCatalyst aplikasi, dan kemudian pilih Instal aplikasi.

   > **Note**
   >
   > Jika Anda memiliki satu situs Jira, langkah ini tidak akan muncul, dan Anda akan secara otomatis diarahkan ke langkah berikutnya.

   2. a. Jika Anda bukan administrator Jira, dari menu dropdown situs, pilih situs Jira untuk menginstal CodeCatalyst aplikasi, lalu pilih Minta aplikasi. Untuk informasi selengkapnya tentang cara menginstal aplikasi Jira, lihat Siapa yang dapat menginstal aplikasi? .

      b. Masukkan alasan Anda perlu menginstal CodeCatalyst ke bidang teks input atau menyimpan teks default, lalu pilih Kirim permintaan.

4. Tinjau tindakan yang dilakukan CodeCatalyst ketika aplikasi diinstal, dan kemudian pilih Dapatkan sekarang.
5. Setelah aplikasi diinstal, pilih Kembali CodeCatalyst untuk kembali ke CodeCatalyst.

Setelah menghubungkan situs Jira Anda CodeCatalyst, Anda dapat melihat situs yang terhubung di tab Situs Jira Terhubung di halaman detail ekstensi Perangkat Lunak Jira.

Jika Anda tidak lagi ingin menggunakan repositori, GitHub repositori Bitbucket, atau masalah Jira, Anda dapat memutuskan sumber pihak ketiga CodeCatalyst Anda. Ketika GitHub akun atau ruang kerja Bitbucket terputus, peristiwa di repositori pihak ketiga tidak akan memulai alur kerja berjalan, dan Anda tidak akan dapat menggunakan repositori tersebut dengan Lingkungan Dev. CodeCatalyst Ketika situs Jira terputus, masalah Jira dari proyek situs tidak akan tersedia di CodeCatalyst proyek, dan CodeCatalyst Masalah akan menjadi penyedia masalah lagi. Untuk informasi selengkapnya, lihat Memutuskan koneksi GitHub akun, ruang kerja Bitbucket, dan situs Jira CodeCatalyst.

Memutuskan koneksi GitHub akun, ruang kerja Bitbucket, dan situs Jira CodeCatalyst

Jika Anda tidak lagi ingin menggunakan repositori, GitHub repositori Bitbucket, atau masalah Jira CodeCatalyst, Anda dapat memutuskan sumber pihak ketiga Anda. Setelah GitHub akun atau ruang kerja Bitbucket terputus, peristiwa di repositori tidak akan memulai alur kerja berjalan, dan Anda tidak akan dapat menggunakan repositori tersebut dengan Lingkungan Dev. CodeCatalyst Ketika situs Jira terputus, masalah Jira dari proyek situs tidak akan tersedia di CodeCatalyst proyek, dan CodeCatalyst Masalah akan menjadi penyedia masalah lagi.

> **Note**
>
> - Untuk memutuskan koneksi GitHub akun, Anda harus terlebih dahulu memutuskan tautan semua GitHub repositori tertaut dari akun itu.
> - Untuk memutuskan sambungan ruang kerja Bitbucket, Anda harus terlebih dahulu memutuskan tautan semua repositori Bitbucket yang ditautkan dari ruang kerja tersebut.
> - Untuk memutuskan koneksi situs Jira, Anda harus terlebih dahulu memutuskan tautan semua proyek Jira yang ditautkan dari akun itu.
>
> Untuk informasi selengkapnya, lihat Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.

Untuk memutuskan sambungan GitHub proyek, ruang kerja Bitbucket, atau situs Jira

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.
3. Lakukan salah satu hal berikut untuk melihat daftar ekstensi yang diinstal untuk ruang Anda:
   a. Pilih Pengaturan, lalu pilih Ekstensi yang diinstal.
   b. Pilih ikon Katalog

      di menu atas.
4. Pilih Konfigurasi untuk salah satu ekstensi berikut yang ingin Anda konfigurasikan: GitHub repositori, repositori Bitbucket. atau Perangkat Lunak Jira.

5. Lakukan salah satu hal berikut tergantung pada ekstensi pihak ketiga yang Anda pilih untuk dikonfigurasi:

   - GitHub repositori: Disonnect ke akun. GitHub

     Di tab GitHub Akun yang terhubung, pilih GitHub akun yang ingin Anda putuskan, lalu pilih Putuskan sambungan GitHub akun.
   - Repositori Bitbucket: Disonnect ke ruang kerja Bitbucket.

     Di tab Workspaces Bitbucket yang terhubung, pilih ruang kerja Bitbucket yang ingin Anda putuskan, lalu pilih Disconnect Bitbucket workspace.
   - Perangkat Lunak Jira: Disonnect ke situs Jira.

     Di tab Situs Jira Terhubung, pilih situs Jira yang ingin Anda putuskan, lalu pilih Putuskan sambungan situs Jira.
6. Di kotak dialog Putuskan sambungan, tinjau efek pemutusan akun.
7. Masukkan putuskan sambungan ke bidang input teks, lalu pilih Putuskan sambungan.

Menautkan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst

Sebelum Anda dapat menggunakan repositori GitHub atau Bitbucket atau mengelola proyek JIRA, Anda harus menghubungkan sumber pihak ketiga yang dimiliki repositori atau proyek dengan ruang Anda. CodeCatalyst Untuk informasi selengkapnya, lihat Menghubungkan GitHub akun, ruang kerja Bitbucket, dan situs Jira CodeCatalyst.

Anda dapat menggunakan repositori tertaut GitHub atau Bitbucket dalam alur kerja, di mana peristiwa di repositori tertaut memulai alur kerja yang mungkin membangun, menguji, atau menyebarkan kode, tergantung pada konfigurasi alur kerja. File konfigurasi alur kerja untuk alur kerja yang menggunakan repositori tertaut GitHub atau Bitbucket disimpan di repositori tertaut. Repositori tertaut juga dapat digunakan dengan Dev Environments untuk membuat, memperbarui, dan menghapus file di repositori tertaut. Anda dapat menautkan repositori Bitbucket ke CodeCatalyst proyek baik dari halaman detail repositori GitHub atau ekstensi GitHub repositori Bitbucket, atau dari tampilan repositori Sumber dalam Kode dalam proyek itu sendiri.

Anda dapat menggunakan proyek Jira yang ditautkan untuk mengelola masalah dan menautkan permintaan CodeCatalyst tarik ke masalah Jira. Status ringkasan permintaan tarik dan status peristiwa CodeCatalyst alur kerja terkait tercermin dalam masalah Jira Anda.

Important

Meskipun Anda dapat menautkan repositori GitHub atau Bitbucket sebagai Kontributor, Anda hanya dapat memutuskan tautan repositori pihak ketiga sebagai administrator Space atau administrator Project. Untuk informasi selengkapnya, lihat Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.

Important

Untuk menautkan proyek Jira Anda ke CodeCatalyst proyek Anda, Anda harus menjadi administrator CodeCatalyst Space atau administrator CodeCatalyst Proyek.

Important

Setelah Anda menginstal ekstensi repositori, repositori apa pun yang Anda tautkan CodeCatalyst akan memiliki kode mereka diindeks dan disimpan. CodeCatalyst Ini akan membuat kode dapat dicari di. CodeCatalyst Untuk lebih memahami perlindungan data untuk kode Anda saat menggunakan repositori tertaut di CodeCatalyst, lihat Perlindungan data di CodeCatalyst Panduan Pengguna Amazon.

Note

- Repositori GitHub atau Bitbucket hanya dapat ditautkan ke satu CodeCatalyst proyek dalam satu spasi.
- Anda tidak dapat menggunakan repositori kosong atau diarsipkan GitHub atau Bitbucket dengan proyek. CodeCatalyst
- Anda tidak dapat menautkan repositori GitHub atau Bitbucket yang memiliki nama yang sama dengan repositori dalam proyek. CodeCatalyst
- Ekstensi GitHub repositori tidak kompatibel dengan repositori GitHub Enterprise Server.

- Ekstensi repositori Bitbucket tidak kompatibel dengan repositori Bitbucket Data Center.
- Sebuah CodeCatalyst proyek hanya dapat ditautkan ke satu proyek Jira. Proyek Jira dapat ditautkan ke beberapa CodeCatalyst proyek.

Topik

Menautkan sumber daya dari penyedia pihak ketiga yang terhubung

Untuk menautkan GitHub repositori, repositori Bitbucket, atau proyek Jira ke proyek dari halaman detail ekstensi CodeCatalyst

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.
3. Lakukan salah satu hal berikut untuk melihat daftar ekstensi yang diinstal untuk ruang ruang Anda:

   a. Pilih Pengaturan, lalu pilih Ekstensi yang diinstal.

   b. Pilih ikon Katalog

   di menu atas.
4. Pilih Konfigurasi untuk salah satu ekstensi berikut: GitHub repositori, repositori Bitbucket, atau Perangkat Lunak Jira.
5. Lakukan salah satu hal berikut tergantung pada ekstensi pihak ketiga yang Anda pilih untuk dikonfigurasi:

   - GitHub repositori: Tautkan repositori. GitHub

     1. Di tab GitHub Repositori Tertaut, pilih Repositori tautan GitHub .
     2. Dari dropdown GitHub akun, pilih GitHub akun yang berisi repositori yang ingin Anda tautkan.
     3. Dari dropdown GitHub repositori, pilih repositori yang ingin Anda tautkan ke proyek. CodeCatalyst

> **Tip**
>
> Jika nama repositori berwarna abu-abu, Anda tidak dapat menautkan repositori itu karena telah ditautkan ke proyek lain di ruang tersebut.

4. (Opsional) Jika Anda tidak melihat GitHub repositori dalam daftar repositori, mungkin tidak dikonfigurasi untuk akses repositori di aplikasi Amazon di. CodeCatalyst GitHub Anda dapat mengonfigurasi GitHub repositori mana yang dapat digunakan CodeCatalyst di akun yang terhubung.

   a. Arahkan ke GitHubakun Anda, pilih Pengaturan, lalu pilih Aplikasi.

   b. Di tab GitHub Aplikasi Terinstal, pilih Konfigurasi untuk CodeCatalyst aplikasi Amazon.

   c. Lakukan salah satu hal berikut untuk mengonfigurasi akses GitHub repositori yang ingin Anda tautkan: CodeCatalyst

      - Untuk menyediakan akses ke semua repositori saat ini dan masa depan, pilih Semua repositori.
      - Untuk menyediakan akses ke repositori tertentu, pilih Hanya pilih repositori, pilih menu tarik-turun Pilih repositori, lalu pilih repositori yang ingin Anda izinkan untuk ditautkan. CodeCatalyst

5. Dari menu tarik-turun CodeCatalyst proyek, pilih CodeCatalyst proyek yang ingin Anda tautkan GitHub repositori.
6. Pilih Tautkan.

Jika Anda tidak lagi ingin menggunakan GitHub repositori CodeCatalyst, Anda dapat memutuskan tautannya dari proyek. CodeCatalyst Ketika repositori tidak terhubung, peristiwa dalam repositori itu tidak akan memulai alur kerja berjalan, dan Anda tidak akan dapat menggunakan repositori itu dengan Dev Environments. CodeCatalyst Untuk informasi selengkapnya, lihat Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.

- Repositori Bitbucket: Tautkan repositori Bitbucket.

  1. Di tab Repositori Bitbucket Tertaut, pilih repositori Link Bitbucket.
  2. Dari dropdown ruang kerja Bitbucket, pilih ruang kerja Bitbucket yang berisi repositori yang ingin Anda tautkan.

3. Dari dropdown repositori Bitbucket, pilih repositori yang ingin Anda tautkan ke proyek. CodeCatalyst

> **Tip**
>
> Jika nama repositori berwarna abu-abu, Anda tidak dapat menautkan repositori itu karena telah ditautkan ke proyek lain di ruang tersebut.

4. Dari menu tarik-turun CodeCatalyst proyek, pilih CodeCatalyst proyek yang ingin Anda tautkan ke repositori Bitbucket.
5. Pilih Tautkan.

Jika Anda tidak lagi ingin menggunakan repositori Bitbucket CodeCatalyst, Anda dapat memutuskan tautannya dari proyek. CodeCatalyst Ketika repositori tidak terhubung, peristiwa dalam repositori itu tidak akan memulai alur kerja berjalan, dan Anda tidak akan dapat menggunakan repositori itu dengan Dev Environments. CodeCatalyst Untuk informasi selengkapnya, lihat Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.

- Perangkat Lunak Jira: Tautkan proyek Jira.

1. Di tab Proyek Jira Tertaut, pilih Tautkan proyek Jira.
2. Dari menu dropdown situs Jira, pilih situs Jira yang berisi proyek yang ingin Anda tautkan.
3. Dari menu tarik-turun proyek Jira, pilih proyek yang ingin Anda tautkan ke proyek. CodeCatalyst
4. Dari menu tarik-turun CodeCatalyst proyek, pilih CodeCatalyst proyek yang ingin Anda tautkan ke proyek Jira.
5. Pilih Tautkan.

Setelah proyek Jira ditautkan ke CodeCatalyst proyek, akses ke CodeCatalyst masalah dinonaktifkan sepenuhnya, dan Masalah di panel CodeCatalyst navigasi akan diganti dengan item masalah Jira yang tertaut ke proyek Jira.

Jika Anda tidak lagi ingin menggunakan proyek Jira CodeCatalyst, Anda dapat memutuskan tautannya dari proyek Anda CodeCatalyst. Ketika proyek Jira dibatalkan tautannya, masalah Jira tidak akan tersedia di CodeCatalyst proyek, dan CodeCatalyst Masalah akan menjadi

penyedia masalah lagi. Untuk informasi selengkapnya, lihat Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.

Untuk menautkan repositori GitHub atau Bitbucket ke CodeCatalyst proyek dari halaman repositori sumber dalam proyek

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst proyek Anda.
3. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
4. Pilih Tambahkan repositori, lalu pilih Repositori tautan.
5. Dari menu tarik-turun penyedia Repositori, pilih salah satu penyedia repositori pihak ketiga berikut: atau Bitbucket. GitHub
6. Lakukan salah satu hal berikut tergantung pada penyedia repositori pihak ketiga yang Anda pilih untuk ditautkan:

   - GitHub repositori: Tautkan repositori. GitHub

     1. Dari menu dropdown GitHub akun, pilih GitHub akun yang berisi repositori yang ingin Anda tautkan.
     2. Dari menu tarik-turun GitHub repositori, pilih GitHub akun yang ingin Anda tautkan proyek Anda. CodeCatalyst
     3. (Opsional) Jika Anda tidak melihat GitHub repositori dalam daftar repositori, mungkin tidak dikonfigurasi untuk akses repositori di aplikasi Amazon di. CodeCatalyst GitHub Anda dapat mengonfigurasi GitHub repositori mana yang dapat digunakan CodeCatalyst di akun yang terhubung.

        a. Arahkan ke GitHubakun Anda, pilih Pengaturan, lalu pilih Aplikasi.

        b. Di tab GitHub Aplikasi Terinstal, pilih Konfigurasi untuk CodeCatalyst aplikasi Amazon.

        c. Lakukan salah satu hal berikut untuk mengonfigurasi akses GitHub repositori yang ingin Anda tautkan: CodeCatalyst

           - Untuk menyediakan akses ke semua repositori saat ini dan masa depan, pilih Semua repositori.

- Untuk menyediakan akses ke repositori tertentu, pilih Hanya pilih repositori, pilih menu tarik-turun Pilih repositori, lalu pilih repositori yang ingin Anda izinkan untuk ditautkan. CodeCatalyst

- Repositori Bitbucket: Tautkan repositori Bitbucket.

  1. Dari menu tarik-turun ruang kerja Bitbucket, pilih ruang kerja Bitbucket yang berisi repositori yang ingin Anda tautkan.
  2. Dari menu dropdown repositori Bitbucket, pilih repositori Bitbucket yang ingin Anda tautkan proyek Anda. CodeCatalyst

> **Tip**
>
> Jika nama repositori berwarna abu-abu, Anda tidak dapat menautkan repositori itu karena telah ditautkan ke proyek lain di Amazon. CodeCatalyst

7. Pilih Tautkan.

Jika Anda tidak lagi ingin menggunakan repositori GitHub atau Bitbucket CodeCatalyst, Anda dapat memutuskan tautannya dari proyek. CodeCatalyst Ketika repositori tidak terhubung, peristiwa dalam repositori itu tidak akan memulai alur kerja berjalan, dan Anda tidak akan dapat menggunakan repositori itu dengan Dev Environments. CodeCatalyst Untuk informasi selengkapnya, lihat Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst.

Setelah menautkan repositori Bitbucket Anda ke CodeCatalyst proyek Anda, Anda dapat menggunakannya dalam CodeCatalyst alur kerja dan Lingkungan Dev. GitHub Anda juga dapat menggunakan repositori tertaut dengan Amazon Q Developer, cetak biru, dan banyak lagi. Untuk informasi selengkapnya, lihat Memulai alur kerja secara otomatis setelah peristiwa repositori pihak ketiga dan Membuat Lingkungan Dev.

Setelah menautkan proyek Jira Anda ke CodeCatalyst proyek Anda, dan menautkan permintaan tarik, pembaruan dari CodeCatalyst tercermin dalam proyek Jira Anda. Untuk informasi selengkapnya tentang menautkan permintaan tarik ke masalah Jira, lihat. Menautkan masalah Jira untuk menarik permintaan CodeCatalyst Untuk informasi lebih lanjut tentang melihat CodeCatalyst acara di Jira, lihatMelihat CodeCatalyst acara di masalah Jira.

Menautkan repositori pihak ketiga selama pembuatan proyek CodeCatalyst

Anda dapat menautkan repositori GitHub atau Bitbucket ke CodeCatalyst proyek baru saat membuat proyek baru. CodeCatalyst Untuk informasi selengkapnya, lihat Membuat proyek dengan repositori pihak ketiga yang ditautkan.

Membatalkan tautan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst

Jika Anda tidak lagi ingin menggunakan repositori atau GitHub repositori Bitbucket atau mengelola proyek JIRA di CodeCatalyst, Anda dapat memutuskan tautan repositori atau proyek dari proyek Anda. CodeCatalyst

Membatalkan tautan repositori GitHub atau Bitbucket tidak menghapus repositori atau membuat perubahan apa pun padanya. Itu tidak menghapus file konfigurasi alur kerja apa pun yang disimpan di repositori tertaut itu. Namun, setelah Anda memutuskan tautan repositori GitHub atau Bitbucket, peristiwa di repositori itu tidak akan lagi memulai alur kerja berjalan, dan Anda tidak dapat menggunakan repositori dengan Dev Environments. Anda dapat memutuskan tautan repositori GitHub atau Bitbucket ke CodeCatalyst proyek dari halaman detail repositori atau ekstensi GitHub repositori Bitbucket, atau dari tampilan repositori Sumber dalam Kode dalam proyek itu sendiri.

Memutuskan tautan proyek Jira tidak akan menghapus proyek, termasuk item perencanaan atau informasi pengembangan, atau membuat perubahan apa pun padanya. Namun, setelah Anda memutuskan tautan proyek Jira, masalah Jira proyek tidak akan lagi tersedia untuk ditautkan ke CodeCatalyst proyek, dan CodeCatalyst Masalah akan menjadi penyedia masalah lagi.

> **Important**
>
> Untuk memutuskan tautan repositori atau GitHub repositori Bitbucket Anda dari CodeCatalyst proyek Anda, Anda harus menjadi administrator Space atau administrator Project.

Untuk memutuskan tautan GitHub repositori, repositori Bitbucket, atau proyek Jira dalam proyek dari halaman detail ekstensi CodeCatalyst

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.
3. Lakukan salah satu hal berikut untuk melihat daftar ekstensi yang diinstal untuk ruang Anda:

a. Pilih Pengaturan, lalu pilih Ekstensi yang diinstal.

b. Pilih ikon Katalog

di menu atas.

4. Pilih Konfigurasi untuk salah satu ekstensi berikut yang ingin Anda konfigurasikan: GitHub repositori, repositori Bitbucket. atau Perangkat Lunak Jira.
5. Lakukan salah satu hal berikut tergantung pada ekstensi pihak ketiga yang Anda pilih untuk dikonfigurasi:

   - GitHub repositori: Putuskan tautan repositori. GitHub

     Di tab GitHub repositori, pilih repositori yang ingin Anda GitHub putuskan tautannya, lalu pilih Batalkan tautan repositori. GitHub
   - Repositori Bitbucket: Putuskan tautan repositori Bitbucket.

     Di tab repositori Bitbucket, pilih repositori Bitbucket yang ingin Anda putuskan tautannya, lalu pilih Unlink Bitbucket repository.
   - Perangkat Lunak Jira: Putuskan tautan proyek Jira.

     Di tab proyek Jira, pilih proyek Jira yang ingin Anda putuskan tautannya, lalu pilih Batalkan tautan proyek Jira.
6. Dalam kotak dialog Unlink, tinjau efek pembatalan tautan repositori.
7. Masukkan unlink ke kolom input teks dan pilih Unlink.

Untuk memutuskan tautan repositori GitHub atau Bitbucket dalam CodeCatalyst proyek dari halaman repositori sumber

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst proyek Anda.
3. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.
4. Pilih tombol radio dari repositori yang ingin Anda putuskan tautannya, lalu pilih Batalkan tautan repositori.
5. Tinjau informasi di kotak dialog. Ikuti instruksi, lalu pilih Batalkan tautan untuk memutuskan tautan repositori.

Melihat repositori pihak ketiga dan mencari masalah Jira di CodeCatalyst

Setelah menautkan GitHub atau repositori Bitbucket, Anda dapat melihatnya CodeCatalyst untuk mengonfirmasi dan mengonfigurasi sumber daya. Anda juga dapat mencari masalah Jira yang ditautkan di CodeCatalyst.

Topik

Melihat repositori pihak ketiga di CodeCatalyst

Anda dapat melihat repositori tertaut GitHub atau Bitbucket dalam daftar repositori sumber untuk proyek Anda atau dari repositori atau halaman detail ekstensi GitHub repositori Bitbucket. Memilihnya dari daftar repositori tidak membukanya. CodeCatalyst Sebaliknya, mereka membuka di penyedia repositori pihak ketiga, di mana Anda dapat melihat dan mengerjakan kode di repositori tertaut.

Untuk melihat repositori tertaut GitHub atau Bitbucket di CodeCatalyst

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst proyek Anda.
3. Di panel navigasi, pilih Kode, lalu pilih Repositori sumber.

Untuk melihat repositori tertaut GitHub atau Bitbucket dari halaman detail ekstensi

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda, lalu pilih tab Ekstensi yang diinstal.
3. Bergantung pada repositori pihak ketiga yang ingin Anda lihat, lakukan salah satu hal berikut:
   - Di GitHub repositori, pilih Konfigurasi, lalu pilih Repositori tertaut untuk melihat semua GitHub GitHub repositori yang terhubung ke proyek di ruang Anda. CodeCatalyst CodeCatalyst
   - Di repositori Bitbucket, pilih Configure, lalu pilih Repositori Bitbucket Linked untuk melihat semua repositori Bitbucket yang terhubung ke proyek di ruang Anda. CodeCatalyst CodeCatalyst

Repositori GitHub atau Bitbucket yang ditautkan ke CodeCatalyst proyek Anda ditampilkan dalam daftar. Pilih repositori GitHub atau Bitbucket untuk melihat dan mengedit file di penyedia repositori pihak ketiga.

> **Note**
>
> Jika alur kerja menggunakan repositori GitHub atau Bitbucket dalam tindakan sumber, perubahan yang Anda buat pada alur kerja YAMAL di editor visual atau editor YAMAL CodeCatalyst akan secara otomatis dikomit dan didorong ke repositori pihak ketiga.

Mencari masalah Jira di CodeCatalyst

Setelah menautkan proyek Jira, Anda dapat mencari proyek Jira yang ditautkan untuk masalah menggunakan bilah pencarian CodeCatalyst global. Anda juga dapat mencari masalah Jira CodeCatalyst saat menautkan ke masalah dari permintaan tarik. Untuk informasi selengkapnya tentang menautkan masalah Jira ke permintaan CodeCatalyst tarik, lihat. Menautkan masalah Jira untuk menarik permintaan CodeCatalyst

Untuk mencari masalah Jira di proyek Jira terkait

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst proyek Anda.
3. Di bilah pencarian global, cari proyek Jira yang ditautkan untuk masalah atau masalah Jira yang ingin Anda tautkan ke permintaan tarik.

Memulai alur kerja secara otomatis setelah peristiwa repositori pihak ketiga

Anda dapat menggunakan repositori tertaut GitHub atau Bitbucket sebagai sumber alur kerja, di mana perubahan pada cabang tertentu dalam repositori tertaut GitHub atau Bitbucket secara otomatis memulai alur kerja.

Alur kerja adalah prosedur otomatis yang menjelaskan cara membangun, menguji, dan menyebarkan kode Anda sebagai bagian dari sistem integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD). Alur kerja mendefinisikan serangkaian langkah, atau tindakan, yang harus diambil selama menjalankan alur kerja. Alur kerja juga mendefinisikan peristiwa, atau pemicu, yang menyebabkan

alur kerja dimulai. Untuk menyiapkan alur kerja, Anda membuat file definisi alur kerja menggunakan editor visual atau YAMAL CodeCatalyst konsol.

