aws

Panduan Pengguna

AWS Studio Aplikasi

AWS Studio Aplikasi: Panduan Pengguna

Table of Contents

Apa itu AWS App Studio?

AWS App Studio adalah layanan bertenaga AI generatif yang menggunakan bahasa alami untuk membantu Anda membuat aplikasi kelas perusahaan. App Studio membuka pengembangan aplikasi untuk profesional teknis tanpa keterampilan pengembangan perangkat lunak, seperti manajer proyek TI, insinyur data, dan arsitek perusahaan. Dengan App Studio, Anda dapat dengan cepat membangun aplikasi yang aman dan dikelola sepenuhnya oleh AWS, tanpa perlu keahlian operasional.

Pembangun dapat menggunakan App Studio untuk membuat dan menerapkan aplikasi guna memodernisasi proses bisnis internal. Beberapa contoh kasus penggunaan adalah manajemen dan pelacakan inventaris, pemrosesan klaim, dan persetujuan kompleks untuk meningkatkan produktivitas karyawan dan hasil pelanggan.

Topik

Apakah Anda pengguna App Studio pertama kali?

Jika Anda pengguna pertama kali App Studio, kami sarankan Anda memulai dengan membaca bagian berikut:

- Untuk pengguna dengan peran administrator yang akan menyiapkan App Studio, mengelola pengguna dan akses, dan mengonfigurasi konektor dengan layanan lain AWS atau pihak ketiga, lihat AWS Konsep App Studio danMenyiapkan dan masuk ke AWS App Studio.
- Untuk pembangun yang akan membuat dan mengembangkan aplikasi, lihat AWS Konsep App Studio danMemulai dengan AWS App Studio.

AWS Konsep App Studio

Kenali konsep utama App Studio untuk membantu mempercepat pembuatan aplikasi dan mengotomatisasi proses untuk tim Anda. Konsep ini mencakup istilah yang digunakan di seluruh App Studio untuk administrator dan builder.

Topik

Peran admin

Admin adalah peran yang dapat ditetapkan ke grup di App Studio. Admin dapat mengelola pengguna dan grup dalam App Studio, menambah dan mengelola konektor, dan mengelola aplikasi yang dibuat oleh pembangun. Selain itu, pengguna dengan peran Admin memiliki semua izin yang disertakan dengan peran Builder.

Hanya pengguna dengan peran Admin yang memiliki akses ke Hub Admin, yang berisi alat untuk mengelola peran, sumber data, dan aplikasi.

Aplikasi (aplikasi)

Aplikasi (app) adalah program perangkat lunak tunggal yang dikembangkan untuk pengguna akhir untuk menyelesaikan tugas-tugas tertentu. Aplikasi di App Studio menyertakan aset seperti halaman dan komponen UI, otomatisasi, dan sumber data yang dapat berinteraksi dengan pengguna.

Otomatisasi

Otomatisasi adalah bagaimana Anda mendefinisikan logika bisnis aplikasi Anda. Komponen utama otomatisasi adalah: pemicu yang memulai otomatisasi, urutan satu atau lebih tindakan, parameter input yang digunakan untuk meneruskan data ke otomatisasi, dan output.

Tindakan otomatisasi

Tindakan otomatisasi, yang biasa disebut sebagai tindakan, adalah langkah logika individual yang membentuk otomatisasi. Setiap tindakan melakukan tugas tertentu, baik itu mengirim email, membuat catatan data, menjalankan fungsi Lambda, atau memanggil. APIs Tindakan ditambahkan ke otomatisasi dari pustaka tindakan, dan dapat dikelompokkan ke dalam pernyataan atau loop bersyarat.

Peran pembangun

Builder adalah peran yang dapat ditetapkan ke grup di App Studio. Pembangun dapat membuat dan membangun aplikasi. Builder tidak dapat mengelola pengguna atau grup, menambah atau mengedit instance konektor, atau mengelola aplikasi pembangun lain.

Pengguna dengan peran Builder memiliki akses ke Builder Hub, yang berisi detail tentang sumber daya seperti aplikasi yang dapat diakses oleh builder bersama dengan informasi bermanfaat seperti sumber belajar.

Komponen

Komponen adalah item fungsional individual dalam UI aplikasi Anda. Komponen terkandung dalam halaman, dan beberapa komponen dapat berfungsi sebagai wadah untuk komponen lainnya. Komponen menggabungkan elemen UI dengan logika bisnis yang Anda inginkan agar elemen UI tersebut berfungsi. Misalnya, salah satu jenis komponen adalah formulir, di mana pengguna dapat memasukkan informasi di bidang dan, setelah dikirimkan, informasi tersebut ditambahkan sebagai catatan basis data.

Lingkungan pengembangan

Lingkungan Pengembangan adalah alat visual untuk membangun aplikasi. Lingkungan ini mencakup tab berikut untuk membangun aplikasi:

- Halaman: Di mana pembangun mendesain aplikasi mereka dengan halaman dan komponen.
- Otomatisasi: Di mana pembangun merancang logika bisnis aplikasi mereka dengan otomatisasi.
- Data: Di mana pembangun mendesain model data aplikasi mereka dengan entitas.

Lingkungan Pengembangan juga berisi konsol debug, dan jendela obrolan AI untuk mendapatkan bantuan kontekstual saat membangun. Pembangun dapat melihat pratinjau aplikasi mereka yang sedang berlangsung dari lingkungan Pengembangan.

Entitas

Entitas adalah tabel data di App Studio. Entitas berinteraksi langsung dengan tabel di sumber data. Entitas mencakup bidang untuk mendeskripsikan data di dalamnya, kueri untuk mencari dan mengembalikan data, dan pemetaan untuk menghubungkan bidang entitas ke kolom sumber data.