> **Tip**
>
> Untuk melihat sekilas bagaimana Anda dapat menggunakan alur kerja dalam sebuah proyek, buat proyek dengan cetak biru. Setiap cetak biru menerapkan alur kerja yang berfungsi yang dapat Anda tinjau, jalankan, dan bereksperimen.

Saat Anda mengonfigurasi alur kerja untuk menggunakan repositori tertaut GitHub atau Bitbucket, file konfigurasi alur kerja disimpan di repositori Bitbucket atau itu. GitHub Konfigurasi alur kerja adalah file YAMAL yang mendefinisikan nama alur kerja, pemicu, sumber daya, artefak, dan tindakan. Untuk informasi selengkapnya tentang file konfigurasi alur kerja, lihatAlur kerja definisi YAMAL.

File konfigurasi alur kerja harus berada di `./codecatalyst/workflows/` direktori di repositori Bitbucket GitHub atau Anda.

Anda dapat menggunakan editor alur kerja untuk membuat dan mengonfigurasi alur kerja. Untuk informasi lebih lanjut, lihat Memulai dengan alur kerja dan Menghubungkan alur kerja ke repositori sumber.

Menambahkan pemicu untuk memulai alur kerja berjalan

Anda dapat mengonfigurasi CodeCatalyst alur kerja untuk memulai proses secara otomatis ketika kode didorong ke cabang tertentu dari repositori Bitbucket Anda GitHub . Untuk memulai alur kerja berjalan secara otomatis, tambahkan pemicu ke `Triggers` bagian file konfigurasi alur kerja.

Contoh: Pemicu push kode sederhana

Contoh berikut menunjukkan pemicu yang memulai alur kerja yang dijalankan setiap kali kode didorong ke cabang mana pun di repositori sumber Anda.

```
Triggers:
  - Type: PUSH
```

Contoh: Pemicu permintaan tarik sederhana

Contoh berikut menunjukkan pemicu yang memulai alur kerja yang dijalankan setiap kali permintaan tarik dibuat terhadap cabang mana pun di repositori sumber Anda.

```
Triggers:
  - Type: PULLREQUEST
    Events:
      - OPEN
```

Untuk informasi selengkapnya, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.

Membatasi akses IP dengan GitHub Enterprise Cloud

Anda dapat membatasi akses ke repositori Bitbucket Anda berdasarkan alamat IP dengan mengatur aturan GitHub atau konfigurasi. Anda dapat melakukan ini melalui pengaturan penyedia pihak ketiga atau fitur kontrol akses.

Bergantung pada penyedia repositori pihak ketiga yang Anda gunakan, lihat salah satu dari berikut ini:

- Ekstensi CodeCatalyst GitHub repositori Amazon kompatibel dengan pembatasan akses GitHub Enterprise Cloud IP. Saat mengonfigurasi organisasi GitHub Enterprise Cloud untuk membatasi akses ke alamat IP tertentu, Anda juga dapat mengaktifkan GitHub aplikasi untuk mengonfigurasi daftar izinkan, yang akan memungkinkan CodeCatalyst mendaftarkan alamat IP-nya secara otomatis. GitHub Atau, Anda dapat menambahkan alamat CodeCatalyst IP secara manual.
- Ekstensi repositori Amazon CodeCatalyst Bitbucket kompatibel dengan pembatasan akses Bitbucket Cloud Premium. Saat mengonfigurasi ruang kerja Bitbucket Cloud Premium untuk membatasi akses ke alamat IP tertentu, Anda juga dapat menambahkan alamat IP atau blok jaringan untuk sekumpulan alamat IP ke daftar yang diizinkan.

Jika alamat CodeCatalyst IP tidak ada dalam daftar izin repositori pihak ketiga, CodeCatalyst aplikasi Amazon tidak akan dapat mengakses repositori pihak ketiga Anda. Untuk informasi selengkapnya, lihat Alamat IP yang digunakan oleh ekstensi repositori pihak ketiga.

Alamat IP yang digunakan oleh ekstensi repositori pihak ketiga

Alamat IP berikut digunakan oleh ekstensi pihak ketiga untuk mengakses sumber daya pihak ketiga Anda:

- GitHub repositori:

```
us-west-2
```

```
  52.32.242.246
  54.148.176.49
  35.164.118.94
eu-west-1
  34.241.64.10
  34.246.255.80
  3.248.38.7
```

- Repositori Bitbucket:

```
us-west-2
  35.160.210.199
  54.71.206.108
  54.71.36.205
eu-west-1
  34.242.64.82
  52.18.37.201
  54.77.75.62
```

Memblokir permintaan tarik pihak ketiga bergabung saat alur kerja gagal

Setelah menautkan GitHub repositori ke CodeCatalyst, Anda dapat menambahkan CodeCatalyst alur kerja untuk permintaan tarik. Satu atau beberapa alur kerja berjalan dapat terjadi pada komit tertentu, dan status run dari setiap alur kerja juga CodeCatalyst tercermin sebagai bagian dari status komit di GitHub atau Bitbucket. Saat komit baru didorong, status alur kerja baru akan tercermin GitHub untuk komit baru tersebut. Jika Anda menjalankan alur kerja lagi untuk komit, status alur kerja baru akan menimpa status sebelumnya untuk komit dan alur kerja tersebut.

Anda dapat menyetel aturan perlindungan cabang GitHub untuk memblokir penggabungan permintaan tarik saat komit terbaru memiliki status jalankan alur kerja yang gagal. Dengan aturan perlindungan cabang, status komit terbaru memengaruhi kemampuan untuk menggabungkan permintaan tarik. GitHub Untuk informasi selengkapnya, GitHub lihat dokumentasi Tentang pemeriksaan status dan Tentang cabang yang dilindungi. Untuk mempelajari lebih lanjut tentang alur kerja, lihat Menjalankan alur kerja danMemulai alur kerja berjalan secara otomatis dengan pemicu.

Menautkan masalah Jira untuk menarik permintaan CodeCatalyst

Anda dapat menautkan permintaan tarik yang dibuat di repositori CodeCatalyst sumber ke masalah Jira. Setelah menautkan masalah Jira, masalah ditampilkan sebagai properti permintaan tarik. Akibatnya, peristiwa permintaan tarik, peristiwa alur kerja, dan peristiwa penerapan dikirim ke Jira dan ditambahkan ke masalah Jira. Permintaan tarik dapat ditautkan ke satu atau beberapa masalah Jira. Anda hanya dapat menautkan permintaan tarik yang ada di repositori CodeCatalyst sumber, bukan yang ada di repositori pihak ketiga seperti. GitHub Sebelum Anda dapat menautkan masalah Jira ke permintaan tarik, proyek Jira Anda harus ditautkan ke proyek. CodeCatalyst Untuk informasi selengkapnya tentang menautkan proyek Jira ke CodeCatalyst proyek, lihat. Menautkan GitHub repositori, repositori Bitbucket, dan proyek Jira di CodeCatalyst

> **Note**
>
> Anda tidak dapat membuat permintaan tarik tanpa repositori sumber dengan dua cabang dalam proyek Anda CodeCatalyst . Untuk informasi selengkapnya tentang permintaan tarik, lihat Bekerja dengan permintaan tarik masuk CodeCatalyst.

Untuk menautkan masalah Jira ke permintaan CodeCatalyst tarik

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst proyek Anda.
3. Di panel navigasi, pilih Kode, lalu pilih Tarik permintaan.
4. Pilih Buat permintaan tarik untuk memasukkan detail permintaan tarik.
5. Dari menu tarik-turun repositori Sumber, pilih repositori sumber tempat Anda ingin menautkan permintaan tarik.
6. Dari menu drop-down cabang Sumber, pilih cabang yang berisi perubahan yang ingin Anda tinjau.
7. Dari menu drop-down cabang Tujuan, pilih cabang tempat Anda ingin menggabungkan perubahan yang ditinjau.
8. Di bidang input teks judul permintaan tarik, masukkan judul permintaan tarik Anda.
9. Pilih Masalah tautan untuk masalah Jira - bidang opsional, pilih drop-down, dan cari masalah Jira yang ingin Anda tambahkan dari proyek Jira yang ditautkan.
10. Pilih masalah Jira yang ingin Anda tambahkan ke permintaan tarik.

11. Pilih Buat untuk membuat permintaan tarik.

Setelah Anda menautkan masalah Jira ke permintaan CodeCatalyst tarik, ringkasan permintaan tarik tersedia. Ringkasan mencakup alur kerja, masalah terkait, pengulas wajib, pengulas opsional, dan penulis.

> **Note**
>
> Penerima tugas dan Dibuat oleh informasi yang terkait dengan masalah Jira tidak tersedia di. CodeCatalyst

Setelah menautkan permintaan tarik, proyek yang disinkronkan dan CodeCatalyst proyek Jira memungkinkan pembaruan dari CodeCatalyst tercermin dalam proyek Jira Anda. Status permintaan tarik tertaut dan peristiwa alur kerja apa pun yang terkait dengan permintaan tarik akan muncul di masalah Jira saat melihatnya di Jira. Untuk informasi lebih lanjut tentang melihat CodeCatalyst acara di Jira, lihatMelihat CodeCatalyst acara di masalah Jira.

Melihat CodeCatalyst acara di masalah Jira

Jika CodeCatalyst proyek dan proyek Jira Anda ditautkan, status ringkasan permintaan tarik dan status peristiwa CodeCatalyst alur kerja terkait tercermin dalam masalah Jira Anda. Misalnya, jika Anda menutup atau menggabungkan permintaan tarik CodeCatalyst, pembaruan status tercermin dalam masalah Jira. CodeCatalyst Acara CI/CD alur kerja yang terkait dengan permintaan CodeCatalyst tarik disinkronkan, sehingga alur kerja yang berhasil akan dikirim ke masalah Jira juga.

Untuk melihat CodeCatalyst peristiwa dalam masalah Jira

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst proyek Anda.
3. Di panel CodeCatalyst navigasi, pilih Kode, pilih Permintaan tarik, lalu pilih permintaan tarik dengan masalah Jira yang ingin Anda lihat di proyek Jira Anda.
4. Di panel Info tambahan, pilih masalah Jira yang ingin Anda lihat di proyek Jira Anda.
5. Dari panel Detail di proyek Jira, pilih permintaan tarik yang terdaftar untuk Pengembangan untuk melihat detail permintaan tarik.
6. (Opsional) Untuk melihat build terbaru, pilih tab Builds.

7. (Opsional) Untuk melihat status pengembangan, pilih tab Deployment.

Cari kode, masalah, proyek, dan pengguna di CodeCatalyst

Gunakan bilah pencarian atau jendela hasil pencarian khusus CodeCatalyst untuk mencari kode, masalah, proyek, dan pengguna CodeCatalyst.

Anda dapat menemukan sumber daya di seluruh ruang dan proyek Anda dengan memasukkan kueri seperti nama, deskripsi, dan status ke dalam bilah pencarian. Anda juga dapat menyaring kueri penelusuran menggunakan bahasa kueri penelusuran.

Topik

Untuk mencari

1. Di bilah pencarian di bilah navigasi atas, masukkan permintaan pencarian.
2. (Opsional) Perbaiki kueri penelusuran Anda menggunakan CodeCatalyst bahasa kueri penelusuran. Untuk informasi selengkapnya, lihat Menyempurnakan kueri penelusuran Anda.
3. Lakukan salah satu hal berikut ini:
   - Untuk mencari sumber daya dalam proyek tempat Anda berada saat ini, pilih Proyek ini.
   - Untuk mencari sumber daya dalam semua proyek di ruang tempat Anda berada saat ini, pilih Ruang ini.
4. Lihat hasil pencarian di jendela hasil penelusuran khusus dengan melakukan salah satu hal berikut:
   - Di bagian bawah jendela hasil pencarian cepat, pilih Lihat semua hasil dalam nama proyek | nama spasi untuk melihat semua hasil pencarian.
   - Tekan Enter untuk melihat semua hasil pencarian.

> **Tip**
>
> Sebutkan pengguna proyek lain dalam komentar atau deskripsi permintaan tarik, atau dalam komentar atau deskripsi masalah, dengan menggunakan tanda @ diikuti dengan nama

tampilan atau nama pengguna mereka. Anda juga dapat menautkan ke sumber daya seperti masalah atau file kode dengan menggunakan tanda @ diikuti dengan nama masalah atau file kode.

Menyempurnakan kueri penelusuran Anda

Jika Anda tidak dapat menemukan apa yang Anda cari setelah mencari, Anda dapat memperbaiki pencarian Anda dengan CodeCatalyst bahasa kueri khusus. Bidang individu tidak memiliki batas karakter, tetapi kueri keseluruhan memiliki batas 1.024 karakter.

Topik

Pemurnian berdasarkan jenis

Untuk mempersempit cakupan pencarian Anda ke jenis informasi tertentu, sertakan *`type:result-type`* dalam pencarian Anda, di mana *`jenis hasil`*`code`,`issue`, `project` atau. `user`

Contoh:

- `type:code AND java`— Tampilkan hasil kode di bidang terkait kode yang berisi “java”.

  Untuk informasi selengkapnya, lihat Kolom kode.
- `type:issue AND Bug`— Tampilkan hasil masalah di bidang terkait masalah yang berisi “Bug”.

  Untuk informasi selengkapnya, lihat Bidang masalah.
- `type:user AND MaryMajor`— Tampilkan hasil pengguna di bidang terkait pengguna yang berisi “”MaryMajor.

  Untuk informasi selengkapnya, lihat Bidang pengguna.
- `type:project AND Datafeeder`— Tampilkan hasil proyek yang berisi “Datafeeder”.

  Untuk informasi selengkapnya, lihat Bidang proyek.

Pemurnian berdasarkan lapangan

Untuk mempersempit cakupan pencarian Anda ke bidang tertentu, sertakan *field-name:query* dalam pencarian Anda, di mana *nama bidang*`title`,,, `username`project, dan sebagainya`description`, dan *kueri* adalah teks yang Anda cari. Untuk daftar bidang, lihatReferensi bidang yang dapat dicari. Anda dapat mencari beberapa kueri menggunakan tanda kurung.

Contoh:

- `title:bug`— Tampilkan hasil di mana judul berisi “bug”.
- `username:John`— Tampilkan hasil di mana nama pengguna berisi “John”.
- `project:DataFeeder`— Tampilkan hasil dalam proyek “DataFeeder”. Kueri tidak peka huruf besar/kecil.
- `description:overview`— Tampilkan hasil di mana deskripsi berisi “ikhtisar”.

Penyempurnaan dengan operator Boolean

Untuk menentukan kendala pada frase pencarian, Anda dapat menggunakan operator `AND` Boolean,, dan. `OR` `NOT` Jika Anda mencantumkan beberapa frasa, CodeCatalyst gabungkan dengan secara `OR` default. Anda dapat mengelompokkan frasa pencarian menggunakan tanda kurung.

- `exception AND type:code`— Tampilkan hanya hasil kode untuk “pengecualian”.
- `path:README.md AND repo:ServerlessAPI`— Tampilkan hasil untuk jalur dengan “README.md” di mana repositori diberi nama “ServerlessAPI”.
- `buildspec.yml AND (repo:ServerlessAPI OR ServerlessWebApp)`— Tampilkan hasil untuk “buildspec.yml"di mana repositori adalah “ServerlessAPI” atau “”. ServerlessWebApp
- `path:java NOT (path:py OR path:ts)`— Tampilkan hasil di mana jalur berisi “java” tetapi tidak “py” atau “ts”.

Penyempurnaan berdasarkan proyek

Untuk mempersempit ruang lingkup pencarian Anda ke proyek tertentu, sertakan *project:name AND query* dalam pencarian Anda, di mana *nama* adalah proyek di mana Anda mencari dan *kueri* adalah konten yang Anda cari.

- `project:name AND query`— Tampilkan hasil di mana jalur berisi kueri dan nama proyek.

Pertimbangan saat bekerja dengan pencarian

Pembaruan konten yang tertunda - Diperlukan beberapa menit agar pembaruan konten, seperti perubahan nama atau penggantian masalah, tercermin dalam hasil pencarian. Pembaruan besar, seperti migrasi basis kode, dapat memakan waktu lebih lama untuk muncul di hasil penelusuran.

Melarikan diri dari karakter khusus — Karakter khusus berikut memerlukan pertimbangan khusus dalam permintaan pencarian Anda:. `+ - & && || ! ( ) { } [ ] ^ " ~ * ? : \` Karakter khusus tidak akan memengaruhi kueri, dan Anda harus menghapusnya atau menghindarinya. Untuk menghindari karakter, tambahkan garis miring terbalik (\) di depannya. Misalnya, kueri penelusuran [Feature] harus berupa Feature atau\ [Feature\].

Mempersempit pencarian - Pencarian tidak peka huruf besar/kecil. Pencarian dalam semua huruf kecil mencegah kueri Anda memecah kata pada perubahan huruf kecil. Misalnya, untuk menanyakan `MyService` dan hanya`MyService`, pertimbangkan kueri `myservice` untuk menghindari hasil yang hanya berisi `my` atau`service`.

Pencarian menggabungkan kata dan bagian kata dengan konjungsi OR-wise secara default. Misalnya, `new function` dapat mengembalikan hasil yang berisi keduanya `new` dan `function` dan juga hasil dengan hanya `new` atau`function`. Untuk menghindari yang terakhir, gabungkan beberapa kata dengan`AND`. Misalnya, Anda dapat mencari`new AND function`.

Cabang default — Pencarian hanya akan mengembalikan hasil kode dari komit terbaru pada cabang default repositori sumber. Untuk menemukan kode di cabang atau komit lain, pertimbangkan untuk mengkloning repositori secara lokal, membuka cabang di Lingkungan Pengembang, atau melihat cabang dan detail di UI. CodeCatalyst Mengubah cabang default menghasilkan pembaruan ke file yang dapat ditemukan oleh penelusuran. Untuk informasi selengkapnya, lihat Mengelola cabang default untuk repositori.

Referensi bidang yang dapat dicari

CodeCatalyst mencari bidang berikut saat Anda memasukkan kueri penelusuran. Alias adalah nama lain yang dapat Anda gunakan untuk mereferensikan bidang dalam bahasa kueri lanjutan.

Kolom kode

Bidang	Alias	Deskripsi
Nama cabang	cabang	Nama cabang file kode aktif.
code	N/A	Informasi tentang isi kode dalam bentuk cuplikan kode yang menunjukkan bagian dari kode sumber yang cocok dengan pencarian.
KomiTiD	N/A	ID komit dari komit di mana file kode yang dikembalikan terakhir diperbarui. Mungkin atau mungkin bukan ID komit di ujung nama cabang yang ditentukan dalam. `branchName`
CommitMessage	N/A	Pesan komit dari komit di mana file kode terakhir diperbarui. Mungkin atau mungkin bukan pesan komit di ujung nama cabang yang ditentukan dalam. `branchName` Jika tidak ada pesan komit yang diberikan, nilai ini akan menjadi string kosong.
filePath	path	Jalur file dari file kode ini.
lastUpdatedBy	N/A	CodeCatalyst pengguna yang terakhir memperbarui kode file. Jika nama pengguna tidak tersedia, nilai ini akan menjadi alamat email pengguna seperti

Bidang	Alias	Deskripsi
		yang dikonfigurasi dalam file konfigurasi Git.
lastUpdatedById	N/A	ID unik pengguna yang dihasilkan sistem yang terakhir memperbarui file kode. Jika ID pengguna tidak tersedia, nilai ini mungkin alamat email pengguna.
lastUpdatedTime	N/A	Waktu ketika data pencarian terakhir diperbarui dengan komit yang berisi kode file (dalam stempel waktu universal terkoordinasi (UTC)).
projectId	N/A	ID unik proyek yang dihasilkan sistem.
projectName	ProjectNames, proyek	Menampilkan nama proyek yang berisi repositori sumber tempat file kode telah dilakukan.
RepositoryID	RepoID	ID unik yang dihasilkan sistem dari repositori sumber.
repositoryName	repositori, repo	Menampilkan nama repositori sumber tempat file kode telah dilakukan.

Bidang masalah

Bidang	Alias	Deskripsi
Penerima Tugas	AssigneEID	ID unik yang dihasilkan sistem dari pengguna yang ditetapkan untuk masalah ini.
penerima tugas	penerima tugas	Nama pengguna pengguna yang ditetapkan untuk masalah ini.
DibuatOleh	N/A	Tampilkan nama pengguna yang membuat masalah.
createdById	N/A	ID unik yang dihasilkan sistem dari pengguna yang membuat masalah.
createdTime	N/A	Waktu masalah dibuat (dalam stempel waktu waktu universal terkoordinasi (UTC)).
deskripsi	N/A	Deskripsi masalah.
Diarsipkan	diarsipkan	Nilai Boolean yang menunjukkan apakah akan membuat masalah dalam keadaan diarsipkan.
diblokir	diblokir	Nilai Boolean yang menunjukkan apakah masalah ditandai sebagai diblokir.
Labelelids	LabelId	ID unik yang dihasilkan sistem dari label untuk suatu masalah.

Bidang	Alias	Deskripsi
lastUpdatedBy	N/A	Tampilkan nama penggunaan yang terakhir memperbarui masalah.
lastUpdatedById	N/A	ID unik yang dihasilkan sistem dari pengguna yang terakhir memperbarui masalah.
lastUpdatedTime	N/A	Waktu masalah terakhir diperbarui (dalam stempel waktu universal terkoordinasi (UTC)).
Prioritas	N/A	Prioritas masalah, jika salah satu telah ditugaskan.
projectId	N/A	ID unik proyek yang dihasilkan sistem.
projectName	ProjectNames, proyek	Proyek di mana masalah ini dapat ditemukan.
ShortID	N/A	Pengenal peningkatan otomatis yang dipersingkat untuk masalah ini.
status	N/A	Status masalah yang menunjukkan jika masalah ada di backlog atau kolom di papan.
StatusSid	N/A	Pengidentifikasi sistem status.
title	N/A	Judul masalah.

Bidang proyek

Bidang	Alias	Deskripsi
deskripsi	N/A	Deskripsi proyek.
lastUpdatedTime	N/A	Waktu ketika metadata proyek terakhir diperbarui (dalam stempel waktu universal terkoordinasi (UTC)).
projectName	proyek	Nama proyek di ruang angkasa.
ProjectPath	N/A	Nama proyek yang dapat dirutekan URL, ditentukan selama pembuatan proyek. Digunakan dalam URL yang memerlukan nama proyek.

Bidang pengguna

Bidang	Alias	Deskripsi
displayName	N/A	Nama yang digunakan untuk pengguna di CodeCatalyst. Nama tampilan tidak unik.
Email	N/A	Alamat email pengguna.
lastUpdatedTime	N/A	Waktu ketika metadata pengguna terakhir diperbaru i (dalam stempel waktu universal terkoordinasi (UTC)).
userName	nama pengguna	Nama pengguna yang dipilih oleh pengguna saat mereka mendaftar CodeCatalyst.

Bidang	Alias	Deskripsi
		Tidak seperti nama tampilan, nama pengguna tidak dapat diubah.

Memecahkan Masalah Amazon CodeCatalyst

Informasi berikut dapat membantu Anda memecahkan masalah umum di. CodeCatalyst Anda juga dapat menggunakan laporan CodeCatalyst kesehatan Amazon untuk menentukan apakah ada masalah layanan yang mungkin memengaruhi pengalaman Anda.

Topik

Memecahkan masalah akses umum

Saya lupa kata sandi

Masalah: Saya lupa kata sandi yang saya gunakan untuk ID AWS Builder dan Amazon CodeCatalyst saya.

Kemungkinan perbaikan: Cara termudah untuk memperbaiki masalah ini adalah dengan mengatur ulang kata sandi Anda.

1. Buka Amazon CodeCatalyst dan masukkan alamat Email Anda. Lalu, pilih Lanjutkan.

2. Pilih Lupa kata sandi?
3. Kami akan mengirimi Anda email dengan tautan agar Anda dapat mengubah kata sandi Anda. Jika Anda tidak melihat email di kotak masuk Anda, periksa folder spam Anda.

Beberapa atau semua Amazon CodeCatalyst tidak tersedia

Masalah: Saya menavigasi ke atau mengikuti tautan ke CodeCatalyst konsol, tetapi saya melihat kesalahan.

Kemungkinan perbaikan: Alasan paling umum untuk masalah ini adalah Anda mengikuti tautan ke proyek atau ruang yang belum diundang, atau ada masalah ketersediaan umum dengan layanan. Periksa laporan Kesehatan untuk melihat apakah ada masalah yang diketahui dengan layanan ini. Jika tidak, hubungi orang yang mengundang Anda ke proyek atau ruang dan minta undangan lain. Jika Anda belum diundang ke proyek atau ruang apa pun, Anda dapat mendaftar dan membuat ruang dan proyek Anda sendiri.

Saya tidak dapat membuat proyek di CodeCatalyst

Masalah: Saya ingin membuat proyek, tetapi tombol Buat proyek ditampilkan sebagai tidak tersedia, atau saya menerima pesan kesalahan.

Kemungkinan perbaikan: Alasan paling umum untuk masalah ini adalah Anda masuk ke konsol dengan ID AWS Pembuat yang tidak memiliki peran administrator Space. Anda harus memiliki peran ini untuk membuat proyek di suatu ruang.

Jika Anda memiliki peran ini dan tombol tidak muncul sebagai tersedia, mungkin ada masalah sementara dengan layanan. Segarkan browser Anda dan coba lagi.

Memecahkan masalah dukungan

Saya mendapatkan kesalahan saat mengakses AWS Support Amazon CodeCatalyst

Masalah: Ketika saya memilih CodeCatalyst opsi AWS Support untuk Amazon, saya menerima pesan kesalahan berikut:

`Unable to assume role`

```
To access support cases, you must add the role
AWSRoleForCodeCatalystSupport to the Akun AWS that is the billing account
for the space.
```

Kemungkinan perbaikan: Tambahkan peran yang diperlukan ke Akun AWS akun penagihan untuk ruang tersebut. Akun yang ditetapkan sebagai akun penagihan untuk ruang tersebut menggunakan `AWSRoleForCodeCatalystSupport` peran dan kebijakan `AmazonCodeCatalystSupportAccess` terkelola. Untuk informasi selengkapnya, lihat Membuat AWSRoleForCodeCatalystSupportperan untuk akun dan ruang Anda.

> **Note**
>
> AWSBuilder ID hanya bisa mendapatkan dukungan untuk alias yang diautentikasi dengannya dan hanya untuk sumber daya berdasarkan izin. CodeCatalyst Dukungan Akun dan Penagihan tersedia untuk semua pengguna di ruang tersebut. Namun, pembangun hanya bisa mendapatkan dukungan untuk sumber daya dan informasi yang mereka miliki izin untuk masuk CodeCatalyst.

Saya tidak dapat membuat kasus dukungan teknis untuk ruang saya

Masalah: Saya tidak dapat membuat kasus dukungan teknis untuk ruang saya.

Perbaikan: Paket Business Support atau Enterprise Support perlu ditambahkan ke akun penagihan ruang agar pengguna di ruang tersebut dapat membuat kasus dukungan teknis. Minta administrator ruang Anda untuk menambahkan AWS Support paket ke akun penagihan ruang Anda atau kunjungi https://repost.aws/ untuk bertanya kepada AWS komunitas.

Akun saya untuk kasus dukungan tidak lagi terhubung ke ruang saya di CodeCatalyst

Masalah: Akun saya untuk kasus dukungan tidak lagi terhubung ke ruang saya di CodeCatalyst.

Perbaikan: Jika pengguna dengan peran administrator Space mengganti akun penagihan ruang, ini akan memutuskan AWS Support paket dan semua kasus terkait dari ruang. AWS SupportKasus yang terkait dengan akun penagihan ruang lama tidak akan lagi terlihat di AWS Support Amazon CodeCatalyst. Pengguna root untuk akun penagihan tersebut dapat melihat dan menyelesaikan kasus lama dari AWS Management Console dan dapat mengatur izin IAM agar pengguna lain dapat

melihat dan menyelesaikan kasus lama. AWS Support Anda tidak akan dapat terus mendapatkan dukungan teknis CodeCatalyst dari akun penagihan ruang lama melaluiAWS Management Console, tetapi Anda dapat menerima dukungan teknis untuk layanan lain sampai AWS Support paket Anda dibatalkan.

Untuk informasi selengkapnya, lihat Memperbarui, menyelesaikan, dan membuka kembali kasus Anda di AWS SupportPanduan Pengguna.

Saya tidak dapat membuka kasus dukungan untuk yang lain Layanan AWS di AWS Support Amazon CodeCatalyst

Masalah: Saya tidak dapat membuka kasus dukungan untuk orang lain Layanan AWS CodeCatalyst. AWS Support

Kemungkinan perbaikan: Anda hanya dapat membuka kasus CodeCatalyst dukungan dari AWS Support for CodeCatalyst. Jika Anda memerlukan dukungan untuk layanan atau sumber daya yang digunakan dari CodeCatalyst ke layanan lainAWS, Amazon, atau layanan pihak ketiga lainnya, Anda perlu membuat kasus melalui AWS Management Console atau saluran dukungan layanan pihak ketiga. Untuk informasi selengkapnya, lihat Membuat kasus dukungan dan manajemen kasus di Panduan AWS Support Pengguna.

Beberapa atau semua Amazon CodeCatalyst tidak tersedia

Masalah: Saya menavigasi ke atau mengikuti tautan ke CodeCatalyst konsol, tetapi saya melihat kesalahan.

Kemungkinan perbaikan: Alasan paling umum untuk masalah ini adalah Anda mengikuti tautan ke proyek atau ruang yang belum diundang, atau ada masalah ketersediaan umum dengan layanan. Periksa laporan Kesehatan untuk melihat apakah ada masalah yang diketahui dengan layanan ini. Jika tidak, hubungi orang yang mengundang Anda ke proyek atau ruang dan minta undangan lain. Jika Anda belum diundang ke proyek atau ruang apa pun, Anda dapat mendaftar dan membuat ruang dan proyek Anda sendiri.

Saya tidak dapat membuat proyek di CodeCatalyst

Masalah: Saya ingin membuat proyek, tetapi tombol Buat proyek ditampilkan sebagai tidak tersedia, atau saya menerima pesan kesalahan.

Kemungkinan perbaikan: Alasan paling umum untuk masalah ini adalah Anda masuk ke konsol dengan ID AWS Pembuat yang tidak memiliki peran administrator Space. Anda harus memiliki peran ini untuk membuat proyek di suatu ruang.

Jika Anda memiliki peran ini dan tombol tidak muncul sebagai tersedia, mungkin ada masalah sementara dengan layanan. Segarkan browser Anda dan coba lagi.

Saya ingin mengirimkan umpan balik di CodeCatalyst

Masalah: Saya menemukan bug CodeCatalyst dan saya ingin mengirimkan umpan balik.

Kemungkinan perbaikan: Anda dapat mengirimkan umpan balik langsung di CodeCatalyst.

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Di panel navigasi, pilih Berikan umpan balik.
3. Pilih jenis umpan balik dari menu tarik-turun dan masukkan umpan balik Anda.

Memecahkan masalah dengan repositori sumber

Informasi berikut dapat membantu Anda memecahkan masalah umum dengan repositori sumber di. CodeCatalyst

Topik

- Permintaan tarik tidak menampilkan perubahan yang saya harapkan
- Permintaan tarik menunjukkan status Tidak dapat digabungkan

Saya telah mencapai penyimpanan maksimum untuk ruang saya dan melihat peringatan atau kesalahan

Masalah: Saya ingin mengkomit kode ke satu atau lebih repositori sumber di CodeCatalyst, tetapi saya melihat kesalahan. Di konsol, saya melihat pesan di halaman repositori sumber bahwa saya telah mencapai batas penyimpanan untuk ruang tersebut.

Kemungkinan perbaikan: Bergantung pada peran Anda dalam proyek atau ruang, Anda dapat mengurangi ukuran satu atau lebih repositori sumber Anda, menghapus repositori sumber yang tidak digunakan, atau mengubah tingkat penagihan Anda menjadi yang memiliki lebih banyak penyimpanan.

- Untuk mengurangi ukuran repositori sumber dalam proyek, Anda dapat menghapus cabang yang tidak digunakan. Lihat informasi yang lebih lengkap di Menghapus cabang dan Peran kontributor.
- Untuk mengurangi penyimpanan keseluruhan untuk suatu ruang, Anda dapat menghapus repositori sumber yang tidak digunakan. Lihat informasi yang lebih lengkap di Menghapus repositori sumber dan Peran administrator proyek.
- Untuk meningkatkan jumlah penyimpanan yang tersedia untuk ruang Anda, Anda dapat mengubah tingkat penagihan menjadi satu dengan penyimpanan lebih banyak. Untuk informasi selengkapnya, lihat Mengubah tingkat CodeCatalyst penagihan di Panduan CodeCatalyst Administrator Amazon.

Saya menerima kesalahan saat mencoba mengkloning atau mendorong ke repositori CodeCatalyst sumber Amazon

Masalah: Ketika saya mencoba mengkloning repositori sumber ke komputer lokal atau ke lingkungan pengembangan terintegrasi (IDE), saya menerima kesalahan izin.

Kemungkinan perbaikan: Anda mungkin tidak memiliki token akses pribadi (PAT) untuk ID AWS Builder Anda, Anda mungkin belum mengonfigurasi sistem manajemen kredensi Anda dengan PAT Anda, atau PAT Anda mungkin telah kedaluwarsa. Coba satu atau lebih solusi berikut:

- Buat token akses pribadi (PAT). Untuk informasi selengkapnya, lihat Berikan akses repositori pengguna dengan token akses pribadi.

- Pastikan Anda telah menerima undangan untuk proyek yang berisi repositori sumber dan bahwa Anda masih anggota proyek itu. Anda tidak dapat mengkloning repositori sumber jika Anda bukan anggota aktif dari proyek itu. Masuk ke konsol dan coba navigasikan ke ruang dan proyek tempat Anda mencoba mengkloning repositori sumber. Jika Anda tidak dapat melihat proyek dalam daftar proyek untuk ruang tersebut, Anda bukan anggota proyek itu, atau Anda belum menerima undangan untuk proyek itu. Untuk informasi selengkapnya, lihat Menerima undangan dan membuat AWS Builder ID.
- Pastikan perintah clone Anda diformat dengan benar dan menyertakan AWS Builder ID Anda. Sebagai contoh:

```
https://LiJuan@git.us-west-2.codecatalyst.aws/
v1/ExampleCorp/MyExampleProject/MyExampleRepo
```

- Gunakan AWS CLI untuk memastikan bahwa Anda memiliki PAT yang terkait dengan AWS Builder ID Anda, dan bahwa itu tidak kedaluwarsa. Jika Anda tidak memilikinya atau PAT kedaluwarsa, buat satu. Untuk informasi selengkapnya, lihat Berikan akses repositori pengguna dengan token akses pribadi.
- Coba buat Lingkungan Dev untuk bekerja dengan kode di repositori sumber alih-alih mengkloningnya ke repo atau IDE lokal. Untuk informasi selengkapnya, lihat Membuat Lingkungan Dev.

Saya menerima kesalahan saat mencoba melakukan atau mendorong ke repositori CodeCatalyst sumber Amazon

Masalah: Ketika saya mencoba mendorong ke repositori sumber, saya menerima kesalahan izin.

Kemungkinan perbaikan: Anda mungkin tidak memiliki peran dalam proyek yang memungkinkan Anda untuk melakukan dan mendorong perubahan kode ke proyek. Lihat peran Anda dalam proyek tempat Anda mencoba mendorong perubahan ke repositori sumber. Lihat informasi yang lebih lengkap di Mendapatkan daftar anggota dan peran proyek mereka dan Memberikan akses dengan peran pengguna.

Jika Anda memiliki peran yang memungkinkan melakukan dan mendorong perubahan, cabang tempat Anda mencoba melakukan perubahan mungkin memiliki aturan cabang yang dikonfigurasi untuk itu yang mencegah Anda mendorong perubahan kode ke cabang itu. Coba buat cabang dan dorong kode Anda ke cabang itu sebagai gantinya. Untuk informasi selengkapnya, lihat Mengelola tindakan yang diizinkan untuk cabang dengan aturan cabang.

Saya memerlukan repositori sumber untuk proyek saya

Masalah: Proyek saya tidak memiliki repositori sumber, atau saya memerlukan repositori sumber lain untuk proyek saya.

Kemungkinan perbaikan: Beberapa proyek dibuat tanpa sumber daya apa pun. Jika Anda adalah anggota proyek, Anda dapat membuat repositori sumber untuk proyek itu di. CodeCatalyst Jika seseorang dengan peran administrator Space menginstal GitHub Repositori dan menghubungkannya ke GitHub akun, Anda dapat menautkan ke GitHub repositori yang tersedia untuk menambahkannya ke proyek Anda jika Anda memiliki peran administrator Project. Untuk informasi selengkapnya, lihat Membuat repositori sumber dan Menautkan repositori sumber.

Repositori sumber saya baru tetapi berisi komit

Masalah: Saya baru saja membuat repositori sumber. Itu harus kosong, tetapi memiliki komit, cabang, dan `README.md` file di dalamnya.

Kemungkinan perbaikan: Ini adalah perilaku yang diharapkan. Semua repositori sumber CodeCatalyst termasuk komit awal yang menetapkan cabang default ke `main` dan menyertakan kode sampel (jika repositori dibuat untuk proyek menggunakan cetak biru yang menyertakan kode sampel) atau file penurunan harga templat untuk file README repositori. Anda dapat membuat cabang tambahan di konsol dan di klien Git. Anda dapat membuat dan mengedit file di konsol, dan menghapus file di lingkungan Dev dan klien Git.

Saya ingin cabang yang berbeda sebagai cabang default saya

Masalah: Repositori sumber saya datang dengan cabang default bernama`main`, tetapi saya ingin cabang yang berbeda sebagai cabang default saya.

Kemungkinan perbaikan: Anda tidak dapat mengubah atau menghapus cabang default di repositori sumber di. CodeCatalyst Anda dapat membuat cabang tambahan dan menggunakan cabang tersebut dalam tindakan sumber dalam alur kerja. Anda juga dapat memilih untuk menautkan GitHub repositori dan menggunakannya sebagai repositori untuk proyek Anda.

Saya menerima email tentang aktivitas dalam permintaan tarik

Masalah: Saya tidak mendaftar atau mengonfigurasi pemberitahuan email tentang aktivitas permintaan tarik, tetapi saya tetap menerimanya.

Kemungkinan perbaikan: Pemberitahuan email dikirim secara otomatis tentang aktivitas permintaan tarik. Untuk informasi selengkapnya, lihat Meninjau kode dengan permintaan tarik di Amazon CodeCatalyst.

Saya lupa token akses pribadi saya (PAT)

Masalah: Saya telah menggunakan PAT untuk mengkloning, mendorong, dan menarik kode untuk repositori sumber, tetapi saya telah kehilangan nilai untuk token saya, dan saya tidak dapat menemukannya di konsol. CodeCatalyst

Kemungkinan perbaikan: Cara tercepat untuk mengatasi masalah ini adalah dengan membuat PAT lain dan mengonfigurasi manajer kredensi atau IDE Anda untuk menggunakan PAT baru ini. Kami hanya menampilkan nilai PAT saat Anda membuatnya. Jika Anda kehilangan nilai ini, itu tidak dapat diambil. Untuk informasi selengkapnya, lihat Berikan akses repositori pengguna dengan token akses pribadi.

Permintaan tarik tidak menampilkan perubahan yang saya harapkan

Masalah: Saya membuat permintaan tarik, tetapi saya tidak melihat perubahan yang saya harapkan untuk dilihat antara cabang sumber dan tujuan.

Kemungkinan perbaikan: Ini mungkin disebabkan oleh sejumlah masalah. Coba satu atau lebih solusi berikut:

- Anda mungkin meninjau perubahan di antara revisi lama, atau Anda mungkin tidak melihat perubahan terbaru. Segarkan browser Anda dan pastikan bahwa Anda telah memilih perbandingan antara revisi yang ingin Anda lihat.
- Tidak semua perubahan dalam permintaan tarik dapat ditampilkan di konsol. Misalnya, Anda tidak dapat melihat submodul Git di konsol, sehingga Anda tidak dapat melihat perbedaan dalam submodul dalam permintaan tarik. Beberapa perbedaan mungkin terlalu besar untuk ditampilkan. Lihat informasi yang lebih lengkap di Kuota untuk repositori sumber di CodeCatalyst dan Melihat file.
- Permintaan tarik menampilkan perbedaan antara basis gabungan dan revisi apa pun yang Anda pilih. Saat Anda membuat permintaan tarik, perbedaan yang ditampilkan untuk Anda adalah perbedaan antara ujung cabang sumber dan ujung cabang tujuan. Setelah permintaan tarik dibuat, perbedaan yang ditampilkan adalah antara revisi dan basis penggabungannya. Basis penggabungan adalah komit yang merupakan ujung cabang tujuan saat revisi dibuat. Basis penggabungan dapat berubah di antara revisi. Untuk informasi selengkapnya tentang perbedaan dan penggabungan basis di Git, lihat git-merge-basedi dokumentasi Git.

Permintaan tarik menunjukkan status Tidak dapat digabungkan

Masalah: Saya ingin menggabungkan permintaan tarik, tetapi statusnya ditampilkan sebagai Tidak dapat digabungkan.

Kemungkinan perbaikan: Ini dapat disebabkan oleh satu atau lebih masalah:

- Semua pengulas yang diperlukan untuk permintaan tarik Anda harus menyetujui permintaan tarik sebelum dapat digabungkan. Tinjau daftar pengulas yang diperlukan untuk setiap pengulas dengan ikon jam di sebelah nama. Ikon jam menunjukkan bahwa pengulas belum menyetujui permintaan tarik.

  > **Note**
  >
  > Jika peninjau yang diperlukan telah dihapus dari proyek Anda sebelum menyetujui permintaan tarik, Anda tidak dapat menggabungkan permintaan tarik. Tutup permintaan tarik dan buat permintaan tarik baru.

- Mungkin ada konflik gabungan antara cabang sumber dan cabang tujuan. CodeCatalyst tidak mendukung semua kemungkinan strategi dan opsi penggabungan Git. Anda dapat mengevaluasi cabang untuk konflik gabungan di Lingkungan Pengembang atau mengkloning repositori dan menggunakan alat IDE atau Git untuk menemukan dan menyelesaikan konflik gabungan. Untuk informasi selengkapnya, lihat Menggabungkan permintaan tarik.

Memecahkan masalah proyek dan cetak biru

Bagian ini dapat membantu Anda memecahkan masalah umum yang mungkin Anda temui saat bekerja dengan proyek dan cetak biru di Amazon. CodeCatalyst

Java API dengan AWS Fargate blueprint kehilangan dependensi untuk apache-maven-3.8.6

Masalah: Untuk proyek yang dibuat dari Java API dengan AWS Fargate cetak biru, alur kerja gagal dengan kesalahan untuk dependensi yang hilang. `apache-maven-3.8.6` Alur kerja gagal dengan output yang mirip dengan contoh berikut:

```
Step 8/25 : RUN wget https://dlcdn.apache.org/maven/maven-3/3.8.6/binaries/apache-
maven-3.8.6-bin.tar.gz -P /tmp
```

```
---> Running in 1851ce6f4d1b
[91m--2023-03-10 01:24:55--  https://dlcdn.apache.org/maven/maven-3/3.8.6/binaries/
apache-maven-3.8.6-bin.tar.gz
[0m[91mResolving dlcdn.apache.org (dlcdn.apache.org)...
[0m[91m151.101.2.132, 2a04:4e42::644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443...
[0m[91mconnected.
[0m[91mHTTP request sent, awaiting response... [0m[91m404 Not Found
2023-03-10 01:24:55 ERROR 404: Not Found.
[0mThe command '/bin/sh -c wget https://dlcdn.apache.org/maven/maven-3/3.8.6/binaries/
apache-maven-3.8.6-bin.tar.gz -P /tmp' returned a non-zero code: 8
[Container] 2023/03/10 01:24:55 Command failed with exit status 8
```

Solusi: Perbarui cetak biru Dockerfile menggunakan langkah-langkah berikut.

1. Di bilah pencarian, masukkan `apache-maven-3.8.6` untuk menemukan dockerfile di dalam proyek yang dibuat dengan Java API dengan AWS Fargate cetak biru.
2. Perbarui Dockerfile (`/static-assets/app/Dockerfile`) untuk digunakan `maven:3.9.0-amazoncorretto-11` sebagai gambar dasar dan hapus ketergantungan pada paket. `apache-maven-3.8.6`
3. (Disarankan) Kami juga merekomendasikan memperbarui ukuran tumpukan Maven menjadi 6 GB.

Di bawah ini adalah contoh Dockerfile.

```
FROM maven:3.9.0-amazoncorretto-11 AS builder

COPY ./pom.xml ./pom.xml
COPY src ./src/

ENV MAVEN_OPTS='-Xmx6g'

RUN mvn -Dmaven.test.skip=true clean package

FROM amazoncorretto:11-alpine

COPY —from=builder target/CustomerService-0.0.1.jar CustomerService-0.0.1.jar
EXPOSE 80
```

```
CMD ["java","-jar","-Dspring.profiles.active=prod","/CustomerService-0.0.1.jar", "—
server.port=80"]
```

Alur kerja cetak biru aplikasi web tiga tingkat modern OnPullRequestgagal dengan kesalahan izin untuk Amazon CodeGuru

Masalah: Ketika saya mencoba menjalankan alur kerja untuk proyek saya, alur kerja gagal dijalankan dengan pesan berikut:

```
Failed at codeguru_codereview: The action failed during runtime. View the action's logs
 for more details.
```

Solusi: Salah satu kemungkinan penyebab kegagalan tindakan ini mungkin karena izin yang hilang dalam kebijakan peran IAM, di mana versi peran layanan yang digunakan oleh CodeCatalyst dalam koneksi Akun AWS tidak memiliki izin yang diperlukan agar tindakan codeguru_codereview berjalan dengan sukses. Untuk memperbaiki masalah ini, peran layanan harus diperbarui dengan izin yang diperlukan, atau Anda harus mengubah peran layanan yang digunakan untuk alur kerja menjadi peran yang memiliki izin yang diperlukan untuk Amazon dan CodeGuru Amazon Reviewer. CodeGuru Dengan menggunakan langkah-langkah berikut, temukan peran Anda dan perbarui izin kebijakan peran untuk memungkinkan alur kerja berjalan dengan sukses.

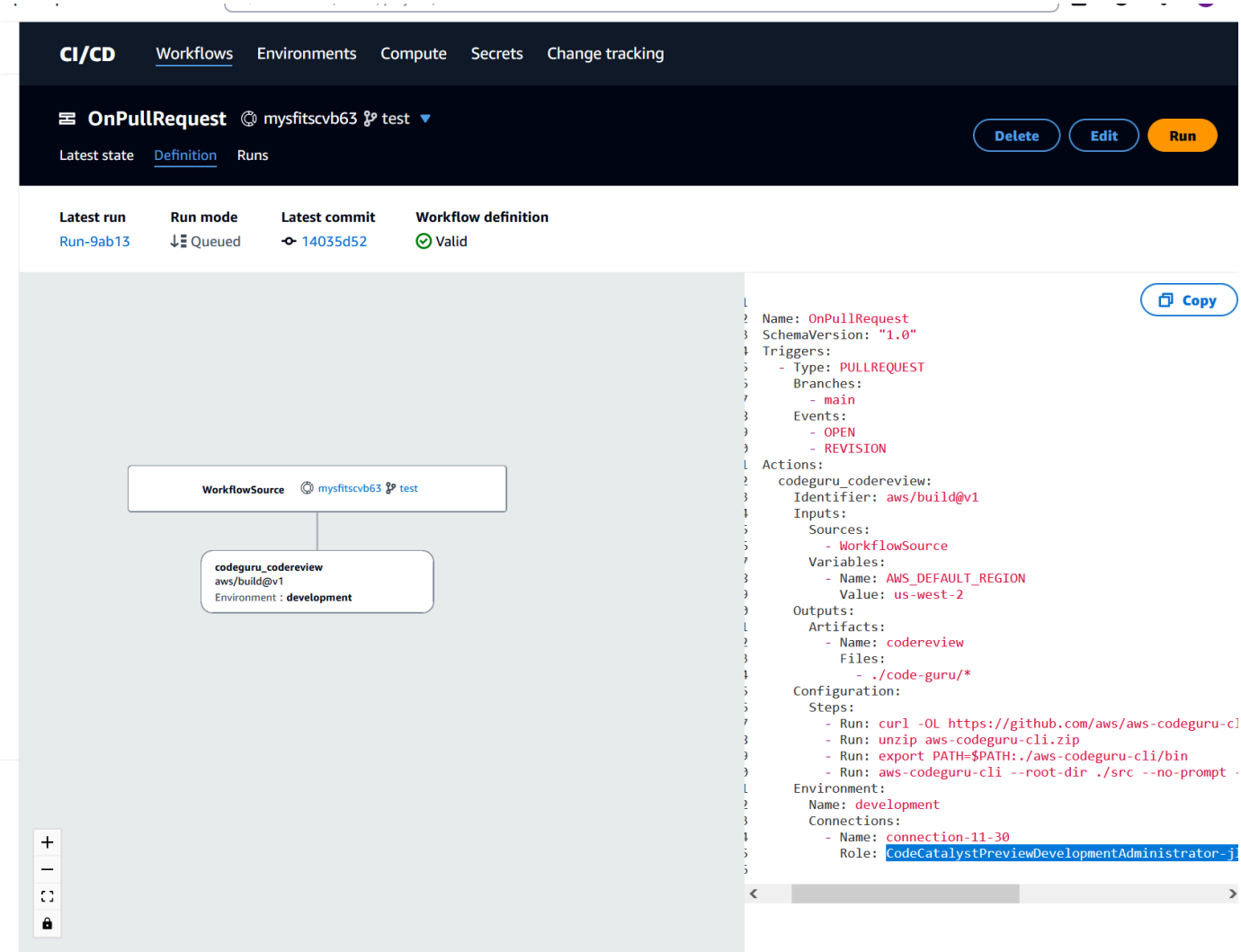
> **Note**
>
> Langkah-langkah ini berlaku untuk alur kerja berikut di CodeCatalyst:
>
> - OnPullRequestAlur kerja disediakan untuk proyek yang dibuat dengan cetak biru aplikasi web tiga tingkat Modern di. CodeCatalyst
> - Alur kerja ditambahkan ke proyek CodeCatalyst dengan tindakan yang mengakses Amazon CodeGuru atau Amazon CodeGuru Reviewer.