Konektor

Konektor adalah koneksi antara App Studio dan AWS layanan lainnya, seperti AWS Lambda Amazon Redshift, atau layanan pihak ketiga. Setelah konektor dibuat dan dikonfigurasi, builder dapat menggunakannya dan sumber daya yang terhubung ke App Studio dalam aplikasi mereka.

Hanya pengguna dengan peran Admin yang dapat membuat, mengelola, atau menghapus konektor.

Halaman

Halaman adalah wadah untuk komponen, yang membentuk UI aplikasi di App Studio. Setiap halaman mewakili layar antarmuka pengguna (UI) aplikasi Anda yang akan berinteraksi dengan pengguna Anda. Halaman dibuat dan diedit di tab Halaman di studio aplikasi.

Pemicu

Pemicu menentukan kapan, dan pada kondisi apa, otomatisasi akan berjalan. Beberapa contoh pemicu adalah `On click` untuk tombol dan `On select` untuk input teks. Jenis komponen menentukan daftar pemicu yang tersedia untuk komponen tersebut. Pemicu ditambahkan ke komponen dan dikonfigurasi di studio aplikasi.

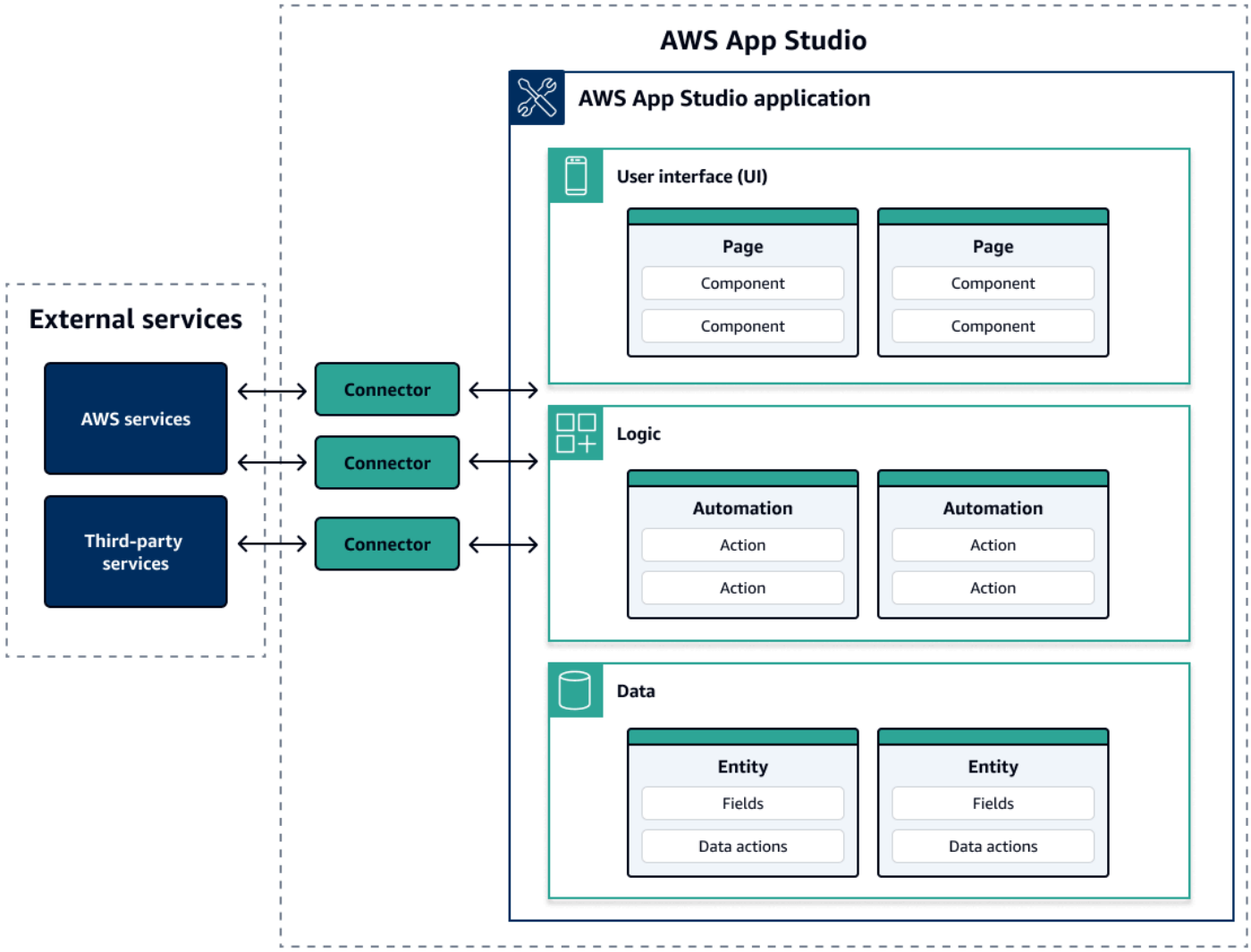
Cara kerja AWS App Studio

Ada beberapa konsep kunci yang perlu dipahami saat menggunakan AWS App Studio untuk membangun aplikasi. Topik ini mencakup dasar-dasar konsep atau sumber daya berikut:

- Menggunakan konektor untuk terhubung ke layanan lain untuk menggunakan sumber daya atau panggilan API mereka dalam aplikasi Anda. Misalnya, Anda dapat menggunakan konektor untuk menyimpan dan mengakses data, atau mengirim pemberitahuan dari aplikasi Anda.
- Menggunakan entitas untuk mengonfigurasi model data aplikasi Anda, yang menghubungkan aplikasi Anda dengan sumber data eksternal Anda.
- Menggunakan halaman dan komponen untuk membangun antarmuka pengguna (UI) aplikasi Anda.
- Menggunakan otomatisasi dan tindakan untuk mengimplementasikan logika atau perilaku aplikasi Anda.
- Siklus hidup pengembangan aplikasi di App Studio: membangun, menguji, dan menerbitkan.