Setiap proyek berisi alur kerja dengan tindakan yang menggunakan peran dan lingkungan yang disediakan oleh yang Akun AWS terhubung ke proyek Anda. CodeCatalyst Alur kerja dengan tindakan dan kebijakan yang ditunjuk disimpan di repositori sumber Anda di direktori /.codecatalyst/workflows. Memodifikasi alur kerja YAMAL tidak diperlukan kecuali Anda menambahkan ID peran baru ke alur kerja yang ada. Untuk informasi tentang elemen dan pemformatan template YAMAL, lihat. Alur kerja definisi YAMAL

Ini adalah langkah-langkah tingkat tinggi yang harus diikuti untuk mengedit kebijakan peran Anda dan memverifikasi alur kerja YAMAL.

Untuk mereferensikan nama peran Anda di alur kerja YAMAL dan memperbarui kebijakan

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda. Arahkan ke proyek Anda.
3. Pilih CI/CD, lalu pilih Alur kerja.
4. Pilih alur kerja berjudul OnPullRequest. Pilih tab Ketentuan.
5. Dalam alur kerja YAMAL, di `Role:` bidang di bawah tindakan codeguru_codereview, buat catatan nama peran. Ini adalah peran dengan kebijakan yang akan Anda ubah di IAM. Contoh berikut menunjukkan nama peran.



6. Lakukan salah satu dari berikut:

- (Disarankan) Perbarui peran layanan yang terhubung ke project Anda dengan izin yang diperlukan untuk Amazon CodeGuru dan Amazon CodeGuru Reviewer. Peran akan memiliki nama `CodeCatalystWorkflowDevelopmentRole-spaceName` dengan pengenal unik ditambahkan. Untuk informasi selengkapnya tentang kebijakan peran dan peran, lihatMemahami peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan. Lanjutkan ke langkah selanjutnya untuk memperbarui kebijakan di IAM.

  > **Note**
  >
  > Anda harus memiliki akses AWS administrator ke Akun AWS dengan peran dan kebijakan.

- Ubah peran layanan yang digunakan untuk alur kerja menjadi peran yang memiliki izin yang diperlukan untuk Amazon dan CodeGuru Amazon CodeGuru Reviewer atau buat peran baru dengan izin yang diperlukan.

7. Masuk ke AWS Management Console dan buka konsol IAM di https://console.aws.amazon.com/iam/.

   Di konsol IAM, temukan peran dari langkah 5, seperti`CodeCatalystPreviewDevelopmentRole`.

8. Dalam peran dari langkah 5, ubah kebijakan izin untuk menyertakan `codeguru-reviewer:*` dan `codeguru:*` izin. Setelah menambahkan izin ini, kebijakan izin akan terlihat mirip dengan yang berikut:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "cloudformation:*",
                "lambda:*",
                "apigateway:*",
                "ecr:*",
                "ecs:*",
                "ssm:*",
                "codedeploy:*",
                "s3:*",
                "iam:DeleteRole",
                "iam:UpdateRole",
```

```
                "iam:Get*",
                "iam:TagRole",
                "iam:PassRole",
                "iam:CreateRole",
                "iam:AttachRolePolicy",
                "iam:DetachRolePolicy",
                "iam:PutRolePolicy",
                "iam:CreatePolicy",
                "iam:DeletePolicy",
                "iam:CreatePolicyVersion",
                "iam:DeletePolicyVersion",
                "iam:PutRolePermissionsBoundary",
                "iam:DeleteRolePermissionsBoundary",
                "sts:AssumeRole",
                "elasticloadbalancing:DescribeTargetGroups",
                "elasticloadbalancing:DescribeListeners",
                "elasticloadbalancing:ModifyListener",
                "elasticloadbalancing:DescribeRules",
                "elasticloadbalancing:ModifyRule",
                "cloudwatch:DescribeAlarms",
                "sns:Publish",
                "sns:ListTopics",
                "codeguru-reviewer:*",
                "codeguru:*"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

9. Setelah Anda melakukan koreksi kebijakan, kembali ke CodeCatalyst dan mulai menjalankan alur kerja lagi.

Masih mencari untuk memecahkan masalah Anda?

Anda dapat pergi ke Amazon CodeCatalyst atau mengisi formulir Support Feedback. Di bagian Permintaan informasi, di bawah Bagaimana kami dapat membantu Anda, sertakan bahwa Anda adalah CodeCatalyst pelanggan Amazon. Berikan detail sebanyak mungkin sehingga kami dapat mengatasi masalah Anda dengan paling efisien.

Memecahkan masalah dengan alur kerja

Lihat bagian berikut untuk memecahkan masalah yang terkait dengan alur kerja di Amazon. CodeCatalyst Untuk informasi lebih lanjut tentang alur kerja, lihat Bangun, uji, dan terapkan dengan alur kerja di CodeCatalyst.

Topik

Bagaimana cara memperbaiki pesan “Alur kerja tidak aktif”?

Masalah: Di CodeCatalyst konsol, di bawah CI/CD, Alur kerja, alur kerja Anda muncul dengan pesan berikut:

```
Workflow is inactive.
```

Pesan ini menunjukkan bahwa file definisi alur kerja berisi pemicu yang tidak berlaku untuk cabang yang Anda gunakan saat ini. Misalnya, file definisi alur kerja Anda mungkin berisi PUSH pemicu yang mereferensikan `main` cabang Anda, tetapi Anda berada di cabang fitur. Karena perubahan yang Anda buat di cabang fitur tidak berlaku`main`, dan tidak akan memulai alur kerja berjalan`main`, CodeCatalyst nonaktifkan alur kerja di cabang dan tandai sebagai. `Inactive`

Kemungkinan perbaikan:

Jika Anda ingin memulai alur kerja di cabang fitur Anda, Anda dapat melakukan hal berikut:

- Di cabang fitur Anda, dalam file definisi alur kerja, hapus `Branches` properti dari `Triggers` bagian sehingga terlihat seperti ini:

  ```
  Triggers:
    - Type: PUSH
  ```

  Konfigurasi ini menyebabkan pemicu diaktifkan saat push ke cabang mana pun, termasuk cabang fitur Anda. Jika pemicu diaktifkan, CodeCatalyst akan memulai alur kerja berjalan menggunakan file definisi alur kerja dan file sumber di cabang apa pun yang Anda dorong.
- Di cabang fitur Anda, dalam file definisi alur kerja, hapus `Triggers` bagian dan jalankan alur kerja secara manual.
- Di cabang fitur Anda, dalam file definisi alur kerja, ubah PUSH bagian tersebut sehingga mereferensikan cabang fitur Anda daripada cabang lain (seperti`main`, misalnya).

> **Important**
>
> Berhati-hatilah untuk tidak melakukan perubahan ini jika Anda tidak bermaksud menggabungkannya untuk kembali ke `main` cabang Anda.

Untuk informasi selengkapnya tentang mengedit file definisi alur kerja, lihatMembuat alur kerja.

Untuk informasi lebih lanjut tentang menggunakan pemicu, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.

Bagaimana cara memperbaiki kesalahan “Definisi alur kerja memiliki *n* kesalahan”?

Masalah: Anda melihat salah satu pesan kesalahan berikut:

Kesalahan 1:

Di halaman CI/CD, alur kerja, di bawah nama alur kerja Anda, Anda melihat:

`Workflow definition has n errors`

Kesalahan 2:

Saat mengedit alur kerja, Anda memilih tombol Validasi dan pesan berikut muncul di bagian atas konsol: CodeCatalyst

`The workflow definition has errors. Fix the errors and choose Validate to verify your changes.`

Kesalahan 3:

Setelah menavigasi ke halaman detail alur kerja Anda, Anda melihat kesalahan berikut di bidang definisi Alur Kerja:

`n errors`

Kemungkinan perbaikan:

- Pilih CI/CD, pilih Alur kerja, dan pilih nama alur kerja yang memiliki kesalahan. Di bidang Definisi alur kerja di dekat bagian atas, pilih tautan ke kesalahan. Detail tentang kesalahan muncul di bagian bawah halaman. Ikuti tips pemecahan masalah dalam kesalahan untuk memperbaiki masalah.
- Pastikan bahwa file definisi alur kerja adalah file YAMG.
- Pastikan properti YAMB dalam file definisi alur kerja bersarang pada tingkat yang tepat. Untuk melihat bagaimana properti harus disarangkan dalam file definisi alur kerja, lihatAlur kerja definisi YAMAL, atau lihat dokumentasi tindakan Anda, yang ditautkan ke from. Menambahkan tindakan ke CodeCatalyst alur kerja
- Pastikan tanda bintang (*) dan karakter khusus lainnya lolos dengan benar. Untuk menghindarinya, tambahkan tanda kutip tunggal atau ganda. Sebagai contoh:

```
Outputs:
  Artifacts:
    - Name: myartifact
      Files:
        - "**/*"
```

Untuk informasi selengkapnya tentang karakter khusus dalam file definisi alur kerja, lihatPedoman dan konvensi sintaks.

- Pastikan properti YAMB dalam file definisi alur kerja menggunakan kapitalisasi yang tepat. Untuk informasi lebih lanjut tentang aturan casing, lihatPedoman dan konvensi sintaks. Untuk menentukan casing yang benar dari setiap properti, lihatAlur kerja definisi YAMAL, atau konsultasikan dokumentasi tindakan Anda, yang terkait dengan dariMenambahkan tindakan ke CodeCatalyst alur kerja.
- Pastikan bahwa `SchemaVersion` properti hadir dan diatur ke versi yang benar dalam file definisi alur kerja. Untuk informasi selengkapnya, lihat SchemaVersion.
- Pastikan bahwa `Triggers` bagian dalam file definisi alur kerja mencakup semua properti yang diperlukan. Untuk menentukan properti yang diperlukan, pilih pemicu di editor visual dan cari bidang yang tidak memiliki informasi, atau lihat dokumentasi referensi pemicu diTriggers.
- Pastikan bahwa `DependsOn` properti dalam file definisi alur kerja dikonfigurasi dengan benar dan tidak memperkenalkan dependensi melingkar. Untuk informasi selengkapnya, lihat Mengkonfigurasi tindakan untuk bergantung pada tindakan lain.
- Pastikan bahwa `Actions` bagian dalam file definisi alur kerja menyertakan setidaknya satu tindakan. Untuk informasi selengkapnya, lihat Tindakan.
- Pastikan bahwa setiap tindakan mencakup semua properti yang diperlukan. Untuk menentukan properti yang diperlukan, pilih tindakan di editor visual dan cari bidang yang tidak memiliki informasi, atau lihat dokumentasi tindakan Anda, yang ditautkan ke dariMenambahkan tindakan ke CodeCatalyst alur kerja.
- Pastikan bahwa semua artefak input memiliki artefak keluaran yang sesuai. Untuk informasi selengkapnya, lihat Mendefinisikan artefak keluaran.
- Pastikan bahwa variabel yang didefinisikan dalam satu tindakan diekspor sehingga mereka dapat digunakan dalam tindakan lain. Untuk informasi selengkapnya, lihat Mengekspor variabel sehingga tindakan lain dapat menggunakannya.

Bagaimana cara memperbaiki kesalahan “Tidak dapat menemukan kredensil” dan “ExpiredToken”?

Masalah: Saat bekerjaTutorial: Menyebarkan aplikasi ke Amazon EKS, Anda melihat salah satu atau kedua pesan kesalahan berikut di jendela terminal mesin pengembangan Anda:

```
Unable to locate credentials. You can configure credentials by running "aws configure".
```

```
ExpiredToken: The security token included in the request is expired
```

Kemungkinan perbaikan:

Kesalahan ini menunjukkan bahwa kredensil yang Anda gunakan untuk mengakses AWS layanan telah kedaluwarsa. Dalam hal ini, jangan jalankan `aws configure` perintah. Sebagai gantinya, gunakan petunjuk berikut untuk menyegarkan kunci AWS akses dan token sesi Anda.

Untuk menyegarkan kunci AWS akses dan token sesi

1. Pastikan Anda memiliki URL portal AWS akses, nama pengguna, dan kata sandi untuk pengguna yang Anda gunakan untuk melengkapi tutorial Amazon EKS (`codecatalyst-eks-user`). Anda seharusnya telah mengkonfigurasi item ini ketika Anda menyelesaikan Langkah 1: Siapkan mesin pengembangan Anda tutorial.

   > **Note**
   >
   > Jika Anda tidak memiliki informasi ini, buka halaman `codecatalyst-eks-user` detail di Pusat Identitas IAM, pilih Atur ulang kata sandi, Hasilkan kata sandi satu kali [...] , dan Atur ulang kata sandi lagi untuk menampilkan informasi di layar.

2. Lakukan salah satu hal berikut ini:

   - Rekatkan URL portal AWS akses ke bilah alamat browser Anda.

     Atau

   - Segarkan halaman portal AWS akses jika sudah dimuat.

3. Masuk dengan nama `codecatalyst-eks-user` pengguna dan kata sandi, jika Anda belum masuk.

4. Pilih Akun AWS, lalu pilih nama yang Akun AWS Anda tetapkan `codecatalyst-eks-user` pengguna dan set izin.
5. Di samping nama set izin (`codecatalyst-eks-permission-set`), pilih Baris perintah atau akses terprogram.
6. Salin perintah di tengah halaman. Mereka terlihat mirip dengan yang berikut:

   ```
   export AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
   export AWS_SECRET_ACCESS_KEY="wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
   export AWS_SESSION_TOKEN="session-token"
   ```

   ... di mana *`sesi-token`* adalah string acak panjang.
7. Tempelkan perintah ke prompt terminal Anda di mesin pengembangan Anda dan tekan Enter.

   Kunci baru dan token sesi dimuat.

   Anda sekarang telah menyegarkan kredensialnya. `kubectl`Perintah AWS CLI`eksctl`,, dan sekarang harus berfungsi.

Bagaimana cara memperbaiki kesalahan “Tidak dapat terhubung ke server”?

Masalah: Saat mengerjakan tutorial yang dijelaskan di[Tutorial: Menyebarkan aplikasi ke Amazon EKS](), Anda melihat pesan kesalahan yang mirip dengan yang berikut ini di jendela terminal mesin pengembangan Anda:

`Unable to connect to the server: dial tcp: lookup` *`long-string`*`.gr7.us-west-2.eks.amazonaws.com on` *`1.2.3.4:5`*`: no such host`

Kemungkinan perbaikan:

Kesalahan ini biasanya menunjukkan bahwa kredenal yang digunakan `kubectl` utilitas untuk terhubung ke kluster Amazon EKS Anda telah kedaluwarsa. Untuk mengatasi masalah ini, segarkan kredensi dengan memasukkan perintah berikut di prompt terminal:

```
aws eks update-kubeconfig --name codecatalyst-eks-cluster --region us-west-2
```

Di mana:

- *codecatalyst-eks-cluster*diganti dengan nama cluster Amazon EKS Anda.
- *us-west-2* diganti dengan Wilayah AWS tempat cluster Anda digunakan.

Mengapa CodeDeploy bidang hilang dari editor visual?

Masalah: Anda menggunakan tindakan Deploy to Amazon ECS, dan Anda tidak melihat CodeDeploy bidang seperti CodeDeploy AppSpecdi editor visual alur kerja. Masalah ini dapat terjadi karena layanan Amazon ECS yang Anda tentukan di bidang Layanan tidak dikonfigurasi untuk melakukan penerapan biru/hijau.

Kemungkinan perbaikan:

- Pilih layanan Amazon ECS yang berbeda pada tab Konfigurasi aksi Deploy to Amazon ECS. Untuk informasi selengkapnya, lihat Menyebarkan aplikasi ke Amazon Elastic Container Service (ECS) dengan alur kerja.
- Konfigurasikan layanan Amazon ECS yang dipilih untuk melakukan penerapan biru/hijau. Untuk informasi selengkapnya tentang mengonfigurasi penerapan biru/hijau, lihat Penerapan Biru/Hijau dengan CodeDeploy di Panduan Pengembang Layanan Kontainer Elastis Amazon.

Bagaimana cara memperbaiki kesalahan kemampuan IAM?

Masalah: Anda menggunakan tindakan AWS CloudFormation tumpukan Deploy, dan Anda melihat `##[error] requires capabilities: [capability-name]` di log tindakan AWS CloudFormation tumpukan Deploy Anda.

Kemungkinan perbaikan: Selesaikan prosedur berikut untuk menambahkan kemampuan ke file definisi alur kerja. Untuk informasi selengkapnya tentang kemampuan IAM, lihat Mengakui sumber daya IAM dalam AWS CloudFormation templat di Panduan Pengguna IAM.

Visual

Untuk menambahkan kemampuan IAM menggunakan editor visual

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.

4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih Visual.
7. Dalam diagram alur kerja, pilih tindakan AWS CloudFormation tumpukan Deploy Anda.
8. Pilih tab Konfigurasi.
9. Di bagian bawah, pilih Advanced - opsional.
10. Dalam daftar drop-down Kemampuan, pilih kotak centang di samping kemampuan yang disebutkan dalam pesan kesalahan. Jika kemampuan tidak tersedia dalam daftar, gunakan editor YAMB untuk menambahkannya.
11. (Opsional) Pilih Validasi untuk memvalidasi kode YAMB alur kerja sebelum melakukan.
12. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.
13. Jika alur kerja baru tidak dimulai secara otomatis, jalankan alur kerja secara manual untuk melihat apakah perubahan memperbaiki kesalahan. Untuk informasi selengkapnya tentang menjalankan alur kerja secara manual, lihatMemulai alur kerja berjalan secara manual.

YAML

Untuk menambahkan kemampuan IAM menggunakan editor YAMG

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Pilih proyek Anda.
3. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.
4. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.
5. Pilih Edit.
6. Pilih YAMG.
7. Dalam aksi AWS CloudFormation tumpukan Deploy, tambahkan `capabilities` properti, seperti ini:

```
DeployCloudFormationStack:
  Configuration:
    capabilities: capability-name
```

Ganti *capability-name dengan nama* kemampuan IAM yang ditunjukkan dalam pesan kesalahan. Gunakan koma dan tidak ada spasi untuk mencantumkan beberapa kemampuan. Untuk informasi lebih lanjut, lihat deskripsi `capabilities` properti diTindakan “Terapkan AWS CloudFormation tumpukan” definisi YAMAL.

8. (Opsional) Pilih Validasi untuk memvalidasi kode YAMB alur kerja sebelum melakukan.
9. Pilih Komit, masukkan pesan komit, dan pilih Komit lagi.
10. Jika alur kerja baru tidak dimulai secara otomatis, jalankan alur kerja secara manual untuk melihat apakah perubahan memperbaiki kesalahan. Untuk informasi selengkapnya tentang menjalankan alur kerja secara manual, lihatMemulai alur kerja berjalan secara manual.

Bagaimana cara memperbaiki kesalahan “npm install”?

Masalah: Tindakan AWS CDK penerapan atau tindakan AWS CDK bootstrap Anda gagal dengan `npm install` kesalahan. Kesalahan ini dapat terjadi karena Anda menyimpan dependensi AWS CDK aplikasi di registri private node package manager (npm) yang tidak dapat diakses oleh tindakan.

Kemungkinan perbaikan: Gunakan petunjuk berikut untuk memperbarui `cdk.json` file AWS CDK aplikasi Anda dengan registri tambahan dan informasi autentikasi.

Sebelum Anda mulai

1. Buat rahasia untuk informasi otentikasi Anda. Anda akan mereferensikan rahasia ini dalam `cdk.json` file alih-alih memberikan padanan cleartext. Untuk membuat rahasia:
   a. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
   b. Pilih proyek Anda.
   c. Di panel navigasi, pilih CI/CD, lalu pilih Rahasia.
   d. Buat dua rahasia dengan properti berikut:

Rahasia pertama	Rahasia kedua
Nama: `npmUsername`	Nama: `npmAuthToken`
Nilai: *npm-username, di mana npm-username* adalah nama pengguna yang	Nilai: *npm-auth-token* , di mana token akses yang *npm-auth-token* digunakan untuk mengautentikasi ke registri npm

Rahasia pertama	Rahasia kedua
digunakan untuk mengautentikasi ke registri npm pribadi Anda. (Opsional) Deskripsi: `The username used to authenticate to the private npm registry.`	pribadi Anda. Untuk informasi selengkap nya tentang token akses npm, lihat Tentang token akses di dokumentasi npm. (Opsional) Deskripsi: `The access token used to authenticate to the private npm registry.`

Untuk informasi lebih lanjut tentang rahasia, lihatMengkonfigurasi dan menggunakan rahasia dalam alur kerja.

2. Tambahkan rahasia sebagai variabel lingkungan ke AWS CDK tindakan Anda. Tindakan akan menggantikan variabel dengan nilai nyata saat dijalankan. Untuk menambahkan rahasia:

   a. Di panel navigasi, pilih CI/CD, lalu pilih Alur kerja.

   b. Pilih nama alur kerja Anda. Anda dapat memfilter berdasarkan repositori sumber atau nama cabang tempat alur kerja ditentukan, atau memfilter berdasarkan nama alur kerja.

   c. Pilih Edit.

   d. Pilih Visual.

   e. Dalam diagram alur kerja, pilih AWS CDK tindakan Anda.

   f. Pilih tab Input.

   g. Tambahkan dua variabel dengan properti berikut:

Variabel pertama	Variabel kedua
Nama: `NPMUSER` Nilai: `${Secrets.npmUsername}`	Nama: `NPMTOKEN` Nilai: `${Secrets.npmAuthToken}`

Anda sekarang memiliki dua variabel yang berisi referensi ke rahasia.

File definisi alur kerja Anda kode YAMB akan terlihat mirip dengan berikut ini:

> **Note**
>
> Contoh kode berikut berasal dari tindakan AWS CDK bootstrap; tindakan AWS CDK penerapan akan terlihat serupa.

```
Name: CDK_Bootstrap_Action
SchemaVersion: 1.0
Actions:
  CDKBootstrapAction:
    Identifier: aws/cdk-bootstrap@v1
    Inputs:
      Variables:
        - Name: NPMUSER
          Value: ${Secrets.npmUsername}
        - Name: NPMTOKEN
          Value: ${Secrets.npmAuthToken}
      Sources:
        - WorkflowSource
    Environment:
      Name: Dev2
      Connections:
        - Name: account-connection
          Role: codecatalystAdmin
    Configuration:
      Parameters:
        Region: "us-east-2"
```

Anda sekarang siap untuk menggunakan `NPMTOKEN` variabel `NPMUSER` dan dalam `cdk.json` file Anda. Pergi ke prosedur selanjutnya.

Untuk memperbarui file cdk.json Anda

1. Ubah ke direktori root AWS CDK proyek Anda, dan buka `cdk.json` file.
2. Temukan `"app":` properti, dan ubah untuk menyertakan kode yang ditunjukkan dengan *huruf miring merah*:

> **Note**
>
> Contoh kode berikut adalah dari sebuah TypeScript proyek. Jika Anda menggunakan JavaScript proyek, kode akan terlihat serupa meskipun tidak identik.

```
{
  "app": "npm set registry=https://your-registry/folder/CDK-package/ --
userconfig .npmrc && npm set //your-registry/folder/CDK-package/:always-auth=true
 --userconfig .npmrc && npm set //your-registry/folder/CDK-package/:_authToken=
\"${NPMUSER}\":\"${NPMTOKEN}\" && npm install  && npx ts-node --prefer-ts-exts bin/
hello-cdk.ts|js",
  "watch": {
    "include": [
      "**"
    ],
    "exclude": [
      "README.md",
      "cdk*.json",
      "**/*.d.ts",
      "**/*.js",
      "tsconfig.json",
      "package*.json",
...
```

3. Dalam kode yang disorot dengan *huruf miring merah*, ganti:

   - *Registry/folder/CDK-package/ Anda* dengan jalur ke dependensi proyek Anda di registri pribadi Anda. AWS CDK
   - *hello-cdk.ts|.js* dengan nama file entrypoint Anda. Ini mungkin file `.ts` (TypeScript) atau `.js` (JavaScript) tergantung pada bahasa yang Anda gunakan.

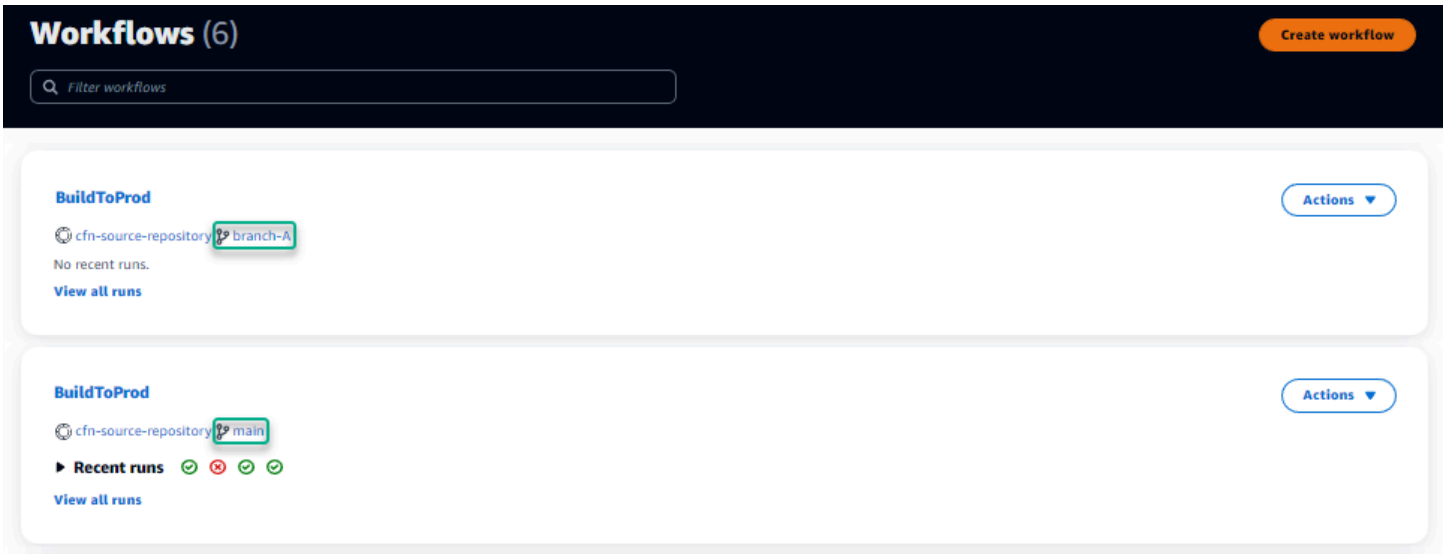
   > **Note**
   >
   > Tindakan akan menggantikan variabel ***NPMUSER dan NPMTOKEN dengan nama pengguna npm dan token*** akses yang Anda tentukan di Rahasia.

4. Simpan `cdk.json` file Anda.

5. Jalankan kembali tindakan secara manual untuk melihat apakah perubahan memperbaiki kesalahan. Untuk informasi selengkapnya tentang menjalankan tindakan secara manual, lihatMemulai alur kerja berjalan secara manual.

Mengapa beberapa alur kerja memiliki nama yang sama?

Alur kerja disimpan per cabang per repositori. Dua alur kerja yang berbeda dapat memiliki nama yang sama jika ada di cabang yang berbeda. Di halaman Alur Kerja, Anda dapat membedakan alur kerja dengan nama yang sama dengan melihat nama cabang. Untuk informasi selengkapnya, lihat Mengatur kode sumber Anda bekerja dengan cabang di Amazon CodeCatalyst.



Dapatkah saya menyimpan file definisi alur kerja saya di folder lain?

Tidak, Anda harus menyimpan semua file definisi alur kerja di `.codecatalyst/workflows` folder. Jika Anda menggunakan repo mono dengan beberapa proyek logis, letakkan semua file definisi alur kerja Anda di `.codecatalyst/workflows` folder, lalu gunakan `FilesChanged` properti di dalam Trigger untuk memicu alur kerja di jalur proyek yang ditentukan. Untuk informasi selengkapnya, lihat Memulai alur kerja berjalan secara otomatis dengan pemicu.

Bagaimana cara menambahkan tindakan secara berurutan ke alur kerja saya?

Secara default, ketika Anda menambahkan tindakan ke alur kerja Anda, itu tidak akan memiliki dependensi dan akan berjalan secara paralel dengan tindakan lain.

Jika Anda ingin mengatur tindakan secara berurutan, Anda dapat mengatur ketergantungan pada tindakan lain dengan mengatur `DependsOn` bidang. Anda juga dapat mengonfigurasi tindakan untuk mengkonsumsi artefak atau variabel yang merupakan output dari tindakan lain. Untuk informasi selengkapnya, lihat Mengkonfigurasi tindakan untuk bergantung pada tindakan lain.

Mengapa alur kerja saya berhasil memvalidasi tetapi gagal saat runtime?

Jika Anda memvalidasi alur kerja Anda menggunakan `Validate` tombol, tetapi alur kerja Anda tetap gagal, itu mungkin karena batasan dalam validator.

Kesalahan apa pun yang mengacu pada CodeCatalyst sumber daya seperti rahasia, lingkungan, atau armada dalam konfigurasi alur kerja tidak akan didaftarkan selama komit. Jika referensi yang tidak valid digunakan, kesalahan hanya akan diidentifikasi ketika alur kerja dijalankan. Demikian pula, jika ada kesalahan dalam konfigurasi tindakan Anda seperti kehilangan bidang wajib atau kesalahan ketik dalam atribut tindakan, mereka akan diidentifikasi hanya ketika alur kerja dijalankan. Untuk informasi selengkapnya, lihat Membuat alur kerja.

Penemuan otomatis tidak menemukan laporan apa pun untuk tindakan saya

Masalah: Saya mengonfigurasi penemuan otomatis untuk tindakan yang menjalankan pengujian, tetapi tidak ada laporan yang ditemukan oleh CodeCatalyst.

Kemungkinan perbaikan: Ini mungkin disebabkan oleh sejumlah masalah. Coba satu atau lebih solusi berikut:

- Pastikan bahwa alat yang digunakan untuk menjalankan tes menghasilkan output dalam salah satu format yang CodeCatalyst mengerti. Misalnya, jika Anda ingin mengizinkan `pytest` CodeCatalyst untuk menemukan laporan pengujian dan cakupan kode, sertakan argumen berikut:

  ```
  --junitxml=test_results.xml --cov-report xml:test_coverage.xml
  ```

  Untuk informasi selengkapnya, lihat Jenis laporan kualitas.
- Pastikan ekstensi file untuk output konsisten dengan format yang dipilih. Misalnya, saat mengonfigurasi `pytest` untuk menghasilkan hasil dalam `JUnitXML` format, periksa apakah ekstensi file tersebut. `.xml` Untuk informasi selengkapnya, lihat Jenis laporan kualitas.
- Pastikan bahwa `IncludePaths` dikonfigurasi untuk menyertakan seluruh sistem file (`**/*`) kecuali Anda sengaja mengecualikan folder tertentu. Demikian pula, pastikan bahwa `ExcludePaths` tidak mengecualikan direktori di mana Anda mengharapkan laporan Anda berada.

- Jika Anda mengonfigurasi laporan secara manual untuk menggunakan file keluaran tertentu, laporan tersebut akan dikecualikan dari penemuan otomatis. Untuk informasi selengkapnya, lihat Contoh laporan kualitas YAMAL.
- Penemuan otomatis mungkin tidak menemukan laporan karena tindakan gagal sebelum keluaran apa pun dihasilkan. Misalnya, build mungkin gagal sebelum pengujian unit apa pun dijalankan.

Tindakan saya gagal pada laporan yang ditemukan secara otomatis setelah saya mengonfigurasi kriteria keberhasilan

Masalah: Saat saya mengaktifkan penemuan otomatis dan mengonfigurasi kriteria keberhasilan, beberapa laporan tidak memenuhi kriteria keberhasilan dan akibatnya tindakan gagal.

Kemungkinan perbaikan: Untuk mengatasi ini, coba satu atau beberapa solusi berikut:

- Ubah `IncludePaths` atau `ExcludePaths` untuk mengecualikan laporan yang tidak Anda minati.
- Perbarui kriteria keberhasilan untuk memungkinkan semua laporan lulus. Misalnya, jika dua laporan ditemukan dengan satu memiliki cakupan garis 50% dan satu lagi 70%, sesuaikan cakupan garis minimum menjadi 50%. Untuk informasi selengkapnya, lihat Kriteria keberhasilan
- Ubah laporan yang gagal menjadi laporan yang dikonfigurasi secara manual. Ini memungkinkan Anda mengonfigurasi kriteria keberhasilan yang berbeda untuk laporan spesifik tersebut. Untuk informasi selengkapnya, lihat Mengkonfigurasi kriteria keberhasilan untuk laporan.

Penemuan otomatis menghasilkan laporan yang tidak saya inginkan

Masalah: Ketika saya mengaktifkan penemuan otomatis, itu menghasilkan laporan yang tidak saya inginkan. Misalnya, CodeCatalyst menghasilkan laporan cakupan kode untuk file yang disertakan dalam dependensi aplikasi saya yang disimpan di. `node_modules`

Kemungkinan perbaikan: Anda dapat menyesuaikan `ExcludePaths` konfigurasi untuk mengecualikan file yang tidak diinginkan. Misalnya, untuk mengecualikan`node_modules`, tambahkan`node_modules/**/*`. Untuk informasi selengkapnya, lihat Sertakan/kecualikan jalur.

Penemuan otomatis menghasilkan banyak laporan kecil untuk satu kerangka pengujian

Masalah: Ketika saya menggunakan kerangka kerja pelaporan pengujian dan cakupan kode tertentu, saya perhatikan bahwa penemuan otomatis menghasilkan sejumlah besar laporan. Misalnya, saat

menggunakan Plugin Maven Surefire, penemuan otomatis menghasilkan laporan yang berbeda untuk setiap kelas pengujian.

Kemungkinan perbaikan: Kerangka kerja Anda mungkin dapat menggabungkan output ke dalam satu file. Misalnya, jika Anda menggunakan Plugin Maven Surefire, Anda dapat menggunakan `npx junit-merge` untuk menggabungkan file secara manual. Ekspresi lengkapnya mungkin terlihat seperti ini:

```
mvn test; cd test-package-path/surefire-reports && npx junit-merge -d ./ && rm
 *Test.xml
```

Alur kerja yang tercantum di bawah CI/CD tidak cocok dengan yang ada di repositori sumber

Masalah: Alur kerja yang ditampilkan pada CI/CD, halaman Alur kerja tidak cocok dengan yang ada di `~/.codecatalyst/workflows/` folder di repositori sumber Anda. Anda mungkin melihat ketidakcocokan berikut:

- Alur kerja muncul di halaman Alur Kerja, tetapi file definisi alur kerja yang sesuai tidak ada di repositori sumber Anda.
- File definisi alur kerja ada di repositori sumber Anda, tetapi alur kerja yang sesuai tidak muncul di halaman Alur Kerja.
- Alur kerja ada di repositori sumber dan halaman Alur Kerja, tetapi keduanya berbeda.

Masalah ini dapat terjadi jika halaman Alur Kerja belum sempat menyegarkan, atau jika kuota alur kerja terlampaui.

Kemungkinan perbaikan:

- Tunggu. Anda biasanya harus menunggu dua atau tiga detik setelah komit ke sumber sebelum Anda melihat perubahan pada halaman Alur Kerja.
- Jika Anda telah melampaui kuota alur kerja, lakukan salah satu hal berikut:

> **Note**
>
> Untuk menentukan apakah kuota alur kerja terlampaui, tinjauKuota untuk alur kerja, dan periksa silang kuota yang didokumentasikan terhadap alur kerja di repositori sumber Anda

atau di halaman Alur Kerja. Tidak ada pesan kesalahan yang menunjukkan bahwa kuota telah terlampaui, jadi Anda harus menyelidikinya sendiri.

- Jika Anda telah melampaui jumlah maksimum alur kerja per kuota ruang, hapus beberapa alur kerja, lalu lakukan komit pengujian terhadap file definisi alur kerja. Contoh komit pengujian mungkin menambahkan spasi ke file.
- Jika Anda telah melampaui kuota ukuran file definisi alur kerja maksimum, ubah file definisi alur kerja untuk mengurangi panjangnya.
- Jika Anda telah melampaui jumlah maksimum file alur kerja yang diproses dalam kuota peristiwa sumber tunggal, lakukan beberapa komit pengujian. Ubah kurang dari jumlah maksimum alur kerja di setiap komit.
- Tingkatkan kuota alur kerja dengan mengaktifkan penagihan tingkatan berbayar. Untuk informasi selengkapnya, lihat Mengelola penagihan di Panduan CodeCatalyst Administrator Amazon.

Saya tidak dapat membuat atau memperbarui alur kerja

Masalah: Saya ingin membuat atau memperbarui alur kerja, tetapi saya melihat kesalahan ketika saya mencoba melakukan perubahan.

Kemungkinan perbaikan: Bergantung pada peran Anda dalam proyek atau ruang, Anda mungkin tidak memiliki izin untuk mendorong kode ke repositori sumber dalam proyek. File YAMB untuk alur kerja disimpan dalam repositori. Untuk informasi selengkapnya, lihat File definisi alur kerja. Peran administrator Space, peran administrator Proyek, dan peran Kontributor semuanya memiliki izin untuk melakukan dan mendorong kode ke repositori dalam proyek.

Jika Anda memiliki peran Kontributor tetapi tidak dapat membuat atau melakukan perubahan pada alur kerja YAMB di cabang tertentu, mungkin ada aturan cabang yang dikonfigurasi untuk cabang tersebut yang mencegah pengguna dengan peran tersebut mendorong kode ke cabang tertentu. Coba buat alur kerja di cabang yang berbeda, atau komit perubahan Anda ke cabang yang berbeda. Untuk informasi selengkapnya, lihat Mengelola tindakan yang diizinkan untuk cabang dengan aturan cabang.

Memecahkan masalah dengan pencarian di CodeCatalyst

Konsultasikan bagian berikut untuk memecahkan masalah yang terkait dengan pencarian di. CodeCatalyst Untuk informasi lebih lanjut tentang alur kerja, lihat Cari kode, masalah, proyek, dan pengguna di CodeCatalyst.

Topik

Saya tidak dapat menemukan pengguna di proyek saya

Masalah: Ketika saya mencoba melihat detail pengguna, saya tidak melihat informasi mereka di proyek.

Kemungkinan perbaikan: Penelusuran saat ini tidak mendukung pencarian pengguna dalam proyek. Untuk mencari pengguna dengan akses ke ruang Anda, beralih ke ruang ini QuickSearch, atau hapus filter proyek apa pun yang mungkin telah Anda tentukan menggunakan bahasa kueri lanjutan.

Saya tidak melihat apa yang saya cari di proyek atau ruang saya

Masalah: Hasil tidak muncul ketika saya mencoba mencari informasi tertentu.

Kemungkinan perbaikan: Pembaruan konten kemungkinan akan memakan waktu beberapa detik untuk diperbarui di hasil penelusuran. Pembaruan besar dapat memakan waktu beberapa menit.

Untuk sumber daya yang belum diperbarui baru-baru ini, Anda mungkin perlu menyempurnakan pencarian Anda. Anda dapat memperbaiki dengan menambahkan lebih banyak kata kunci atau menggunakan bahasa kueri lanjutan. Untuk informasi selengkapnya tentang menyempurnakan kueri Anda, lihat. Menyempurnakan kueri penelusuran Anda

Jumlah hasil pencarian terus berubah saat saya menavigasi halaman

Masalah: Jumlah hasil pencarian tampaknya berubah ketika saya pergi ke halaman berikutnya, jadi tidak jelas berapa banyak total hasil yang ada.

Kemungkinan perbaikan: Saat menavigasi halaman hasil penelusuran, Anda mungkin melihat perubahan jumlah hasil penelusuran yang cocok dengan kueri Anda. Jumlah hasil mungkin diperbarui untuk mencerminkan jumlah kecocokan yang lebih akurat yang ditemukan saat Anda menavigasi halaman.

Saat Anda menavigasi hasil, Anda mungkin melihat pesan berikut: Tidak ada hasil untuk “tes”. Anda akan menerima pesan jika Anda tidak memiliki akses ke hasil yang tersisa.

Kueri penelusuran saya belum selesai

Masalah: Hasil kueri penelusuran saya tidak muncul, dan tampaknya memakan waktu terlalu lama.

Kemungkinan perbaikan: Pencarian Anda mungkin tidak selesai ketika ada banyak pencarian yang dilakukan pada saat yang sama di ruang, baik secara terprogram atau karena aktivitas tim yang tinggi. Jika Anda menjalankan pencarian terprogram, jeda atau kurangi. Jika tidak, coba lagi dalam beberapa detik.

Memecahkan masalah dengan akun yang terkait dengan ruang Anda

Di CodeCatalyst, Anda dapat menambahkan Akun AWS ke ruang Anda untuk memberikan izin ke sumber daya dan untuk tujuan penagihan. Informasi berikut dapat membantu Anda memecahkan masalah umum dengan akun terkait di. CodeCatalyst

Topik

- Permintaan Akun AWS koneksi saya menerima kesalahan token yang tidak valid
- Alur kerja CodeCatalyst proyek Amazon saya gagal dengan kesalahan untuk akun, lingkungan, atau peran IAM yang dikonfigurasi
- Saya memerlukan akun, peran, dan lingkungan terkait untuk membuat proyek
- Saya tidak dapat mengakses halaman Amazon CodeCatalyst Spaces di AWS Management Console
- Saya ingin akun yang berbeda sebagai akun penagihan saya

Permintaan Akun AWS koneksi saya menerima kesalahan token yang tidak valid

Masalah: Saat membuat permintaan koneksi dengan token koneksi, halaman tidak menerima token dan menunjukkan kesalahan yang menyatakan bahwa token tidak valid.

Kemungkinan perbaikan: Pastikan Anda memberikan ID akun yang ingin Anda tambahkan ke ruang Anda. Anda harus memiliki izin administratif untuk Anda Akun AWS atau dapat bekerja dengan administrator Anda untuk menambahkan akun.

Ketika Anda memilih untuk memverifikasi akun, jendela browser baru akan terbuka diAWS Management Console. Akun yang sama harus masuk di sisi konsol. Coba lagi setelah memverifikasi hal berikut:

- Anda masuk ke AWS Management Console dengan yang sama Akun AWS yang ingin Anda tambahkan ke ruang Anda.
- Anda masuk ke AWS Management Console dengan US West (Oregon) Wilayah AWS (`us-west-2`) dipilih.
- Jika Anda telah tiba dari halaman penagihan dan Anda ingin menambahkan Akun AWS sebagai akun penagihan yang ditentukan untuk ruang Anda, pastikan akun tersebut belum menjadi akun penagihan untuk ruang lain.

Alur kerja CodeCatalyst proyek Amazon saya gagal dengan kesalahan untuk akun, lingkungan, atau peran IAM yang dikonfigurasi

Masalah: Saat alur kerja berjalan dan tidak menemukan akun yang dikonfigurasi atau peran IAM yang terkait dengan ruang Anda, Anda harus mengisi bidang peran, koneksi, dan lingkungan secara manual di alur kerja YAMG. Lihat tindakan alur kerja yang gagal, dan perhatikan apakah pesan galatnya adalah sebagai berikut:

- Peran tidak tersedia untuk digunakan dengan koneksi yang terkait dengan lingkungan.
- Tindakan tidak berhasil. Status: GAGAL; Nilai yang diberikan untuk koneksi akun atau lingkungan tidak valid. Verifikasi koneksi terkait dengan ruang Anda dan lingkungan terkait dengan proyek Anda.
- Tindakan tidak berhasil. Status: GAGAL; Nilai yang diberikan untuk peran IAM tidak valid. Verifikasi nama ada, peran IAM ditambahkan ke koneksi akun Anda, dan koneksi sudah dikaitkan dengan ruang Amazon CodeCatalyst Anda

Kemungkinan perbaikan: Pastikan bahwa alur kerja bidang YAMAL memiliki nilai yang akurat untuk Lingkungan, Koneksi, dan Peran. Tindakan CodeCatalyst alur kerja yang memerlukan lingkungan adalah membangun atau menyebarkan tindakan yang menjalankan AWS sumber daya atau yang menghasilkan tumpukan AWS sumber daya.

Pilih blok tindakan alur kerja yang gagal dan kemudian pilih Visual. Pilih tab Konfigurasi. Jika bidang Environment, Connection name, dan Role name tidak diisi, maka Anda perlu memperbarui alur kerja secara manual. Gunakan langkah-langkah berikut untuk mengedit alur kerja YAMG:

- Perluas `/.codecatalyst` direktori, lalu perluas `/workflows` direktori. Buka file Yaml alur kerja. Pastikan bahwa peran IAM dan informasi akun ditentukan dalam YAMAL yang telah Anda konfigurasikan untuk alur kerja Anda. Contoh:

  ```
  Actions:
    cdk_bootstrap:
      Identifier: action-@v1
      Inputs:
        Sources:
          - WorkflowSource
      Environment:
        Name: Staging
        Connections:
          - Name: account-connection
            Role: build-role
  ```

  Properti Lingkungan, Koneksi, dan Peran diperlukan untuk menjalankan tindakan pembuatan dan penerapan CodeCatalyst alur kerja dengan AWS sumber daya. Sebagai contoh, lihat parameter YAMAL referensi tindakan CodeCatalyst build untuk Lingkungan, Koneksi, dan Peran.
- Pastikan ruang Anda memiliki akun yang ditambahkan ke dalamnya, dan pastikan akun tersebut memiliki peran atau peran IAM yang sesuai yang ditambahkan ke akun. Anda dapat menyesuaikan atau menambahkan akun jika Anda memiliki peran administrator Space. Untuk informasi selengkapnya, lihat Memungkinkan akses ke AWS sumber daya yang terhubung Akun AWS.

Saya memerlukan akun, peran, dan lingkungan terkait untuk membuat proyek

Masalah: Dalam opsi pembuatan proyek, proyek saya tidak memiliki akun tambahan yang tersedia di ruang saya, atau saya memerlukan akun lain yang ditambahkan ke ruang saya untuk digunakan proyek saya.

Kemungkinan perbaikan: Untuk ruang Anda, Anda dapat menambahkan otorisasi Akun AWS untuk menambahkannya ke proyek Anda jika Anda memiliki peran administrator Space. Anda juga harus memiliki Akun AWS tempat Anda memiliki izin administratif atau dapat bekerja dengan AWS administrator Anda.

Untuk memastikan akun dan peran akan tersedia di layar pembuatan proyek, Anda harus terlebih dahulu menambahkan akun dan peran. Untuk informasi selengkapnya, lihat Memungkinkan akses ke AWS sumber daya yang terhubung Akun AWS.

Anda memiliki opsi untuk memilih membuat peran layanan dengan kebijakan peran yang disebut kebijakan CodeCatalystWorkflowDevelopmentRole-***spaceName***peran. Peran akan memiliki nama `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* dengan pengenal unik ditambahkan. Untuk informasi selengkapnya tentang kebijakan peran dan peran, lihatMemahami peran CodeCatalystWorkflowDevelopmentRole-***spaceName***layanan. Untuk langkah-langkah untuk membuat peran, lihatMembuat CodeCatalystWorkflowDevelopmentRole-***spaceName***peran untuk akun dan ruang Anda. Peran ditambahkan ke akun Anda dan tersedia di halaman pembuatan proyek di CodeCatalyst.

Saya tidak dapat mengakses halaman Amazon CodeCatalyst Spaces di AWS Management Console

Masalah: Ketika saya mencoba mengakses CodeCatalyst halaman Amazon di AWS Management Console untuk menambahkan akun ke CodeCatalyst ruang saya atau menambahkan peran ke akunAWS, saya menerima kesalahan izin.

Kemungkinan perbaikan:

Untuk ruang Anda, Anda dapat menambahkan otorisasi Akun AWS untuk menambahkannya ke proyek Anda jika Anda memiliki peran administrator Space. Anda juga harus memiliki Akun AWS tempat Anda memiliki izin administratif atau dapat bekerja dengan AWS administrator Anda. Anda harus terlebih dahulu memastikan Anda masuk AWS Management Console dengan akun yang sama

yang ingin Anda kelola. Setelah Anda masuk keAWS Management Console, Anda dapat membuka konsol dan mencoba lagi.

Buka CodeCatalyst halaman Amazon AWS Management Console di https://us-west-2.console.aws.amazon.com/codecatalyst/home?region=us-west-2#/.

Saya ingin akun yang berbeda sebagai akun penagihan saya

Masalah: Ketika saya mengatur CodeCatalyst login saya, saya menyelesaikan beberapa langkah untuk mengatur ruang saya dan mengaitkan yang berwenangAkun AWS. Sekarang, saya ingin mengotorisasi akun lain untuk penagihan.

Kemungkinan perbaikan: Untuk ruang Anda, Anda dapat mengotorisasi akun penagihan jika Anda memiliki peran administrator Space. Anda juga harus memiliki Akun AWS tempat Anda memiliki izin administratif atau dapat bekerja dengan AWS administrator Anda.

Untuk informasi selengkapnya, lihat Mengelola penagihan di Panduan CodeCatalyst Administrator Amazon.

Memecahkan masalah dengan Lingkungan Dev

Konsultasikan bagian berikut untuk memecahkan masalah yang terkait dengan Lingkungan Pengembang. Untuk informasi selengkapnya tentang Lingkungan Pengembang, lihatMenulis dan memodifikasi kode dengan Dev Environments di CodeCatalyst.