Untuk informasi selengkapnya tentang konsep App Studio, lihatAWS Konsep App Studio.

Gambar berikut adalah diagram sederhana tentang bagaimana App Studio dan sumber dayanya diatur.



Dalam aplikasi di App Studio, halaman, otomatisasi, dan entitas semuanya berinteraksi satu sama lain. Anda menggunakan konektor untuk menghubungkan sumber daya ini ke layanan eksternal seperti penyedia data, penyimpanan, atau pemberitahuan. Agar berhasil membangun aplikasi, penting untuk memahami bagaimana semua konsep dan sumber daya ini berinteraksi satu sama lain.

Menghubungkan aplikasi Anda ke layanan lain

Salah satu manfaat terbesar menggunakan App Studio untuk membangun aplikasi adalah dapat dengan mudah mengintegrasikan aplikasi Anda dengan layanan lain. Di App Studio, Anda terhubung ke layanan lain dengan menggunakan konektor yang khusus untuk layanan dan sumber daya atau panggilan API yang ingin Anda gunakan dengan aplikasi Anda.

Anda membuat konektor di tingkat instans App Studio, dan bukan di aplikasi individual. Setelah Anda membuat konektor, Anda dapat menggunakannya di berbagai bagian aplikasi, tergantung pada layanan yang terhubung dan aplikasi.

Berikut ini adalah contoh fungsionalitas dalam aplikasi yang menggunakan konektor untuk terhubung ke layanan lain:

- Kasus penggunaan yang paling umum, digunakan di hampir semua aplikasi, adalah untuk menyimpan dan mengakses data yang digunakan dalam aplikasi dengan menghubungkan ke layanan AWS data seperti Amazon Redshift, Amazon DynamoDB, atau Amazon Aurora.
- Aplikasi yang memungkinkan mengunggah dan melihat gambar, seperti tanda terima, dapat menggunakan Amazon S3 untuk menyimpan dan mengakses file gambar.
- Aplikasi peringkas teks dapat mengirim input teks ke Amazon Bedrock dan menampilkan ringkasan yang dikembalikan.

> **Note**
>
> Anda harus memiliki peran Admin di App Studio untuk membuat konektor. Saat membuat konektor, Anda harus menyertakan kredensi dan informasi yang tepat tentang sumber daya atau panggilan API yang ingin Anda gunakan.

Mengonfigurasi model data aplikasi Anda

Data aplikasi Anda adalah informasi yang mendukung aplikasi. Di App Studio, Anda membuat dan menggunakan entitas yang mewakili berbagai jenis data yang Anda simpan dan kerjakan. Misalnya, dalam aplikasi pelacakan untuk pertemuan pelanggan, Anda mungkin memiliki tiga entitas yang mewakili pertemuan pelanggan, agenda, dan peserta.

Entitas berisi bidang yang memiliki tipe, seperti integer atau string, yang menggambarkan data yang disimpan. Meskipun Anda menggunakan entitas untuk menentukan model data Anda, Anda harus menghubungkan entitas Anda ke layanan penyimpanan data eksternal seperti Amazon Redshift atau Amazon DynamoDB untuk menyimpan data. Anda dapat menganggap entitas sebagai perantara antara aplikasi App Studio dan data di layanan eksternal.

Anda dapat menggunakan tindakan data untuk berinteraksi dengan data dalam aplikasi Anda dari komponen dan otomatisasi. Dua tindakan data yang paling umum digunakan adalah `getAll`

tindakan dan `getByID` tindakan. Misalnya, aplikasi Anda dapat menggunakan tindakan `getAll` data untuk mengisi tabel dengan data Anda, dan `getByID` tindakan untuk mengisi komponen detail dengan informasi lebih lanjut tentang entri data tertentu.

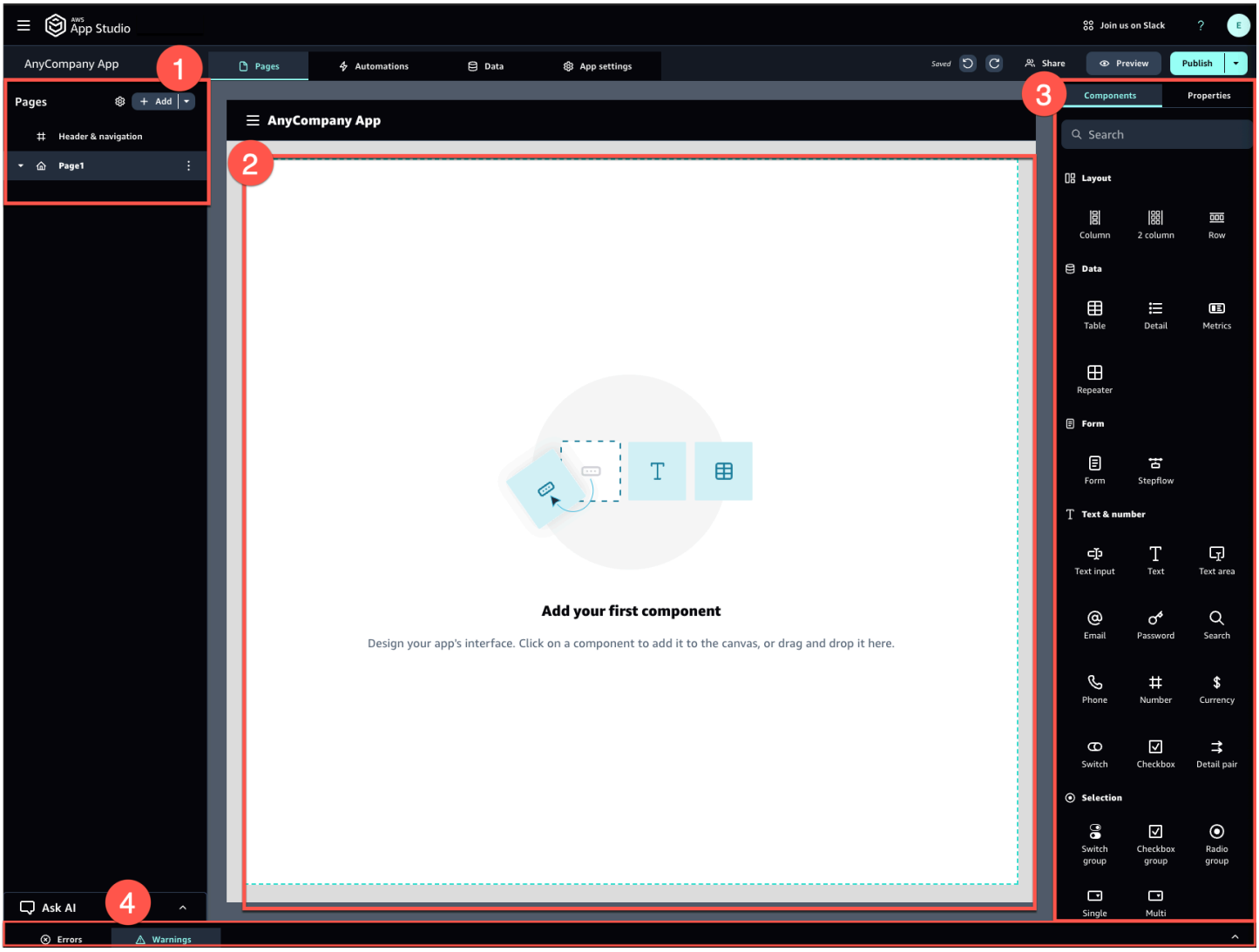
Anda juga dapat menambahkan data sampel ke entitas Anda untuk lebih mudah menguji aplikasi Anda tanpa perlu memanggil layanan eksternal.