Topik

- Pembuatan Lingkungan Pengembang saya tidak berhasil karena masalah dengan kuota
- Saya tidak dapat mendorong perubahan dari Lingkungan Pengembang saya ke cabang tertentu di repositori
- Lingkungan Pengembang saya tidak dilanjutkan
- Lingkungan Dev saya terputus
- Lingkungan Dev saya yang terhubung dengan VPC gagal
- Saya tidak dapat menemukan direktori mana proyek saya berada
- Saya tidak dapat terhubung ke Lingkungan Pengembang saya melalui SSH
- Saya tidak dapat terhubung ke Lingkungan Pengembang saya melalui SSH karena konfigurasi SSH lokal saya hilang

- Saya tidak dapat terhubung ke Lingkungan Pengembang saya melalui SSH karena saya mengalami masalah dengan profil saya AWS Configcodecatalyst
- Memecahkan masalah dengan IDE
- Memecahkan masalah dengan devfiles

Pembuatan Lingkungan Pengembang saya tidak berhasil karena masalah dengan kuota

Masalah: Saya ingin membuat Lingkungan Pengembang di CodeCatalyst, tetapi saya melihat kesalahan. Di konsol, saya melihat pesan di halaman Dev Environments bahwa saya telah mencapai batas penyimpanan untuk ruang tersebut.

Kemungkinan perbaikan: Bergantung pada peran Anda dalam proyek atau ruang, Anda dapat menghapus satu atau beberapa Lingkungan Pengembang Anda sendiri, atau jika Anda memiliki peran administrator Space, Anda dapat menghapus Lingkungan Dev yang tidak digunakan yang dibuat oleh pengguna lain. Anda juga dapat memutuskan untuk mengubah tingkat penagihan ke tingkat yang mencakup lebih banyak penyimpanan.

- Untuk melihat batas penyimpanan, lihat tab Penagihan di CodeCatalyst ruang Amazon untuk melihat apakah kuota Penggunaan telah mencapai batas maksimum yang diizinkan. Jika kuota telah mencapai maksimum, hubungi seseorang dengan peran administrator Space untuk menghapus Lingkungan Dev yang tidak diperlukan atau pertimbangkan untuk mengubah tingkat penagihan.
- Untuk menghapus Lingkungan Dev yang Anda buat yang tidak lagi Anda butuhkan, lihatMenghapus Lingkungan Dev.

Jika masalah berlanjut dan Anda mendapatkan kesalahan di IDE Anda, periksa apakah Anda memiliki CodeCatalyst peran yang memungkinkan Anda membuat Lingkungan Pengembang. Peran administrator Space, peran administrator Proyek, dan peran Kontributor semuanya memiliki izin untuk membuat Lingkungan Pengembang. Untuk informasi selengkapnya, lihat Memberikan akses dengan peran pengguna.

Saya tidak dapat mendorong perubahan dari Lingkungan Pengembang saya ke cabang tertentu di repositori

Masalah: Saya ingin melakukan dan mendorong perubahan kode di Lingkungan Dev saya ke cabang di repositori sumber, tetapi saya melihat kesalahan.

Kemungkinan perbaikan: Bergantung pada peran Anda dalam proyek atau ruang, Anda mungkin tidak memiliki izin untuk mendorong kode ke repositori sumber dalam proyek. Peran administrator Space, peran administrator Proyek, dan peran Kontributor semuanya memiliki izin untuk mendorong kode ke repositori dalam proyek.

Jika Anda memiliki peran Kontributor tetapi tidak dapat mendorong kode ke cabang tertentu, mungkin ada aturan cabang yang dikonfigurasi untuk cabang tertentu yang mencegah pengguna dengan peran tersebut mendorong kode ke cabang tertentu. Coba dorong perubahan Anda ke cabang yang berbeda, atau buat cabang dan kemudian dorong kode Anda ke cabang itu. Untuk informasi selengkapnya, lihat Mengelola tindakan yang diizinkan untuk cabang dengan aturan cabang.

Lingkungan Pengembang saya tidak dilanjutkan

Masalah: Lingkungan Pengembang saya tidak dilanjutkan setelah saya menghentikannya.

Kemungkinan perbaikan: Untuk memperbaiki masalah, lihat tab Penagihan di CodeCatalyst ruang Amazon untuk melihat apakah kuota Penggunaan telah mencapai batas maksimum. Jika kuota telah mencapai batas maksimum, hubungi administrator Space Anda untuk menaikkan tingkat penagihan.

Lingkungan Dev saya terputus

Masalah: Lingkungan Pengembang saya terputus saat saya menggunakannya.

Kemungkinan perbaikan: Untuk memperbaiki masalah, periksa koneksi internet Anda. Jika Anda tidak terhubung ke internet, sambungkan dan lanjutkan bekerja di Lingkungan Pengembang Anda.

Lingkungan Dev saya yang terhubung dengan VPC gagal

Masalah: Saya mengaitkan koneksi VPC ke Lingkungan Pengembang saya dan mengalami kesalahan.

Kemungkinan perbaikan: Docker menggunakan perangkat lapisan tautan yang disebut jaringan jembatan yang memungkinkan kontainer yang terhubung ke jaringan jembatan yang sama untuk

berkomunikasi. Jembatan default biasanya menggunakan subnet `172.17.0.0/16` untuk jaringan kontainer. Jika subnet VPC untuk instance lingkungan Anda menggunakan rentang alamat yang sama dengan yang sudah digunakanDocker, konflik alamat IP mungkin terjadi. Untuk mengatasi konflik alamat IP yang disebabkan oleh Amazon VPC dan Docker menggunakan blok alamat IPv4 CIDR yang sama, konfigurasikan blok CIDR yang berbeda. `172.17.0.0/16`

> **Note**
>
> Anda tidak dapat mengubah rentang alamat IP dari VPC atau subnet yang ada.

Saya tidak dapat menemukan direktori mana proyek saya berada

Masalah: Saya tidak dapat menemukan direktori mana proyek saya berada.

Kemungkinan perbaikan: Untuk menemukan proyek Anda, ubah direktori ke`/projects`. Ini adalah direktori tempat Anda dapat menemukan proyek Anda.

Saya tidak dapat terhubung ke Lingkungan Pengembang saya melalui SSH

Untuk memecahkan masalah koneksi Anda ke Lingkungan Dev Anda melalui SSH, Anda dapat menjalankan `ssh` perintah dengan `-vvv` opsi untuk menampilkan informasi lebih lanjut tentang cara mengatasi masalah Anda:

```
ssh -vvv codecatalyst-dev-env=<space-name>=<project-name>=<dev-environment-id>
```

Saya tidak dapat terhubung ke Lingkungan Pengembang saya melalui SSH karena konfigurasi SSH lokal saya hilang

Jika konfigurasi SSH lokal Anda (`~/.ssh/config`) hilang atau konten `Host codecatalyst-dev-env*` bagian sudah kedaluwarsa, Anda tidak akan dapat terhubung ke Lingkungan Pengembang Anda melalui SSH. Untuk memecahkan masalah ini, hapus `Host codecatalyst-dev-env*` bagian dan jalankan perintah pertama dari modal Akses SSH lagi. Untuk informasi selengkapnya, lihat Menghubungkan ke Lingkungan Dev menggunakan SSH.

Saya tidak dapat terhubung ke Lingkungan Pengembang saya melalui SSH karena saya mengalami masalah dengan profil saya AWS Config`codecatalyst`

Pastikan AWS Config (`~/.aws/config`) untuk `codecatalyst` profil Anda cocok dengan yang dijelaskan diMenyiapkan untuk menggunakan AWS CLI dengan CodeCatalyst. Jika tidak, hapus profil untuk `codecatalyst` dan jalankan perintah pertama dari modal Akses SSH lagi. Untuk informasi selengkapnya, lihat Menghubungkan ke Lingkungan Dev menggunakan SSH.

Memecahkan masalah dengan IDE

Konsultasikan bagian berikut untuk memecahkan masalah yang terkait dengan IDE di. CodeCatalyst Untuk informasi lebih lanjut tentang IDE, lihatMembuat Lingkungan Dev dalam IDE.

Topik

- Saya memiliki versi gambar runtime yang tidak cocok di AWS Cloud9
- Saya tidak dapat mengakses file saya /projects/projects di AWS Cloud9
- Saya tidak dapat meluncurkan Lingkungan Dev saya dalam AWS Cloud9 menggunakan devfile khusus
- Saya mengalami masalah di AWS Cloud9
- Di JetBrains, saya tidak dapat terhubung ke Lingkungan Pengembang saya melalui CodeCatalyst
- Saya tidak dapat menginstal AWS Toolkit untuk IDE saya
- Di IDE saya, saya tidak dapat meluncurkan Lingkungan Dev saya

Saya memiliki versi gambar runtime yang tidak cocok di AWS Cloud9

AWS Cloud9menggunakan versi berbeda dari aset frontend dan gambar runtime backend. Menggunakan versi yang berbeda dapat menyebabkan ekstensi Git dan berfungsi AWS Toolkit dengan tidak benar. Untuk memperbaiki masalah, navigasikan ke dasbor Dev Environment, hentikan Lingkungan Dev Anda, lalu mulai lagi. Untuk memperbaiki masalah menggunakan API, gunakan `UpdateDevEnvironment` API untuk memperbarui runtime. Untuk informasi selengkapnya, lihat UpdateDevEnvironmentdi referensi Amazon CodeCatalyst API.

Saya tidak dapat mengakses file saya **`/projects/projects`** di AWS Cloud9

AWS Cloud9Editor tidak dapat mengakses file di direktori`/projects/projects`. Untuk memperbaiki masalah, gunakan AWS Cloud9 terminal untuk mengakses file Anda atau memindahkannya ke direktori lain.

Saya tidak dapat meluncurkan Lingkungan Dev saya dalam AWS Cloud9 menggunakan devfile khusus

Gambar devfile Anda mungkin tidak kompatibel denganAWS Cloud9. Untuk memperbaiki masalah, tinjau devfile dari repositori Anda dan Lingkungan Dev yang sesuai dan buat yang baru untuk melanjutkan.

Saya mengalami masalah di AWS Cloud9

Untuk masalah lain, periksa bagian pemecahan masalah di AWS Cloud9Panduan Pengguna.

Di JetBrains, saya tidak dapat terhubung ke Lingkungan Pengembang saya melalui CodeCatalyst

Untuk memperbaiki masalah, periksa apakah Anda hanya JetBrains menginstal versi terbaru. Jika Anda memiliki beberapa versi, hapus instalan versi yang lebih lama dan daftarkan penangan protokol Anda lagi dengan menutup IDE dan browser. Kemudian buka JetBrains dan daftarkan pengendali protokol lagi.

Saya tidak dapat menginstal AWS Toolkit untuk IDE saya

Untuk memperbaiki masalah ini untuk VS Code, instal secara manual AWS Toolkit for Visual Studio Code dari GitHub.

Untuk memperbaiki masalah ini JetBrains, instal secara manual AWS Toolkit for JetBrains dari GitHub.

Di IDE saya, saya tidak dapat meluncurkan Lingkungan Dev saya

Untuk memperbaiki masalah ini untuk VS Code, periksa apakah Anda memiliki versi terbaru VS Code dan AWS Toolkit for Visual Studio Code diinstal. Jika Anda tidak memiliki versi terbaru, perbarui dan luncurkan Lingkungan Dev Anda. Untuk informasi selengkapnya, lihat Amazon CodeCatalyst untuk Kode VS.

Untuk memperbaiki masalah ini JetBrains, periksa apakah Anda memiliki versi terbaru JetBrains dan AWS Toolkit for JetBrains diinstal. Jika Anda tidak memiliki versi terbaru, perbarui dan luncurkan Lingkungan Dev Anda. Untuk informasi selengkapnya, lihat Amazon CodeCatalyst untuk JetBrains.

Memecahkan masalah dengan devfiles

Konsultasikan bagian berikut untuk memecahkan masalah yang terkait dengan devfiles di. CodeCatalyst Untuk informasi lebih lanjut tentang devfiles, lihatMengonfigurasi devfile untuk Lingkungan Dev.

Topik

Lingkungan Dev saya menggunakan devfile universal default meskipun saya telah menerapkan gambar khusus dalam devfile khusus

Jika CodeCatalyst mengalami kesalahan saat memulai Lingkungan Dev yang menggunakan devfile kustom, Dev Environment default ke devfile universal default. Untuk memperbaiki masalah, Anda dapat memeriksa kesalahan yang tepat di log di bawah`/aws/mde/logs/devfile.log`. Anda juga dapat memeriksa apakah `postStart` eksekusi berhasil di log Anda:`/aws/mde/logs/devfileCommand.log`.

Proyek saya tidak dibangun di Lingkungan Dev saya dengan devfile universal default

Untuk memperbaiki masalah, periksa apakah Anda tidak menggunakan devfile khusus. Jika Anda tidak menggunakan devfile khusus, lihat `devfile.yaml` file di repositori sumber proyek untuk mencari dan memperbaiki kesalahan apa pun.

Saya ingin memindahkan devfile repositori untuk Lingkungan Dev

Anda dapat memindahkan devfile default `/projects/devfile.yaml` ke repositori kode sumber Anda. Untuk memperbarui lokasi devfile, gunakan perintah berikut:`/aws/mde/mde start --location` *`repository-name`*`/devfile.yaml`.

Saya mengalami masalah saat memulai devfile saya

Jika ada masalah saat memulai devfile Anda, itu akan masuk ke mode pemulihan sehingga Anda masih dapat terhubung ke lingkungan Anda dan memperbaiki devfile Anda. Saat dalam mode pemulihan, berjalan `/aws/mde/mde status` tidak akan berisi lokasi devfile Anda.

```
{
    "status": "STABLE"
}
```

Anda dapat memeriksa kesalahan di log di bawah`/aws/mde/logs`, memperbaiki devfile, dan mencoba menjalankan `/aws/mde/mde start` lagi.

Saya tidak yakin bagaimana cara memeriksa status devfile saya

Anda dapat memeriksa status devfile Anda dengan menjalankan`/aws/mde/mde status`. Setelah menjalankan perintah ini, Anda mungkin melihat salah satu dari berikut ini:

- `{"status": "STABLE", "location": "devfile.yaml" }`

  Ini menunjukkan bahwa devfile Anda benar.
- `{"status": "STABLE" }`

  Ini menunjukkan bahwa devfile Anda tidak dapat memulai dan telah memasuki mode pemulihan.

Anda dapat memeriksa kesalahan yang tepat di log di bawah`/aws/mde/logs/devfile.log`.

Anda juga dapat memeriksa apakah `postStart` eksekusi berhasil di log Anda:`/aws/mde/logs/devfileCommand.log`.

Untuk informasi selengkapnya, lihat Menentukan gambar devfile universal untuk Lingkungan Dev.

Devfile saya tidak kompatibel dengan perkakas yang disediakan pada gambar terbaru

Di Lingkungan Pengembang Anda, `devfile` atau `devfile postStart` mungkin gagal jika `latest` perkakas tidak memiliki perkakas yang diperlukan untuk proyek tertentu. Untuk memperbaiki masalah, lakukan hal berikut:

1. Arahkan ke devfile Anda.
2. Di devfile Anda, perbarui ke versi gambar granular alih-alih. `latest` Ini mungkin terlihat mirip dengan yang berikut:

```
components:
  - container:
      image: public.ecr.aws/amazonlinux/universal-image:1.0
```

3. Buat Lingkungan Dev baru menggunakan devfile yang diperbarui.

Memecahkan masalah dengan masalah

Informasi berikut dapat membantu Anda memecahkan masalah umum dengan masalah di. CodeCatalyst

Topik

- Saya tidak dapat memilih penerima tugas untuk masalah saya

Saya tidak dapat memilih penerima tugas untuk masalah saya

Masalah: Saat membuat masalah, daftar penerima tugas kosong.

Kemungkinan perbaikan: Daftar penerima tugas secara langsung ditautkan ke CodeCatalyst pengguna yang terdaftar sebagai anggota untuk proyek. Untuk memverifikasi bahwa akses profil pengguna berfungsi dengan baik, pilih ikon profil dan kemudian pilih Profil pengguna. Jika informasi profil pengguna tidak terisi, periksa laporan kesehatan untuk setiap insiden. Jika memang terisi, ajukan tiket layanan.

Memecahkan masalah antara Amazon CodeCatalyst dan AWS SDK atau AWS CLI

Informasi berikut dapat membantu Anda memecahkan masalah umum saat bekerja dengan CodeCatalyst dan AWS CLI atau SDK. AWS

Topik

Saya menerima kesalahan ketika saya masuk aws codecatalyst di baris perintah atau terminal yang mengatakan itu adalah pilihan yang tidak valid

Masalah: Ketika saya mencoba menggunakan AWS CLI with CodeCatalyst, satu atau lebih aws codecatalyst perintah tidak diakui sebagai valid.

Solusi: Penyebab paling umum untuk masalah ini adalah Anda menggunakan versi AWS CLI yang tidak berisi pembaruan terbaru untuk layanan dan perintah terbaru. Perbarui instalasi Anda AWS CLI dan kemudian coba lagi. Untuk informasi selengkapnya, lihat Menyiapkan untuk menggunakan AWS CLI dengan CodeCatalyst.

Saya menerima kesalahan kredensial ketika saya menjalankan perintah aws codecatalyst

Masalah: Ketika saya mencoba menggunakan AWS CLI with CodeCatalyst, saya menerima pesan yang menyatakan `You can configure credentials by running "aws configure".` atau`Unable to locate authorization token`.

Solusi: Anda harus mengkonfigurasi AWS CLI profil untuk bekerja dengan CodeCatalyst perintah. Untuk informasi selengkapnya, lihat Menyiapkan untuk menggunakan AWS CLI dengan CodeCatalyst.

Memahami status layanan saat ini dengan laporan CodeCatalyst kesehatan

Laporan CodeCatalyst kesehatan Amazon adalah dasbor publik yang menyediakan pengguna dengan daftar agregat up-to-the-minute pemberitahuan mengenai kinerja sumber daya dan ketersediaan layanan CodeCatalyst yang memiliki dampak luas. Anda dapat melihat sumber daya mana yang mengalami masalah dan dapat memengaruhi aplikasi CodeCatalyst. Ini memungkinkan Anda untuk melacak pemadaman dan sistem downtime sumber daya lainnya secara luas. Ketika insiden terjadi, indikator biru muncul di ikon laporan kesehatan. Selain itu, CodeCatalyst secara otomatis mengirimkan peringatan dan pemberitahuan email ke semua pengguna dengan peran administrator Space dalam proyek, memberikan rincian dan riwayat insiden dalam waktu dekat.

Dasbor menyediakan daftar semua peristiwa aktif dan catatan hingga 100 insiden sebelumnya yang terjadi dalam 30 hari terakhir. Anda dapat mengatur daftar insiden berdasarkan tanggal insiden diperbarui. Anda juga dapat menyegarkan daftar insiden hingga pembaruan menit.

Berikut adalah alur kerja yang mungkin untuk menggunakan laporan CodeCatalyst kesehatan:

Mateo Jackson adalah pengembang di Budding Space dengan izin administrator Space. Saat mencoba membuat permintaan tarik, dia terus mendapatkan pesan kesalahan. Dia memeriksa emailnya dan menemukan bahwa dia menerima email insiden sistem yang dibuat secara otomatis dari CodeCatalyst memberikan riwayat terperinci tentang masalah sistem yang memengaruhi ruangnya. Dia memilih Lihat pembaruan dan dibawa ke laporan CodeCatalyst kesehatan di mana dia dapat melihat semua insiden yang dilaporkan sistem. Dia memilih insiden dari daftar untuk mengetahui informasi lebih lanjut. Layar terpisah terbuka yang memberikan stempel waktu pembaruan terakhir, riwayat, kemampuan yang terkena dampak, waktu mulai, dan status insiden saat ini. Dia juga dapat melihat bahwa masalah ini sedang berlangsung, tetapi tim layanan telah mulai mengerjakannya. Setiap kali ada pembaruan pada riwayat atau status insiden, ia menerima email. Jika dia tidak memiliki akses ke emailnya, dia dapat memilih ikon lonceng di panel atas untuk sampai ke laporan CodeCatalyst kesehatan.

CodeCatalyst konsep laporan kesehatan

Mempelajari konsep-konsep berikut akan membantu Anda memahami laporan CodeCatalyst kesehatan dan bagaimana mereka memungkinkan Anda melacak kesehatan aplikasi, layanan, dan sumber daya Anda.

Insiden

Insiden adalah peristiwa sistem yang mempengaruhi aplikasi dan sumber daya di dalamnya CodeCatalyst. Anda dapat memilih insiden untuk melihat riwayat acara yang terperinci, termasuk waktu dimulai dan jika tim layanan sedang berupaya menyelesaikannya.

Status

Status adalah status real-time dari insiden tersebut. Ini akan ditampilkan sebagai Sedang Berlangsung atau Terselesaikan.

Kemampuan yang terkena dampak

Kemampuan yang terkena dampak adalah sumber daya atau aplikasi yang terpengaruh oleh insiden tersebut. Insiden tunggal dapat memengaruhi beberapa area dalam sistem, termasuk permintaan tarik, masalah, alur kerja, pengujian, penerapan, dan sumber.

Diperbarui pada

Diperbarui pada memberikan stempel waktu pembaruan terakhir untuk insiden tersebut.

AWS Supportuntuk Amazon CodeCatalyst

Saat Anda membuat spasi, Anda harus menghubungkan Akun AWS dan menetapkannya sebagai akun penagihan untuk ruang Anda. Yang Akun AWS Anda tetapkan sebagai akun penagihan Anda juga merupakan tempat Anda mengakses AWS Support paket Anda untuk Amazon. CodeCatalyst Jika Anda memerlukan dukungan, Anda dapat membuat kasus dukungan dari yang ditunjuk iniAkun AWS.

CodeCatalyst pengguna di ruang menggunakan CodeCatalyst halaman AWS Support untuk Amazon CodeCatalyst untuk mengelola kasus dukungan. Anda dapat meningkatkan ke AWS Support paket seperti Business Support atau Enterprise Support untuk membuat dan mengelola kasus dukungan CodeCatalyst teknis CodeCatalyst. Support tersedia melalui telepon, web, atau obrolan untuk kasus dukungan.

Hanya kasus khusus untuk CodeCatalyst layanan dan sumber daya yang dapat didukung melalui AWS Support Amazon CodeCatalyst. CodeCatalyst sumber daya termasuk sumber daya yang digunakan di dalam CodeCatalyst dan oleh pengguna di CodeCatalyst, tetapi ini tidak termasuk sumber daya yang digunakan untuk layanan lain AWS atau pihak ketiga. Jika Anda memerlukan dukungan untuk AWS layanan lain, Anda harus membukanya melaluiAWS Management Console.

Untuk mengubah rencana dukungan, lihat Mengubah rencana dukungan.

> **Note**
>
> Paket Dukungan Pengembang tidak dirancang untuk lingkungan produksi. Jika akun penagihan spasi memiliki paket Dukungan Pengembang, paket ini tidak akan mengalir ke semua administrator ruang dan anggota ruang di dalamnya. AWS Support CodeCatalyst

Penagihan AWS Support untuk Amazon CodeCatalyst

Saat Anda membuat spasi CodeCatalyst, pengguna di ruang tersebut dapat membuat dan mengelola kasus AWS Support dukungan dari Amazon CodeCatalyst. Anda dapat membuat dua jenis kasus pelanggan:

- Kasus dukungan akun dan penagihan tersedia untuk semua CodeCatalyst pengguna di ruang tersebut. Anda bisa mendapatkan bantuan terkait penagihan dan pertanyaan akun berdasarkan izin Anda masuk. CodeCatalyst
- Kasus dukungan teknis menghubungkan Anda dengan teknisi dukungan teknis untuk bantuan terkait masalah teknis dan ekstensi layanan ke aplikasi pihak ketiga. Jika Anda memiliki Basic Support, Anda tidak dapat membuat kasus dukungan teknis.

  Yang Akun AWS ditunjuk sebagai akun penagihan untuk ruang tersebut harus memiliki paket Business Support atau Enterprise Support untuk ruang yang akan digunakan AWS Support CodeCatalyst untuk kasus teknis.

  > **Note**
  >
  > Jika ruang Anda digunakan AWS Support untuk Amazon CodeCatalyst dari akun yang tidak memiliki paket Dukungan Bisnis atau Dukungan Perusahaan, Anda masih dapat menggunakan Amazon AWS Support CodeCatalyst untuk kasus akun dan penagihan.

Untuk dukungan teknis, Anda harus membuka semua kasing melalui CodeCatalyst konsol. Anda tidak dapat membuat kasus dukungan teknis untuk CodeCatalyst dari AWS SupportdalamAWS Management Console.

> **Note**
>
> Permintaan peningkatan batas layanan tidak tersedia AWS Support untuk Amazon CodeCatalyst. Permintaan ini hanya dapat dikirimkan oleh pengguna root untuk akun penagihan ruang di. AWS Support Center Console

AWS Supportuntuk Amazon CodeCatalyst memiliki perjanjian dukungan yang sama denganAWS Support, dengan pertimbangan berikut:

- Daftar keparahan, waktu respons, dan SLA yang AWS Support berlaku untuk kasus dukungan di AWS Support for CodeCatalyst, sebagaimana dirinci dalam Memilih tingkat keparahan.
- Administrator ruang dan anggota ruang tidak dapat menggunakan AWS Support API atau AWS SDK atau AWS Support aplikasi di Slack untuk membuat kasus. CodeCatalyst CodeCatalyst kasus dukungan hanya dapat diajukan dari CodeCatalyst.