Membangun UI aplikasi Anda

Di App Studio, Anda membangun UI aplikasi Anda dengan halaman dan komponen. Halaman adalah layar individual dari aplikasi Anda dan merupakan wadah untuk komponen. Komponen adalah blok bangunan UI aplikasi Anda. Ada banyak jenis komponen, seperti tabel, formulir, pemirsa gambar, dan tombol.

Gambar berikut menunjukkan tab Pages di studio aplikasi, tempat Anda menambahkan atau mengonfigurasi halaman dan komponen dalam aplikasi Anda. Area utama berikut disorot dan diberi nomor:

1. Panel Pages sisi kiri. Di sinilah Anda mengelola halaman, header aplikasi, dan pengaturan navigasi. Anda dapat melihat semua halaman dan komponen aplikasi Anda.
2. Kanvas, yang menampilkan komponen halaman saat ini. Anda dapat memilih komponen di kanvas untuk mengonfigurasi propertinya.
3. Panel Komponen atau Properti sisi kanan. Dengan tidak ada yang dipilih, panel Components ditampilkan, yang menampilkan daftar komponen yang dapat ditambahkan ke halaman Anda. Jika Anda memilih halaman atau komponen, panel Properties ditampilkan, di mana Anda mengkonfigurasi halaman atau komponen.
4. Panel Kesalahan dan Peringatan bawah. Panel ini menampilkan kesalahan atau peringatan apa pun dalam aplikasi Anda, yang paling sering disebabkan oleh masalah konfigurasi. Anda dapat memilih panel untuk memperluasnya dan melihat pesan.



Sebagai contoh, aplikasi di mana pengguna harus memasukkan informasi mungkin memiliki halaman dan komponen berikut:

- Halaman input yang menyertakan komponen formulir yang digunakan pengguna untuk mengisi dan mengirimkan informasi.
- Halaman tampilan daftar yang berisi komponen tabel dengan informasi tentang setiap input.
- Halaman tampilan terperinci yang berisi komponen detail dengan informasi lebih lanjut tentang setiap input.

Komponen dapat mencakup informasi statis atau data, seperti formulir dengan bidang yang ditentukan. Mereka juga dapat menyertakan informasi dinamis dengan menggunakan otomatisasi,

seperti penampil gambar yang mengambil gambar dari bucket Amazon S3 dan menampilkannya kepada pengguna.

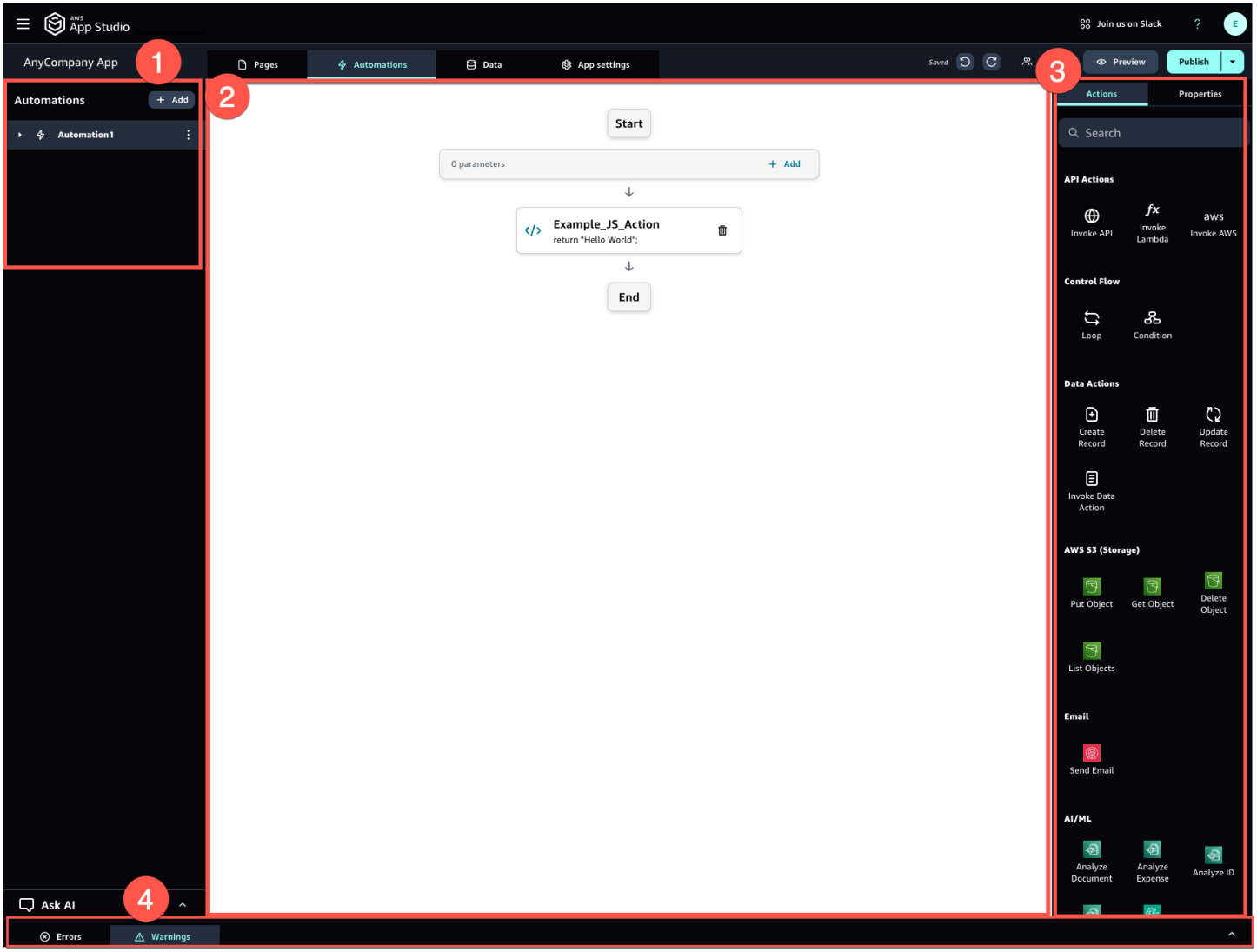
Penting untuk memahami konsep parameter halaman. Anda menggunakan parameter halaman untuk mengirim informasi dari satu halaman ke halaman lainnya. Contoh umum dari kasus penggunaan untuk parameter halaman adalah pencarian dan pemfilteran, di mana istilah pencarian dari satu halaman dikirim ke tabel atau daftar item untuk difilter di halaman lain. Contoh kasus penggunaan lainnya adalah melihat detail item, di mana pengidentifikasi item dikirim ke halaman penampil terperinci.

Menerapkan logika atau perilaku aplikasi Anda

Anda dapat menganggap logika atau perilaku aplikasi Anda sebagai fungsionalitas aplikasi. Anda dapat menentukan apa yang terjadi ketika pengguna memilih tombol, mengirimkan informasi, menavigasi ke halaman baru, atau berinteraksi dengan cara lain. Di App Studio, Anda menentukan logika aplikasi Anda dengan otomatisasi dan tindakan. Otomatisasi adalah wadah untuk tindakan, yang merupakan blok bangunan dari fungsionalitas otomatisasi.

Gambar berikut menunjukkan tab Otomasi studio aplikasi, tempat Anda menambahkan atau mengonfigurasi otomatisasi dan tindakannya di aplikasi Anda. Area utama berikut disorot dan diberi nomor:

- Panel otomasi sisi kiri. Di sinilah Anda mengelola otomatisasi. Anda dapat melihat semua otomatisasi dan tindakan aplikasi Anda.
- Kanvas, yang menampilkan otomatisasi saat ini. Ini menampilkan parameter otomatisasi yang dikonfigurasi (yang dijelaskan nanti di bagian ini) dan tindakan. Anda dapat memilih komponen di kanvas untuk mengonfigurasi propertinya.
- Panel Actions dan Properties sisi kanan. Dengan tidak ada yang dipilih, panel Actions ditampilkan. Ini menampilkan daftar tindakan yang dapat ditambahkan ke otomatisasi Anda. Jika Anda memilih otomatisasi, Anda dapat melihat dan mengonfigurasi propertinya, seperti input dan output otomatisasi. Jika Anda memilih tindakan, Anda dapat melihat dan mengonfigurasi properti tindakan.
- Panel Kesalahan dan Peringatan bawah. Panel ini menampilkan kesalahan atau peringatan dalam aplikasi Anda (paling sering dari masalah konfigurasi). Anda dapat memilih panel untuk memperluasnya dan melihat pesan.



Otomatisasi bisa sederhana (seperti menambahkan angka dan mengembalikan hasilnya), atau lebih kuat (seperti mengirim input ke layanan lain dan mengembalikan hasilnya). Komponen utama otomatisasi adalah sebagai berikut:

- Pemicu, yang menentukan kapan otomatisasi dijalankan. Contohnya adalah ketika pengguna menekan tombol di UI.
- Input otomatisasi, yang mengirimkan informasi ke otomatisasi. Anda menentukan input otomatisasi dengan parameter otomatisasi. Misalnya, jika Anda ingin menggunakan Amazon Bedrock untuk mengembalikan ringkasan teks kepada pengguna, Anda mengonfigurasi teks yang akan diringkas sebagai parameter otomatisasi.
- Tindakan, yang merupakan blok bangunan fungsionalitas otomatisasi. Anda dapat menganggap setiap tindakan sebagai langkah dalam otomatisasi. Tindakan dapat memanggil APIs, memanggil kustom JavaScript, membuat catatan data, dan melakukan fungsi lainnya. Anda juga dapat