> **Note**
>
> CodeCatalyst tidak sepenuhnya terintegrasi dengan AWS Trusted Advisor atau Deteksi AWS Insiden dan Respons. Validasi bagaimana CodeCatalyst terintegrasi untuk memastikan praktik bisnis Anda selaras dengan integrasi saat ini.

Anda harus menjadi pengguna di ruang di mana Anda ingin meminta dukungan.

> **Note**
>
> Jika Anda memiliki lebih dari satu builder di ruang Anda, kami sarankan Anda membeli paket Business Support atau Enterprise Support. Rencana ini memberikan dukungan teknis untuk ruang hingga 5.000 pembangun.

Yang Akun AWS ditunjuk sebagai akun penagihan untuk ruang menggunakan `AWSRoleForCodeCatalystSupport` peran dan kebijakan AmazonCodeCatalystSupportAccessterkelola. Ini memungkinkan CodeCatalyst pengguna di ruang untuk mengakses CodeCatalyst halaman AWS Support untuk Amazon. Untuk informasi selengkapnya tentang peran dan kebijakan ini, lihat AmazonCodeCatalystSupportAccess. Untuk pertimbangan lain tentang penagihan, lihat Mengelola penagihan di Panduan Administrator Amazon CodeCatalyst .

Berikut adalah aliran yang mungkin untuk pembangun yang membuat kasus dukungan di CodeCatalyst:

Mateo Jackson adalah pengembang proyek di CodeCatalyst. Setelah mendaftar Akun AWS yang mengelola penagihan AWS Support untuk Amazon CodeCatalyst dan meningkatkan ke paket Dukungan Bisnis, semua pembangun di ruang tersebut dapat membuat kasus dukungan teknis. Mateo mengajukan kasus dukungan teknis untuk alur kerja yang gagal dalam proyek mereka. Mateo menggunakan CodeCatalyst halaman AWS Support untuk Amazon untuk mengisi formulir dan membuat kasus, memberikan ID alur kerja dan detail lainnya dalam permintaan. Kasus dibuat dengan ID kasus dan menyertakan ID akun yang Akun AWS ditunjuk sebagai akun penagihan dan terkait dengan paket dukungan untuk ruang tersebut.

Meskipun semua pembangun dapat membuat kasus dukungan di AWS Support for CodeCatalyst, Anda tidak dikenakan biaya untuk setiap kasus yang dibuat. Anda dapat membuka kasus dan

kontak yang hampir tidak terbatas berdasarkan paket AWS Support Premium yang Anda beli di akun penagihan ruang Anda.

> **Note**
>
> Akun penagihan ruang adalah Akun AWS yang dikenakan biaya untuk CodeCatalyst pengguna dan sumber daya. Jika Anda telah dikerahkan ke tambahanAkun AWS, hubungi AWS Support melalui AWS Management Console untuk bantuan dengan sumber daya yang dikerahkan ke layanan lain.
> Anda dapat mengidentifikasi yang Akun AWS Anda gunakan dari alur kerja.

Menyiapkan ruang Anda AWS Support untuk Amazon CodeCatalyst

AWS Supportuntuk Amazon CodeCatalyst mengelola kasus dukungan sebagai bagian dari integrasi AWS Support API dengan CodeCatalyst.

Peran tersebut adalah `AWSRoleForCodeCatalystSupport` peran layanan yang digunakan untuk kasus dukungan di ruang Anda. Peran harus ditambahkan ke akun penagihan yang ditunjuk untuk ruang tersebut. Untuk informasi selengkapnya atau untuk membuat peran, lihatMembuat AWSRoleForCodeCatalystSupportperan untuk akun dan ruang Anda.

> **Note**
>
> Untuk ruang yang dibuat sebelum 20 April 2023, Anda harus membuat peran agar dukungan untuk bekerja CodeCatalyst untuk ruang Anda. Jika membuat spasi setelah 20 April 2023, Anda dapat membuat peran selama pembuatan ruang, di halaman Detail penagihan di CodeCatalyst, atau dengan mengklik tautan spanduk dukungan di. CodeCatalyst

Untuk mengatur dukungan untuk ruang Anda

1. Saat Anda membuat CodeCatalyst spasi, Anda diinstruksikan untuk menghubungkan akun penagihan. Akun penagihan yang ditunjuk untuk ruang tersebut akan ditagih oleh. AWS Untuk informasi selengkapnya tentang membuat ruang, lihatMembuat peran ruang dan pengembangan pertama Anda (dimulai tanpa undangan).

2. Saat Anda membuat CodeCatalyst spasi, opsi tersedia untuk membuat peran `AWSRoleForCodeCatalystSupport` layanan yang memungkinkan CodeCatalyst pengguna mengakses dukungan. Peran tersebut menggunakan kebijakan terkelola`AmazonCodeCatalystSupportAccess`. Peran harus ditambahkan ke yang Akun AWS ditunjuk sebagai akun penagihan untuk ruang tersebut. Untuk informasi lebih lanjut tentang pembuatan peran, lihat Membuat AWSRoleForCodeCatalystSupportperan untuk akun dan ruang Anda.
3. Untuk akun penagihan yang ditunjuk untuk ruang tersebut, administrator ruang disarankan untuk membeli paket Business Support atau Enterprise Support untuk akun tersebutAkun AWS. Semua anggota di ruang akan dapat mengelola kasus AWS Support dukungan dari Amazon CodeCatalyst, dan saluran dukungan akan diselaraskan dengan AWS Support paket yang telah Anda beli di mana integrasi selesai.
4. Untuk membuat dan mengelola kasus dukungan di CodeCatalyst, lihatMembuat kasus CodeCatalyst dukungan di CodeCatalyst.

Mengakses dukungan untuk CodeCatalyst di AWS Management Console

Jika akun penagihan yang diaktifkan dukungan untuk ruang terputus, AWS Support kasus yang terkait dengan akun penagihan ruang sebelumnya dan paket dukungan terkait tidak akan lagi terlihat di Amazon. AWS Support CodeCatalyst Pengguna root untuk akun penagihan tersebut dapat melihat dan menyelesaikan kasus lama dari AWS Management Console dan dapat mengatur izin IAM agar pengguna lain dapat melihat dan menyelesaikan kasus lama. AWS Support Anda masih dapat mengambil bagian dalam manfaat dari rencana dukungan Anda dari AWS Management Console untuk semua yang lain Layanan AWS dan menyelesaikan setiap kasus CodeCatalyst dukungan yang sebelumnya tidak diselesaikan.

Untuk informasi selengkapnya, lihat Memperbarui, menyelesaikan, dan membuka kembali kasus Anda di AWS SupportPanduan Pengguna.

Kasus dukungan untuk informasi how-to umum juga CodeCatalyst dapat dibuka diAWS Management Console, tetapi tidak ada dukungan teknis yang dapat diterima melalui saluran ini untuk. CodeCatalyst Untuk informasi selengkapnya, lihat Membuat kasus dukungan dan manajemen kasus di Panduan AWS Support Pengguna.

Berikut adalah aliran yang mungkin bagi pengguna yang menyelesaikan kasus dukungan untuk CodeCatalyst diAWS Management Console:

Meskipun semua pembangun dapat membuat kasus dukungan AWS Support untuk Amazon CodeCatalyst, permintaan dukungan ditagih dari akun yang ditetapkan sebagai akun penagihan untuk ruang tersebut. Mateo Jackson adalah pengembang proyek CodeCatalyst yang membuka kasus dukungan teknis untuk alur kerja yang gagal dalam proyek mereka. Namun, akun penagihan untuk ruang yang didaftarkan AWS Support untuk Amazon CodeCatalyst dan telah membeli paket Dukungan Bisnis telah terputus dari ruang tersebut. Satu-satunya cara bagi Mateo untuk melihat komunikasi terbaru dan menyelesaikan kasus yang dibuka CodeCatalyst adalah dengan mengelola ID kasus dari AWS Support Pusat diAWS Management Console. Untuk melakukan ini, Mateo diberikan izin IAM dari pengguna root dari akun penagihan ruang sebelumnya yang dilampirkan ke kasus dukungan mereka dan menyelesaikan kasus melalui di konsol. AWS Support

> **Important**
>
> Jika Anda mengubah akun penagihan yang ditunjuk untuk ruang Anda, AWS Support paket Anda akan tetap dapat diakses hingga akhir bulan melalui AWS Management Console satu-satunya. Anda harus membeli kembali AWS Support pada akun penagihan yang diperbarui untuk terus mengakses kasus dukungan yang Anda buat sebelumnya di. CodeCatalyst Sebaiknya tunggu sampai Anda menyelesaikan semua kasus dukungan untuk mengubah akun penagihan ruang untuk menghindari dampak apa pun dalam mengakses kasus dukungan Anda melalui AWS Support Amazon. CodeCatalyst

Membuat kasus CodeCatalyst dukungan di CodeCatalyst

Anda dapat membuat kasus dukungan di CodeCatalyst halaman AWS Support untuk Amazon.

AWSBuilder ID hanya bisa mendapatkan dukungan untuk alias yang diautentikasi dan hanya untuk sumber daya berdasarkan izinnya. Opsi akun dan penagihan tersedia untuk semua administrator ruang dan anggota ruang. Namun, pengguna hanya bisa mendapatkan dukungan untuk sumber daya yang mereka miliki akses ke dalam CodeCatalyst dan tidak dalam kaitannya dengan mengelola penagihan untuk akun.

Anda dapat membuat Kasus Akun dan Penagihan atau kasus dukungan teknis untuk CodeCatalyst sumber daya Anda menggunakan CodeCatalyst halaman AWS Support for untuk ruang Anda.

> **Note**
>
> Hanya kasus khusus untuk CodeCatalyst layanan dan sumber daya yang dapat didukung melalui AWS Support Amazon CodeCatalyst. CodeCatalyst sumber daya termasuk sumber daya yang digunakan di dalam CodeCatalyst dan oleh pengguna di CodeCatalyst, tetapi ini tidak termasuk sumber daya yang digunakan untuk layanan lain AWS atau pihak ketiga. Jika Anda memerlukan dukungan untuk AWS layanan lain, Anda harus membukanya melaluiAWS Management Console.

Untuk membuat kasus dukungan di CodeCatalyst

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.

   > **Tip**
   >
   > Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Di bagian atas halaman, pilih ? ikon, dan kemudian pilih Support.
4. Pilih Buat kasus.
5. Pilih salah satu opsi berikut:

   - Akun dan penagihan
   - Teknis

   > **Note**
   >
   > Di AWS Support Amazon CodeCatalyst, jika paket Business Support atau Enterprise Support ditambahkan ke akun penagihan ruang, dukungan kasus CodeCatalyst teknis akan tersedia untuk semua administrator ruang dan anggota ruang. Untuk informasi pemecahan masalah, lihat Saya tidak dapat membuat kasus dukungan teknis untuk ruang saya.
   >
   > AWS Supportrencana tidak menjangkau ruang. Jika Anda adalah anggota dari beberapa ruang, administrator ruang Anda perlu membeli paket AWS Support Premium untuk setiap ruang untuk menerima dukungan teknis di semua ruang.

6. Pilih Layanan, Kategori, dan Tingkat Keparahan. Untuk informasi tentang memilih tingkat keparahan, lihat Memilih tingkat keparahan.

   - Bimbingan umum
   - Sistem terganggu
   - Sistem produksi terganggu
   - Sistem produksi turun
   - Sistem kritis bisnis turun

7. Pilih Langkah selanjutnya: Informasi tambahan.
8. Pada halaman Informasi tambahan, untuk Subjek, masukkan judul tentang masalah Anda.
9. Untuk Deskripsi, ikuti petunjuk untuk menjelaskan kasus Anda, seperti berikut ini:

   - Memecahkan masalah informasi yang khusus untuk CodeCatalyst, seperti ID alur kerja, log, atau tangkapan layar
   - Pesan eror yang Anda terima
   - Langkah pemecahan masalah yang Anda ikuti

   > **Note**
   >
   > Jangan membagikan informasi sensitif apa pun dalam kasus korespondensi, seperti kredensi, kartu kredit, URL yang ditandatangani, atau informasi identitas pribadi.

10. (Opsional) Pilih Lampirkan file untuk menambahkan file yang relevan ke kasus Anda, seperti log kesalahan atau tangkapan layar. Anda dapat melampirkan hingga tiga file. Setiap file dapat mencapai 5 MB.
11. Dalam nama Space, nama ruang Anda ditampilkan.
12. Dalam nama Builder, nama lengkap yang terkait dengan AWS Builder ID Anda terisi otomatis.
13. (Opsional) Pilih proyek dalam nama Proyek (jika ada).

    > **Note**
    >
    > Anda hanya akan diperlihatkan proyek yang memiliki izin untuk Anda. Jika Anda memerlukan akses ke proyek lain, minta administrator proyek Anda untuk memberi Anda akses sebelum membuat kasus dukungan.

14. Pilih Langkah selanjutnya: Hubungi kami.
15. Dalam Bahasa kontak pilihan, pilih default. Hanya bahasa Inggris yang tersedia saat ini.
16. Pilih opsi Web, Telepon, atau Obrolan untuk metode kontak.
17. Tinjau detail kasus Anda, lalu pilih Kirim. Nomor ID kasus dan ringkasan muncul.

    Kasus dukungan dibuat pada tingkat ruang dan dapat dilihat oleh semua anggota dengan akses ke ruang dan proyek (jika dipilih) yang ditentukan dalam kasus dukungan Anda. Tidak ada cara untuk menghilangkan kasus dukungan dari pengguna individu saat ini.

Menyelesaikan kasus dukungan di CodeCatalyst

Anda dapat menyelesaikan kasus dukungan terbuka dari CodeCatalyst halaman AWS Support untuk Amazon.

Anda harus memiliki peran administrator Space atau anggota Space di ruang tempat Anda ingin menyelesaikan kasus dukungan. Jika Anda tidak memiliki peran administrator Space, atau jika proyek dipilih saat kasus dibuat, Anda juga harus memiliki keanggotaan proyek untuk melihat dan menyelesaikan kasus tersebut.

Untuk menyelesaikan kasus dukungan terbuka di CodeCatalyst

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.

   > **Tip**
   >
   > Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Di bagian atas halaman, pilih? ikon, dan kemudian pilih AWS Supportuntuk Amazon CodeCatalyst.
4. Pilih tautan untuk kasus dukungan yang ingin Anda kelola. Pilih Resolve case (Selesaikan kasus).

Membuka kembali kasus dukungan di CodeCatalyst

Anda dapat menggunakan membuka kembali kasus dukungan yang diselesaikan dari CodeCatalyst halaman AWS Support untuk Amazon.

> **Note**
>
> Anda dapat membuka kembali kasus dukungan hingga 14 hari sejak masalah teratasi. Namun, Anda tidak dapat membuka kembali kasus yang tidak aktif selama lebih dari 14 hari. Jika Anda tidak dapat membuka kembali kasus Anda, buka kasus baru dan sertakan ID kasus sebelumnya sebagai referensi.
> Jika Anda membuka kembali kasus yang ada yang memiliki informasi berbeda dari masalah Anda saat ini, agen dukungan mungkin meminta Anda untuk membuat kasus baru.

Untuk membuka kembali kasus dukungan di CodeCatalyst

1. Buka CodeCatalyst konsol di https://codecatalyst.aws/.
2. Arahkan ke CodeCatalyst ruang Anda.

   > **Tip**
   >
   > Jika Anda memiliki lebih dari satu spasi, pilih spasi di bilah navigasi atas.

3. Di bagian atas halaman, pilih? ikon, lalu pilih AWS Support for CodeCatalyst.
4. Pilih tautan untuk kasus dukungan yang ingin Anda kelola. Pilih Buka Kembali. Pilih OK di layar konfirmasi, lalu pilih Kirim.
5. Isi Deskripsi dengan informasi terbaru tentang masalah yang sama. Jangan membagikan informasi sensitif apa pun dalam kasus korespondensi, seperti kredensi, kartu kredit, URL yang ditandatangani, atau informasi identitas pribadi.

Kuota untuk CodeCatalyst

Tabel berikut menjelaskan kuota dan batas untuk Amazon CodeCatalyst. Anda dapat menemukan informasi tambahan untuk aspek-aspek spesifik CodeCatalyst dalam topik berikut:

- Kuota untuk repositori sumber di CodeCatalyst
- Kuota untuk identitas, izin, dan akses di CodeCatalyst
- Kuota untuk alur kerja
- Kuota untuk Lingkungan Pengembang di CodeCatalyst
- Kuota untuk proyek
- Kuota untuk cetak biru di CodeCatalyst
- Kuota untuk spasi
- Kuota untuk masalah di CodeCatalyst

Jumlah maksimum spasi dalam akun	5
Jumlah maksimum spasi yang dapat dibuat pengguna dalam satu bulan kalender	5
Jumlah minimum Akun AWS untuk sebuah ruang	1
Jumlah maksimum koneksi akun untuk suatu ruang	5.000
Jumlah maksimum Akun AWS sebagai akun penagihan untuk spasi	1
Jumlah maksimum koneksi VPC untuk suatu ruang	100
Jumlah maksimum proyek dalam suatu ruang	100
Jumlah maksimum proyek yang dapat dimiliki pengguna	1.000

Deskripsi ruang	Deskripsi ruang bersifat opsional. Jika ditentukan, panjangnya harus antara 0 dan 200 karakter. Mereka dapat berisi kombinasi huruf, angka, spasi, titik, garis bawah, koma, tanda hubung, dan karakter khusus berikut: `? & $ % + = / \ ; : \n \t \r`
Nama ruang	Nama ruang harus unik di seberang CodeCatalyst. Anda tidak dapat menggunakan kembali nama spasi yang dihapus. Nama spasi harus antara 3 dan 63 karakter panjangnya. Mereka juga harus mulai dengan karakter alfanumerik. Nama spasi dapat berisi kombinasi huruf, angka, titik, garis bawah, dan tanda hubung. Mereka tidak dapat berisi salah satu karakter berikut: `! ? @ # $ % ^ & * ( ) + = { } [ ] | /\ > < ~ ` ' " ; :`

Riwayat dokumen untuk Amazon CodeCatalyst

Tabel berikut menjelaskan riwayat dokumentasi dan pembaruan untuk keseluruhan dokumentasi CodeCatalyst.

Perubahan	Deskripsi	Tanggal
Konten baru: Menambahkan langkah-langkah untuk bekerja dengan koneksi pribadi	Menambahkan langkah-langkah untuk membuat dan menghapus koneksi pribadi. Koneksi pribadi memungkinkan Anda mengelola GitHub sumber daya untuk proyek dan cetak biru di Amazon. CodeCatalyst	Juni 6, 2024
Konten yang diperbarui: Gunakan repositori pihak ketiga dengan cetak biru	Dokumentasi yang diperbarui untuk kemampuan membuat repositori GitHub atau Bitbucket saatMembuat proyek dengan cetak biru,Menerapkan cetak biru dalam proyek untuk menambahkan sumber daya, atau membuat cetak biru kustom.	Juni 5, 2024
Konten baru: Ekstensi repositori Bitbucket	Menambahkan konten baru untuk menggunakan ekstensi repositori Bitbucket di. CodeCatalyst	Juni 5, 2024
Konten baru: Jenis tindakan	Memperbarui Tindakan pihak ketiga topik untuk menyebutkan tindakan SonarCloud Pindai.	29 Mei 2024

Konten yang diperbarui: Tindakan “AWS CDK menyebarkan” definisi YAMAL	Memperbaiki `CdkRootRootPath` contoh.	28 Mei 2024
Konten yang diperbarui	Judul topik yang diperbarui dan konten yang direorganisasi untuk meningkatkan keterbacaan dan penemuan. Jika Anda ingin memberikan umpan balik tentang perubahan ini, gunakan tautan Berikan umpan balik ini.	17 Mei 2024
Konten baru: Melihat riwayat perubahan pada file	Memperbarui dokumentasi untuk mencerminkan fungsionalitas baru untuk melihat riwayat perubahan pada file dalam repositori sumber.	1 Mei 2024
Konten yang diperbarui: Tutorial: Menggunakan fitur AI generatif	Memperbarui tutorial untuk mencerminkan integrasi dengan Amazon Q Developer.	April 29, 2024
Konten yang diperbarui: Tutorial: Menggunakan fitur AI generatif	Memperbarui tutorial untuk mencerminkan memungkinkan Amazon Q menganalisis masalah untuk kompleksitas, menyarankan dan membuat tugas, dan bekerja dalam tugas dalam suatu masalah.	April 22, 2024

Konten baru: Mengembangkan alur kerja	DitambahkanMengembangkan alur kerja,Memerlukan persetujuan pada alur kerja berjalan, dan beberapa topik lain yang terkait dengan persetujuan alur kerja.	April 22, 2024
Konten baru: Tutorial: Membuat aplikasi full-stack dengan cetak biru PDK yang dapat dikomposisi	Menambahkan tutorial baru untuk menggunakan cetak biru AWS Project Development Kit (AWS PDK) dalam proyek Amazon. CodeCatalyst	April 9, 2024
Konten baru: Menggunakan tugas untuk memecah masalah menjadi tujuan yang lebih kecil	Menambahkan konten untuk mendukung peluncuran tugas dalam masalah. Tugas dapat ditambahkan ke masalah untuk memecah, mengatur, dan melacak pekerjaan masalah itu lebih lanjut.	April 4, 2024
Konten yang diperbarui: Berbagi data antar tindakan dalam alur kerja menggunakan artefak	Memperbarui Berbagi data antar tindakan dalam alur kerja menggunakan artefak topik untuk menyertakan dua subtopik baru: Dapatkah saya membagikan artefak tanpa menentukannya sebagai output dan input? dan Bisakah saya berbagi artefak antar alur kerja?	April 2, 2024

Konten yang diperbarui: Keterbatasan GitHub Tindakan di CodeCatalyst	Memperbarui Keterbatasan GitHub Tindakan di CodeCatalyst topik untuk menunjukkan bahwa GitHub Tindakan berjalan pada image Docker lingkungan runtime yang lebih lama.	April 2, 2024
Konten baru: Tindakan “AWS CDK menyebarkan” definisi YAMAL	Menambahkan `CloudAssemblyRootPath` properti baru keTindakan “AWS CDK menyebarkan” definisi YAMAL.	April 1, 2024
Konten yang diperbarui: Menentukan gambar Docker lingkungan runtime	Memperbarui Menentukan gambar Docker lingkungan runtime topik untuk menyertakan informasi tentang gambar lingkungan runtime Maret 2024 yang baru.	Maret 26, 2024
Konten yang diperbarui: Bekerja dengan peran	Informasi izin peran terkonsolidasi ke dalam satu tabel. Tabel ada dalam Melihat izin yang tersedia untuk setiap peran topik baru.	Maret 18, 2024
Konten baru: Lihat semua spasi dan proyek untuk pengguna	Menambahkan informasi tentang melihat daftar di halaman beranda pengguna yang menampilkan setiap CodeCatalyst spasi atau proyek untuk pengguna yang masuk. CodeCatalyst Lihat Lihat semua spasi dan proyek untuk pengguna.	Maret 18, 2024

Konten baru: Contoh: Pemicu dengan tarikan dan cabang	Menambahkan contoh pemicu permintaan tarik. Membuat koreksi kecil di seluruh Memulai alur kerja berjalan secara otomatis dengan pemicu topik.	Maret 11, 2024
Konten yang diperbarui: Bekerja dengan peran	Memperbarui dokumentasi untuk peran untuk menyertakan izin untuk membuat, menghapus, dan melihat lingkungan.	Maret 4, 2024
Konten yang diperbarui: Tutorial: Menggunakan fitur AI generatif	Memperbarui tutorial untuk mencerminkan perubahan saat membuat dan menetapkan masalah ke Amazon Q.	Maret 4, 2024
Konten baru: Masalah komponen	Menambahkan konten baru tentang cara bekerja dengan komponen masalah sebagai pengembang cetak biru khusus.	Februari 27, 2024
Konten yang diperbarui: Jenis tindakan	Memperbarui CodeCatalyst Tindakan Lab topik untuk menyertakan daftar tindakan CodeCatalyst Labs.	Februari 21, 2024
Konten yang diperbarui: Bekerja dengan permintaan tarik	Memperbarui dokumentasi untuk mencerminkan fungsionalitas baru dengan aturan persetujuan dan persyaratan utama untuk menggabungkan permintaan tarik.	Februari 15, 2024

Konten yang diperbarui: Menggabungkan permintaan tarik	Menambahkan dokumentasi untuk permintaan tarik untuk menyertakan informasi tentang penggantian persyaratan penggabungan untuk menggabungkan permintaan tarik yang belum menerima persetujuan dari pengulas wajib atau memenuhi aturan persetujuan.	Februari 15, 2024
Konten baru: Kelola aturan persetujuan	Menambahkan dokumentasi untuk permintaan tarik untuk menyertakan informasi tentang membuat dan mengelola aturan persetujuan.	Februari 15, 2024
Konten yang diperbarui: Bekerja dengan peran	Memperbarui dokumentasi untuk peran untuk menyertakan izin untuk bekerja dengan aturan persetujuan dan permintaan tarik.	Februari 14, 2024
Konten yang diperbarui: Bagaimana cara memperbaiki kesalahan “Definisi alur kerja memiliki *n* kesalahan”?	Memperbarui Bagaimana cara memperbaiki kesalahan “Definisi alur kerja memiliki *n* kesalahan”? bagian untuk menyertakan lebih banyak kiat pemecahan masalah.	Februari 9, 2024
Konten baru: Melihat status alur kerja	Menambahkan bagian yang menjelaskan status alur kerja.	Februari 9, 2024
Konten baru: Melihat status alur kerja	Menambahkan bagian yang menjelaskan status alur kerja.	Februari 9, 2024