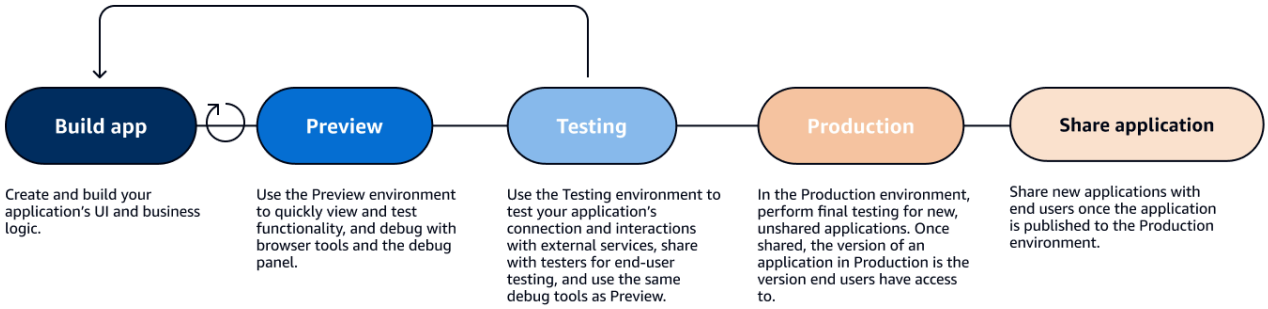
mengelompokkan tindakan ke dalam loop atau kondisi untuk menyesuaikan fungsionalitas lebih lanjut. Anda juga dapat memanggil otomatisasi lain dengan tindakan.

- Output otomatisasi, yang dapat Anda gunakan dalam komponen atau bahkan otomatisasi lainnya. Misalnya, output otomatisasi dapat berupa teks yang ditampilkan dalam komponen teks, gambar yang akan ditampilkan dalam komponen penampil gambar, atau input ke otomatisasi lain.

Siklus hidup pengembangan aplikasi Anda

Siklus hidup pengembangan aplikasi Anda mencakup tahapan berikut: pembuatan, pengujian, dan penerbitan. Ini disebut siklus, karena Anda mungkin akan iterasi melalui dan di antara tahap-tahap ini saat Anda membuat dan mengulangi aplikasi Anda.

Gambar berikut menunjukkan timeline yang disederhanakan dari siklus hidup pengembangan aplikasi di App Studio:



App Studio menawarkan berbagai alat untuk mendukung siklus hidup aplikasi Anda. Alat-alat ini mencakup tiga lingkungan berbeda berikut, yang ditunjukkan pada diagram sebelumnya:

- Lingkungan Pratinjau, tempat Anda dapat melihat pratinjau aplikasi untuk melihat tampilannya bagi pengguna akhir, dan menguji fungsionalitas tertentu. Gunakan lingkungan Pratinjau untuk menguji dan mengulangi aplikasi Anda dengan cepat tanpa perlu mempublikasikannya. Aplikasi di lingkungan pratinjau tidak berkomunikasi atau mentransfer data dengan layanan eksternal. Ini berarti Anda tidak dapat menguji interaksi dan fungsionalitas yang bergantung pada layanan eksternal di lingkungan Pratinjau.
- Lingkungan Pengujian, tempat Anda dapat menguji koneksi dan interaksi aplikasi Anda dengan layanan eksternal. Di sinilah Anda dapat melakukan pengujian pengguna akhir dengan membagikan versi yang dipublikasikan ke lingkungan Pengujian ke grup penguji.
- Lingkungan Produksi, tempat Anda dapat melakukan pengujian akhir aplikasi baru sebelum membagikannya dengan pengguna akhir. Setelah aplikasi dibagikan, versi aplikasi yang

dipublikasikan ke lingkungan Produksi adalah versi yang akan dilihat dan digunakan pengguna akhir.

Pelajari selengkapnya

Sekarang setelah Anda mengetahui dasar-dasar cara kerja pengembangan aplikasi di App Studio, Anda dapat mulai membangun aplikasi Anda sendiri, atau mempelajari lebih dalam tentang konsep dan sumber daya.

Untuk mulai membangun, kami sarankan Anda mencoba salah satu tutorial memulai:

- Ikuti Tutorial: Menghasilkan aplikasi menggunakan AI untuk mempelajari cara menggunakan asisten pembuat AI untuk memulai membangun aplikasi.
- Ikuti Tutorial: Mulai membangun dari aplikasi kosong untuk mempelajari cara membuat aplikasi dari awal sambil mempelajari dasar-dasarnya.

Untuk mempelajari lebih lanjut tentang sumber daya atau konsep yang disebutkan dalam topik ini, lihat topik berikut:

- Connect App Studio ke layanan lain dengan konektor
- Mengonfigurasi model data aplikasi Anda dengan entitas
- Membangun antarmuka pengguna aplikasi Anda dengan halaman dan komponen
- Mendefinisikan dan menerapkan logika bisnis aplikasi Anda dengan otomatisasi
- Mempratinjau, menerbitkan, dan berbagi aplikasi

Menyiapkan dan masuk ke AWS App Studio

Menyiapkan AWS App Studio berbeda tergantung pada peran Anda:

- Penyiapan pertama kali sebagai administrator AWS atau organisasi: Untuk menyiapkan App Studio untuk pertama kalinya sebagai administrator, Anda membuat AWS akun jika tidak memilikinya, membuat instance App Studio, dan mengonfigurasi akses pengguna menggunakan grup Pusat Identitas IAM. Setelah instance dibuat, siapa pun yang memiliki peran Administrator di App Studio dapat melakukan pengaturan lebih lanjut—misalnya, mengonfigurasi konektor untuk menghubungkan layanan lain (seperti sumber data) ke instance App Studio Anda. Untuk informasi tentang penyiapan pertama kali, lihatMembuat dan menyiapkan instans App Studio untuk pertama kalinya.
- Memulai sebagai pembangun: Ketika Anda menerima undangan untuk bergabung dengan App Studio sebagai pembuat, Anda harus menerima undangan dan mengaktifkan kredensional pengguna IAM Identity Center Anda dengan memberikan kata sandi. Setelah itu, Anda dapat masuk ke App Studio dan mulai membangun aplikasi. Untuk informasi tentang menerima undangan dan bergabung dengan instans App Studio, lihatMenerima undangan untuk bergabung dengan App Studio.

Membuat dan menyiapkan instans App Studio untuk pertama kalinya

Mendaftar untuk AWS akun

AWS Akun diperlukan untuk menyiapkan App Studio. Hanya satu AWS akun yang diperlukan untuk menggunakan App Studio—Pembuat dan administrator tidak memerlukan AWS akun untuk menggunakan App Studio karena akses dikelola dengan AWS IAM Identity Center.

Untuk membuat Akun AWS

1. Buka https://portal.aws.amazon.com/billing/pendaftaran.
2. Ikuti petunjuk online.

   Bagian dari prosedur pendaftaran melibatkan tindakan menerima panggilan telepon dan memasukkan kode verifikasi di keypad telepon.

Saat Anda mendaftar untuk sebuah Akun AWS, sebuah Pengguna root akun AWSdibuat. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan tugas yang memerlukan akses pengguna root.

Buat pengguna administratif untuk mengelola AWS sumber daya

Saat pertama kali membuat AWS akun, Anda mulai dengan seperangkat kredensional default dengan akses lengkap ke semua AWS sumber daya di akun Anda. Identitas ini disebut pengguna root AWS akun. Untuk membuat AWS peran dan sumber daya yang akan digunakan dengan App Studio, kami sangat menyarankan agar Anda tidak menggunakan pengguna root AWS akun. Sebagai gantinya, kami menyarankan Anda membuat dan menggunakan pengguna administratif.

Gunakan topik berikut untuk membuat pengguna administratif untuk mengelola AWS peran dan sumber daya untuk digunakan dengan App Studio.

- Untuk satu AWS akun mandiri, lihat Membuat pengguna IAM pertama Anda di Panduan Pengguna IAM. Anda dapat memberikan nama pengguna apa pun, tetapi harus memiliki kebijakan `AdministratorAccess` izin.
- Untuk beberapa AWS akun yang dikelola AWS Organizations, lihat Menyiapkan akses akun AWS untuk pengguna administratif Pusat Identitas IAM di Panduan AWS IAM Identity Center Pengguna.

Membuat instance App Studio di AWS Management Console

Untuk menggunakan App Studio, Anda harus membuat instance dari halaman landing App Studio di halaman AWS Management Console. Ada dua metode yang dapat digunakan untuk membuat instance App Studio:

1. Mudah dibuat: Dengan metode yang disederhanakan ini, Anda hanya menyiapkan satu pengguna untuk mengakses dan menggunakan App Studio sebagai bagian dari pengaturan. Anda harus menggunakan metode ini jika mengevaluasi App Studio untuk organisasi atau tim Anda, atau jika Anda hanya berencana untuk menggunakan App Studio sendiri. Anda dapat menambahkan lebih banyak pengguna atau grup ke App Studio setelah penyiapan. Perhatikan bahwa jika Anda memiliki instance organisasi IAM Identity Center, Anda tidak dapat menggunakan metode ini.
2. Pembuatan standar: Dengan metode ini, Anda menambahkan pengguna atau grup dan menetapkan peran mereka di App Studio sebagai bagian dari pengaturan. Anda harus

menggunakan metode ini jika Anda ingin menambahkan lebih dari satu pengguna ke App Studio saat menyiapkan.

> **Note**
>
> Anda hanya dapat membuat satu instance App Studio, di semua AWS Wilayah. Jika Anda memiliki instance yang sudah ada, Anda harus menghapusnya sebelum membuat yang lain. Untuk informasi selengkapnya, lihat Menghapus instans App Studio.

Easy create

Untuk membuat instance App Studio di AWS Management Console with easy create

1. Buka konsol App Studio di https://console.aws.amazon.com/appstudio/.
2. Arahkan ke AWS Wilayah tempat Anda ingin membuat instance App Studio.
3. Pilih Mulai.
4. Pilih Easy create dan pilih Next.
5. Langkah selanjutnya untuk menyiapkan App Studio ditentukan oleh apakah Anda memiliki instans akun IAM Identity Center atau tidak. Untuk menemukan informasi selengkapnya tentang instans Pusat Identitas IAM, termasuk berbagai jenis dan cara menemukan jenis yang Anda miliki, lihat Mengelola instans organisasi dan akun Pusat Identitas IAM di Panduan Pengguna Pusat Identitas AWS IAM.
   - Jika Anda memiliki instance akun IAM Identity Center:
     a. Di Izin akun, tinjau izin yang diperlukan untuk mengaktifkan App Studio. Jika akun Anda tidak memiliki izin yang diperlukan, Anda tidak akan dapat mengaktifkan App Studio. Anda harus mendapatkan izin yang diperlukan ditambahkan ke akun Anda, atau beralih ke akun yang memilikinya.
     b. Di Tambahkan pengguna, cari dan pilih alamat email pengguna di instans akun Pusat Identitas IAM Anda yang akan mengakses App Studio. Pengguna ini akan memiliki peran Admin dalam instance App Studio. Jika Anda tidak melihat pengguna yang ingin Anda berikan akses ke App Studio, Anda mungkin perlu menambahkannya ke instans Pusat Identitas IAM Anda.
   - Jika Anda tidak memiliki instance akun IAM Identity Center:

> **Note**
>
> Menyiapkan App Studio secara otomatis membuat instans akun Pusat Identitas IAM dengan pengguna yang Anda konfigurasikan selama proses penyiapan. Setelah penyiapan selesai, Anda dapat menambahkan atau mengelola pengguna dan grup di konsol Pusat Identitas IAM di https://console.aws.amazon.com/singlesignon/.

a. Di Izin akun, tinjau izin yang diperlukan untuk mengaktifkan App Studio. Jika akun Anda tidak memiliki izin yang diperlukan, Anda tidak akan dapat mengaktifkan App Studio. Anda harus mendapatkan izin yang diperlukan ditambahkan ke akun Anda, atau beralih ke akun yang memilikinya.

b. Di Tambahkan pengguna, berikan alamat Email, Nama depan, Nama belakang, dan Nama Pengguna untuk pengguna yang mengakses App Studio. Pengguna ini akan memiliki peran Admin dalam instance App Studio.