Perbarui konten: Kuota untuk alur kerja	Memperbarui Kuota untuk alur kerja topik dengan Jumlah maksimum tindakan per alur kerja dan Jumlah maksimum lingkungan yang terkait dengan kuota Akun AWS per ruang.	Februari 7, 2024
Konten yang diperbarui: Pembuatan lingkungan	Memperbarui Pembuatan lingkungan bagian untuk menunjukkan bahwa Anda dapat menggunakan maksimum satu koneksi akun per lingkungan.	Januari 31, 2024
Konten baru: Repositori cetak biru GitHub kustom	Menambahkan konten baru untuk GitHub repositori yang dibuat tersedia untuk umum.	10 Januari 2024
Konten yang diperbarui: Mengkonfigurasi npm dengan CodeCatalyst	Instruksi konfigurasi umum yang diperbaru i untuk menggunakan npm dengan CodeCatal yst, dan menambahkan kejelasan seputar `always-auth=true` opsi.	Januari 5, 2024
Konten yang diperbarui: Bekerja dengan permintaan tarik	Memperbarui dokumenta si untuk mencerminkan fungsionalitas baru dengan fitur AI generatif di CodeCatal yst.	28 November 2023

Konten yang diperbarui: Membuat masalah	Memperbarui dokumentasi untuk mencerminkan fungsionalitas baru dengan fitur AI generatif di CodeCatalyst.	28 November 2023
Konten baru: Tutorial: Menggunakan fitur AI generatif	Menambahkan tutorial untuk menggunakan fitur AI generatif di Amazon CodeCatalyst.	28 November 2023
Konten baru: Cetak biru khusus dan manajemen siklus hidup	Menambahkan konten baru untuk menggunakan cetak biru kustom dan fitur manajemen siklus hidup di Amazon. CodeCatalyst	27 November 2023
Konten yang diperbarui: Tutorial: Membuat proyek dengan cetak biru aplikasi web tiga tingkat Modern	Memperbarui tutorial dengan perbaikan dan informasi pemecahan masalah.	22 November 2023
Konten yang diperbarui: Memulai alur kerja berjalan secara otomatis dengan pemicu	Memperbaiki beberapa contoh dan deskripsi yang terkait dengan pemicu permintaan tarik. Menambahkan Memicu pertimbangan saat bercabang bagian.	22 November 2023

Konten baru: Masuk dengan SSO	Menambahkan informasi tentang masuk dengan Single Sign-On (SSO) dan tautan ke informasi tentang pengaturan dan pengelola an CodeCatalyst ruang yang mendukung federasi identitas . Lihat Siapkan dan masuk ke CodeCatalyst dan Masuk dengan SSO.	17 November 2023
Konten yang diperbarui: Bekerja dengan peran	Memperbarui dokumentasi peran untuk menyertakan izin untuk bekerja dengan tim, koneksi VPC, sistem masuk tunggal, dan sumber daya mesin.	16 November 2023
Konten yang diperbarui: Bekerja dengan permintaan tarik	Memperbarui dokumenta si untuk mencerminkan perubahan dalam cara perubahan untuk permintaan tarik ditampilkan.	16 November 2023
Konten yang diperbarui: Kuota untuk CodeCatalyst	Memperbarui Kuota untuk CodeCatalyst topik dengan Jumlah maksimum koneksi VPC untuk kuota ruang.	16 November 2023
Konten baru: Mengelola tim untuk ruang dan CodeCatalyst proyek	Menambahkan informasi tentang menggunakan tim dengan spasi. Lihat Mengizink an akses ruang menggunak an tim dan Mengizinkan akses proyek menggunakan tim.	16 November 2023

Konten baru: Mengelola sumber daya mesin untuk cetak biru dan alur kerja dalam suatu ruang	Menambahkan informasi tentang penggunaan sumber daya mesin dengan spasi. Lihat Memungkinkan akses ruang untuk sumber daya mesin .	16 November 2023
Konten baru: Mengelola sumber daya mesin untuk cetak biru dan alur kerja dalam sebuah proyek CodeCatalyst	Menambahkan informasi tentang penggunaan sumber daya mesin dengan CodeCatalyst proyek. Lihat Mengizinkan akses proyek untuk sumber daya mesin.	16 November 2023
Konten baru: Mengaitkan koneksi VPC dengan lingkungan	Menambahkan dokumentasi untuk mengaitkan koneksi VPC dengan lingkungan, yang dapat digunakan dalam alur kerja.	16 November 2023
Konten baru: Mengaitkan koneksi VPC ke Lingkungan Dev	Menambahkan dokumentasi untuk menggunakan Lingkungan Dev dengan koneksi VPC.	16 November 2023
Konten baru	Publikasi awal Panduan CodeCatalyst Administrator Amazon.	16 November 2023
Konten baru: Tindakan “AWS CDK menyebarkan” definisi YAMAL	Menambahkan `CdkCliVersion` properti baru ke Tindakan “AWS CDK menyebarkan” definisi YAMAL dan Tindakan “AWS CDK bootstrap” definisi YAMAL.	14 November 2023

Konten yang diperbarui: Bekerja dengan peran	Memperbarui dokumentasi untuk peran untuk menyertakan izin untuk bekerja dengan aturan cabang.	13 November 2023
Konten yang diperbarui: Memecahkan masalah dengan repositori sumber, alur kerja, dan Lingkungan Pengembang	Memperbarui topik pemecahan masalah untuk menyertakan informasi tentang bekerja dengan aturan cabang.	13 November 2023
Konten yang diperbarui: Membangun dan menguji definisi YAMAL tindakan	Memperbarui dokumentasi untuk `Environment` properti. Sekarang menjadi bidang opsional untuk tindakan build dan test.	13 November 2023
Konten baru: Kelola aturan cabang	Menambahkan dokumentasi untuk cabang untuk menyertakan informasi tentang melihat aturan apa pun untuk cabang di repositori sumber, dan membuat serta mengelola aturan cabang.	13 November 2023
Konten yang diperbarui: Bekerja dengan permintaan tarik	Memperbarui dokumentasi untuk mencerminkan perubahan dalam cara informasi tentang permintaan tarik ditampilkan.	10 November 2023
Konten yang diperbarui: Caching file di antara alur kerja berjalan	Diperbarui dokumentasi untuk memasukkan batasan caching file.	10 November 2023
Konten yang diperbarui: Tutorial: Menyebarkan aplikasi ke Amazon EKS	Memperbarui dokumentasi untuk menyebutkan cetak biru Penerapan Aplikasi EKS.	9 November 2023

Konten baru: Paket di CodeCatalyst	Ditambahkan dokumentasi untuk menggunakan paket di CodeCatalyst.	1 November 2023
Konten baru dan diperbarui: Bekerja dengan peran	Memperbarui dokumentasi untuk empat peran baru di CodeCatalyst: Power user, Limited access, Reviewer, dan Read only.	1 November 2023
Konten yang diperbarui: Mengekspor parameter GitHub output sehingga tindakan lain dapat menggunakannya	Memperbarui contoh untuk menggunakan file `GITHUB_OUTPUT` lingkungan alih-alih `set-output` perintah. Menggunakan file lingkungan adalah GitHub metode yang direkomendasikan untuk mengatur parameter output.	24 Oktober 2023
Konten baru: Memulai alur kerja berjalan secara otomatis dengan pemicu	Ditambahkan dokumentasi untuk pemicu jadwal.	16 Oktober 2023
Konten yang diperbarui: Tindakan “Terapkan ke Kluster Kubernetes” Definisi YAMAL	Menambahkan informasi tentang penggunaan `CodeCatalystWorkflowDevelopmentRole-`*`spaceName`* peran ke Tindakan “Terapkan ke Kluster Kubernetes” Definisi YAMAL dan Tutorial: Menyebarkan aplikasi ke Amazon EKS topik.	September 22, 2023

Konten yang diperbarui: Nama peran dan kebijakan baru untuk CodeCatalystWorkflowDevelopmentRole-*spaceName* peran tersebut	Memperbarui langkah dan deskripsi peran untuk perubahan nama peran pengembang menjadi CodeCatalystWorkflowDevelopmentRole-*spaceName* . Peran pengembang sekarang menggunakan kebijakan `AdministratorAccess` AWS terkelola. Lihat Memahami peran CodeCatalystWorkflowDevelopmentRole-*spaceName* layanan dan Membuat peran ruang dan pengembangan pertama Anda (dimulai tanpa undangan).	20 September 2023
Konten yang diperbarui: Mengkonfigurasi dan menggunakan variabel dalam alur kerja	Memperkenalkan dua konsep baru: variabel yang ditentukan pengguna dan variabel yang telah ditentukan. Konsep-konsep ini harus membuat Mengkonfigurasi dan menggunakan variabel dalam alur kerja bagian lebih mudah dibaca dan dipahami.	September 19, 2023
Konten yang diperbarui: Bekerja dengan komit	Memperbarui dokumentasi untuk mencerminkan perubahan dalam informasi yang ditampilkan dan memberikan rincian tentang melihat komit dengan beberapa orang tua.	7 September 2023

Konten baru: Memulai alur kerja berjalan secara otomatis dengan pemicu	Menambahkan contoh berikut ke Memulai alur kerja berjalan secara otomatis dengan pemicu topik: Contoh: Pemicu 'push to main' sederhana	September 6, 2023
Konten yang diperbarui: Bekerja dengan permintaan tarik	Memperbarui dokumenta si untuk mencerminkan perubahan urutan tampilan untuk cabang sumber dan cabang tujuan saat membuat permintaan tarik.	Agustus 30, 2023
Konten baru: Lihat dan ubah cabang default	Menambahkan dokumentasi untuk cabang untuk menyertak an informasi tentang melihat dan mengubah cabang default untuk repositori sumber.	Agustus 30, 2023
Konten yang diperbarui: Tindakan “Terapkan ke Kluster Kubernetes” Definisi YAMAL	Menambahkan catatan tentang Helm dan Kustomize ke deskripsi Manifests properti di. Tindakan “Terapkan ke Kluster Kubernetes” Definisi YAMAL	15 Agustus 2023
Konten baru: Mengelola lampiran masalah	Menambahkan dokumenta si untuk bekerja dengan dan mengelola lampiran tentang masalah.	15 Agustus 2023
Konten yang diperbarui: Memulai alur kerja berjalan secara otomatis dengan pemicu	Meningkatkan dan memperlua s dokumentasi yang terkait dengan pemicu alur kerja.	11 Agustus 2023

Konten baru: Memecahkan masalah izin peran	Menambahkan informasi tentang memperbarui izin peran untuk menjalankan alur kerja yang memerlukan akses ke Amazon. CodeGuru Lihat Alur kerja cetak biru aplikasi web tiga tingkat modern OnPullRequestgagal dengan kesalahan izin untuk Amazon CodeGuru.	11 Agustus 2023
Konten baru: Bagaimana cara memperbaiki pesan “Alur kerja tidak aktif”?	Ditambahkan topik pemecahan masalah berikut: Bagaimana cara memperbaiki pesan “Alur kerja tidak aktif”?	11 Agustus 2023
Konten baru: Menerapkan aplikasi ke Amazon Elastic Kubernetes Service dengan alur kerja	Menambahkan dokumenta si untuk aksi klaster Deploy ke Kubernetes. Untuk informasi selengkapnya, lihat Menerapkan aplikasi ke Amazon Elastic Kubernetes Service dengan alur kerja dan Tutorial: Menyebarkan aplikasi ke Amazon EKS.	Juli 27, 2023

Pembaruan tentang bagaimana peristiwa manajemen dicatat untuk CodeCatalyst ruang	Menambahkan informasi tentang bagaimana peristiwa manajemen dicatat untuk tindakan tertentu dalam CodeCatalyst ruang dengan AWS CloudTrail. Menambahkan informasi tentang bagaimana semua peristiwa dalam ruang dapat dilihat dengan list-event-logs perintah. Lihat Memantau peristiwa dan panggilan API menggunakan logging.	Juli 20, 2023
Konten yang diperbarui: Memulai alur kerja berjalan secara otomatis dengan pemicu	Memperbarui dokumentasi untuk menunjukkan bahwa pemicu permintaan tarik sekarang didukung dengan repositori GitHub sumber. Sebelumnya, pemicu permintaan tarik hanya didukung dengan repositori CodeCatalyst sumber.	14 Juli 2023
Konten yang diperbarui: Kuota untuk alur kerja	Memperbarui Kuota untuk alur kerja topik dengan Jumlah waktu maksimum suatu tindakan dapat menjalankan kuota.	Juni 27, 2023
Konten yang diperbarui: Alur kerja definisi YAMAL	Memperbaiki kesalahan pemformatan di blok `Compute` kode.	Juni 27, 2023

Konten yang diperbarui: Perlindungan data	Diperbarui dokumentasi untuk memasukkan informasi tambahan tentang replikasi data.	26 Juni 2023
Konten baru: Menentukan versi mayor, minor, atau patch dari suatu tindakan	Menambahkan Menentukan versi mayor, minor, atau patch dari suatu tindakan topik.	Juni 21, 2023
Konten yang diperbarui: Menyebarkan ke dalam Akun AWS dan VPC dengan lingkungan CodeCatalyst	Mengklarifikasi Tindakan apa yang mendukung lingkungan? bagian tersebut.	Juni 14, 2023
Konten diperbarui: dokumentasi masalah direorganisasi	Menata ulang sebagian besar dokumentasi masalah agar lebih selaras dengan keseluruhan kumpulan dokumentasi dan alur pengguna.	31 Mei 2023
Konten yang diperbarui: Masalah tampilan switcher	Memperbarui berbagai alur pengguna agar selaras dengan pengalih tampilan masalah yang diperbarui.	31 Mei 2023
Konten yang diperbarui: Mengelola pemberitahuan	Memperbarui dokumentasi untuk pemberitahuan untuk menyertakan informasi tentang mengkonfigurasi pemberitahuan Slack pribadi.	30 Mei 2023
Konten yang diperbarui: Mengelola pemberitahuan	Memperbarui dokumentasi untuk pemberitahuan untuk menyertakan informasi tentang mengkonfigurasi pemberitahuan Slack pribadi.	30 Mei 2023

Konten baru: model CodeCatalyst kepercayaan	Menambahkan topik baru dengan informasi tentang model kepercayaan, yang memungkinkan CodeCatalyst untuk mengambil peran layanan dalam terhubung Akun AWS. Menambahkan bagian baru tentang prinsip layanan yang ditentukan untuk. CodeCatalyst Lihat Memahami model CodeCatalyst kepercayaan.	Mei 20, 2023
Konten yang diperbarui: Menghubungkan alur kerja ke repositori sumber	Sederhanakan instruksi diMereferensikan file dalam repositori sumber.	10 Mei 2023
Konten yang diperbarui: Kuota untuk alur kerja	Memperbarui Kuota untuk alur kerja topik dengan Panjang maksimum kuota nilai variabel keluaran.	10 Mei 2023
Konten baru: Berbagi data antar tindakan dalam alur kerja menggunakan artefak	Menambahkan dua contoh: Contoh: Mereferensikan file dalam satu artefak danContoh: Mereferensikan file dalam artefak saat ada WorkflowSource .	10 Mei 2023
Konten yang diperbarui: Bekerja dengan permintaan tarik	Memperbarui dokumentasi untuk permintaan tarik untuk menyertakan informasi tentang mengonfigurasi preferensi email untuk peristiwa permintaan tarik.	April 21, 2023

Konten yang diperbarui: Mengelola pemberitahuan	Memperbarui dokumentasi notifikasi untuk menyertakan informasi tentang mengonfig urasi preferensi email untuk peristiwa permintaan tarik.	April 21, 2023
Pembaruan kebijakan terkelola	MenambahkanKebijakan terkelola AWS: AmazonCod eCatalystFullAccess,Kebijakan terkelola AWS: AmazonCod eCatalystReadOnlyAccess, dan kebijakan Kebijakan terkelola AWS: AmazonCod eCatalystSupportAccess terkelola. Lihat CodeCatalyst pembaruan kebijakan AWS terkelola.	20 April 2023
Konten baru: Menghapus target penerapan	Menambahkan Menghapus target penerapan topik.	20 April 2023
Konten baru: Jenis tindakan	Menambahkan CodeCatalyst tindakan topik.	20 April 2023
Pembaruan untuk mengelola pengguna dengan peran administrator Space di ruang	Menambahkan informasi tentang menghapus atau mengubah peran untuk pengguna dengan peran administrator Space dalam spasi. Lihat Menghapus atau mengubah peran untuk pengguna dengan peran administrator Space.	19 April 2023

Pembaruan untuk mengelola Lingkungan Dev	Menambahkan informasi tentang mengelola Lingkungan Dev sebagai administrator Space. Lihat Mengelola Lingkungan Pengembang untuk suatu ruang.	19 April 2023
Konten yang diperbarui: Menemukan dan melihat masalah	Menata ulang Menemukan dan melihat masalah topik dan subtopik.	19 April 2023
Konten yang diperbarui: Mengkonfigurasi gambar Docker lingkungan komputasi dan runtime untuk alur kerja	Menambahkan dukungan untuk arsitektur Arm64 di Amazon Linux 2.	19 April 2023
Konten baru: Memindahkan masalah dalam grup	Menambahkan dokumentasi untuk memindahkan masalah dalam grup di Dewan dan Semua tampilan masalah.	19 April 2023
Konten yang diperbarui: Kuota untuk alur kerja	Memperbarui Kuota untuk alur kerja topik dengan kuota yang hilang, dan memperbarui ukuran total maksimum kuota variabel keluaran tindakan tunggal menjadi 120 KB (dari 2 KB).	18 April 2023
Konten baru: Melihat kode sumber tindakan	Menambahkan Melihat kode sumber tindakan topik.	18 April 2023
Konten baru: Mencoba kembali kasus uji laporan	Menambahkan Mencoba kembali kasus uji laporan topik.	11 April 2023
Konten baru: Menghentikan alur kerja	Menambahkan Menghentikan alur kerja topik.	April 10, 2023

Konten baru: Menambahkan bagian untuk menandai sumber daya untuk koneksi akun antara AWS dan Amazon CodeCatalyst	Menambahkan informasi untuk menandai sumber daya koneksi akun dan mengelola kebijakan IAM untuk sumber daya koneksi. Lihat Menggunakan tag untuk mengontrol akses ke sumber daya koneksi akun dan CodeCatalyst referensi izin.	6 April 2023
Konten baru: Jenis tindakan	Menambahkan Jenis tindakan topik.	6 April 2023
Konten yang diperbarui: Tindakan “Terapkan AWS CloudFormation tumpukan” definisi YAMAL	Memperbarui deskripsi `parameter-overrides` properti. Sekarang mendukung file JSON.	5 April 2023
Konten baru: Membuat proyek CodeCatalyst dengan GitHub repositori tertaut	Menambahkan bagian baru Membuat proyek berjudul Membuat proyek dengan repositori pihak ketiga yang ditautkan dengan instruksi untuk membuat proyek yang menautkan ke GitHub repositori Anda.	5 April 2023
Konten yang diperbarui: Bekerja dengan notifikasi	Diperbarui dokumentasi untuk pemberitahuan untuk menyertakan informasi tentang mengkonfigurasi email tentang peristiwa proyek.	31 Maret 2023
Konten baru	Publikasi awal panduan Amazon CodeCatalyst Action Development Kit.	31 Maret 2023

Konten yang diperbarui: Merestrukturisasi bagian Spasi di Amazon CodeCatalyst	Memperbarui bagian Spaces dengan menghapus halaman arahan dan mengkonso lidasikan topik.	29 Maret 2023
Konten yang diperbarui: Tutorial: Menyebarkan aplikasi ke Amazon ECS	Diubah Langkah 1: Siapkan AWS pengguna dan AWS CloudShell untuk menjelask an cara membuat pengguna di AWS IAM Identity Center alih-alih AWS Identity and Access Management. Membuat pengguna IAM tidak lagi direkomendasikan.	Maret 23, 2023
Konten yang diperbarui: Bekerja dengan peran	Memperbarui dokumenta si untuk peran administr ator Space, administrator Proyek, dan Kontributor untuk menyertakan izin untuk menautkan masalah untuk menarik permintaan.	13 Maret 2023
Konten yang diperbarui: Bekerja dengan permintaan tarik	Memperbarui dokumenta si untuk permintaan tarik untuk menyertakan informasi tentang masalah penautan ke permintaan tarik.	13 Maret 2023
Konten yang diperbarui: Bekerja dengan masalah	Memperbarui dokumenta si untuk masalah untuk menyertakan informasi tentang masalah penautan ke permintaan tarik.	13 Maret 2023

Konten baru: Melihat status dan detail semua proses dalam proyek Anda	Menambahkan bagian yang menjelaskan halaman jalankan alur kerja agregat baru.	8 Maret 2023
Konten baru: Bagaimana cara memperbaiki kesalahan “Definisi alur kerja memiliki *n* kesalahan”?	Menambahkan bagian tentang cara memecahkan masalah kesalahan “Definisi alur kerja memiliki kesalahan”.	7 Maret 2023
Konten yang diperbarui: Membuat alur kerja	Memperbarui instruksi untuk mencerminkan UI baru.	3 Maret 2023
Konten baru: Menginteg rasikan universal-test-runner ke dalam tindakan uji	Menambahkan Menginteg rasikan universal-test-runner ke dalam tindakan uji topik.	3 Maret 2023
Konten yang diperbarui: Bangun, uji, dan terapkan dengan alur kerja di CodeCatalyst	Memperbarui berbagai bagian untuk mencerminkan repositor i sumber baru, cabang, dan filter nama alur kerja pada halaman ringkasan Alur Kerja.	2 Maret 2023
Konten baru: Melacak status penerapan dengan komit	Menambahkan bagian tentang melihat kualitas kode dan status penerapan dengan komit.	27 Februari 2023
Konten baru: Variabel yang dihasilkan oleh sumber (BranchName“” dan CommidId “”)	Ditambahkan variabel `BranchName` standar baru.	16 Februari 2023

Konten yang diperbarui: Mengelola anggota luar angkasa di Amazon CodeCatalyst	Informasi terbaru tentang mengubah peran anggota, mengundang anggota, dan menghapus anggota dalam dua tabel baru berdasarkan peran yang ditetapkan pengguna. CodeCatalyst	15 Februari 2023
Konten yang diperbarui: Menambahkan langkah-langkah untuk manajemen PAT di CodeCatalyst konsol Amazon	Menambahkan langkah-langkah untuk melihat, membuat, dan menghapus PAT di konsol.	15 Februari 2023
Konten yang diperbarui: Menentukan gambar Docker lingkungan runtime	Menambahkan lebih banyak alat ke tabel versi alat gambar Default.	10 Januari 2023
Konten yang diperbarui: Berbagi data antar tindakan dalam alur kerja menggunakan artefak	Memperbaiki jalur artefak.	Januari 3, 2023
Konten yang diperbarui: GitHub Tindakan “Tindakan” definisi YAMAL	Memperbaiki cuplikan kode di bagian. `Steps`	Januari 3, 2023
Konten yang diperbarui: Menghubungkan alur kerja ke repositori sumber	Memperbaiki jalur sumber.	Januari 3, 2023
Konten yang diperbarui: Memperbarui permintaan tarik	Memperbarui dokumentasi untuk menyertakan informasi tentang memperbarui pengulas yang diperlukan atau opsional untuk permintaan tarik.	Desember 23, 2022

Konten baru: Caching file di antara alur kerja berjalan	Ditambahkan halaman untuk file caching dalam alur kerja.	Desember 20, 2022
Konten yang diperbarui: Bekerja dengan permintaan tarik	Memperbarui dokumentasi untuk permintaan tarik untuk menyertakan informasi tentang pemberitahuan.	16 Desember 2022
Konten baru: Tindakan “AWS CDK menyebarkan” definisi YAMAL	Menambahkan `CdkRootPath` properti baru.	16 Desember 2022
Konten baru: Berbagi komputasi di seluruh tindakan	Menambahkan Berbagi komputasi di seluruh tindakan topik.	14 Desember 2022
Konten yang diperbarui: Berbagi data antar tindakan dalam alur kerja menggunakan artefak	Contoh tetap yang menunjukkan cara menentukan artefak masukan.	13 Desember 2022
Konten baru: GitHub Tindakan “Tindakan” definisi YAMAL	Menambahkan halaman referensi khusus untuk GitHub tindakan Tindakan.	13 Desember 2022
Konten yang diperbarui: Kuota untuk proyek di CodeCatalyst	Memperbarui dokumentasi dengan maksimal 100 proyek dalam satu ruang.	Desember 2, 2022
Konten baru	Publikasi awal Panduan CodeCatalyst Pengguna Amazon.	Desember 1, 2022

AWS Glosarium

Untuk AWS terminologi terbaru, lihat AWS glosarium di Referensi.Glosarium AWS

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.