6. Di CodeCatalyst ruang Create Amazon, berikan nama untuk CodeCatalyst ruang yang akan digunakan App Studio untuk menyimpan kode sumber dan informasi lainnya.
7. Dalam akses dan peran Layanan, tinjau peran layanan dan peran terkait layanan yang dibuat secara otomatis saat Anda menyiapkan App Studio untuk menyediakan layanan dengan izin yang diperlukan. Pilih Lihat izin untuk melihat izin persis yang diberikan untuk peran layanan, atau Lihat kebijakan untuk melihat kebijakan izin yang dilampirkan pada peran terkait layanan.
8. Dalam Pengakuan, akui pernyataan dengan memilih kotak centang mereka.
9. Pilih Siapkan untuk membuat instance Anda.

> **Note**
>
> Untuk menambahkan lebih banyak pengguna atau grup ke instans App Studio setelah penyiapan, Anda harus menambahkannya ke instans Pusat Identitas IAM Anda.

Standard create

Untuk membuat instance App Studio AWS Management Console dengan metode standar

1. Buka konsol App Studio di https://console.aws.amazon.com/appstudio/.
2. Arahkan ke AWS Wilayah tempat Anda ingin membuat instance App Studio.
3. Pilih Mulai.
4. Pilih Standard create dan pilih Next.
5. Langkah-langkah untuk menyiapkan App Studio ditentukan oleh apakah Anda memiliki instans Pusat Identitas IAM atau tidak, dan jenis instans. Untuk menemukan informasi selengkapnya tentang instans Pusat Identitas IAM, termasuk berbagai jenis dan cara menemukan jenis yang Anda miliki, lihat Mengelola instans organisasi dan akun Pusat Identitas IAM di Panduan Pengguna Pusat Identitas AWS IAM.
   - Jika Anda memiliki instance organisasi dari IAM Identity Center:
     a. Di Konfigurasikan akses ke App Studio dengan Single Sign-On, pilih grup Pusat Identitas IAM yang ada untuk memberi mereka akses ke App Studio. Grup App Studio akan dibuat berdasarkan konfigurasi yang ditentukan. Anggota grup yang ditambahkan ke grup Admin akan memiliki peran Admin, dan anggota grup yang ditambahkan ke grup Builder akan memiliki peran Builder di App Studio. Peran didefinisikan sebagai berikut:
        - Admin dapat mengelola pengguna dan grup dalam App Studio, menambah dan mengelola konektor, dan mengelola aplikasi yang dibuat oleh pembangun. Selain itu, pengguna dengan peran Admin memiliki semua izin yang disertakan dengan peran Builder.
        - Pembangun dapat membuat dan membangun aplikasi. Builder tidak dapat mengelola pengguna atau grup, menambah atau mengedit instance konektor, atau mengelola aplikasi pembangun lain.

     b. Di CodeCatalyst ruang Create Amazon, berikan nama untuk CodeCatalyst ruang yang akan digunakan untuk menyimpan kode sumber App Studio dan informasi lainnya.
   - Jika Anda memiliki instance akun instans IAM Identity Center:
     a. Di Izin akun, tinjau izin yang diperlukan untuk mengaktifkan App Studio. Jika akun Anda tidak memiliki izin yang diperlukan, Anda tidak akan dapat mengaktifkan App

Studio. Anda harus mendapatkan izin yang diperlukan ditambahkan ke akun Anda, atau beralih ke akun yang memilikinya.

b. Di Konfigurasikan akses ke App Studio dengan Single Sign-On, di akun IAM Identity Center, pilih Gunakan instance akun yang ada

c. Di AWS Wilayah, pilih Rergion tempat instans akun Pusat Identitas IAM Anda berada.

d. Pilih grup Pusat Identitas IAM yang ada untuk memberi mereka akses ke App Studio. Grup App Studio akan dibuat berdasarkan konfigurasi yang ditentukan. Anggota grup yang ditambahkan ke grup Admin akan memiliki peran Admin, dan anggota grup yang ditambahkan ke grup Builder akan memiliki peran Builder di App Studio. Peran didefinisikan sebagai berikut:

   - Admin dapat mengelola pengguna dan grup dalam App Studio, menambah dan mengelola konektor, dan mengelola aplikasi yang dibuat oleh pembangun. Selain itu, pengguna dengan peran Admin memiliki semua izin yang disertakan dengan peran Builder.
   - Pembangun dapat membuat dan membangun aplikasi. Builder tidak dapat mengelola pengguna atau grup, menambah atau mengedit instance konektor, atau mengelola aplikasi pembangun lain.

- Jika Anda tidak memiliki instans Pusat Identitas IAM:

> **Note**
>
> Menyiapkan App Studio secara otomatis membuat instance akun Pusat Identitas IAM dengan grup yang Anda konfigurasikan selama proses penyiapan. Setelah penyiapan selesai, Anda dapat menambahkan atau mengelola pengguna dan grup di konsol Pusat Identitas IAM di https://console.aws.amazon.com/singlesignon/.

a. Di Izin akun, tinjau izin yang diperlukan untuk mengaktifkan App Studio. Jika akun Anda tidak memiliki izin yang diperlukan, Anda tidak akan dapat mengaktifkan App Studio. Anda harus mendapatkan izin yang diperlukan ditambahkan ke akun Anda, atau beralih ke akun yang memilikinya.

b. Di Konfigurasikan akses ke App Studio dengan Single Sign-On, di akun IAM Identity Center, pilih Buat instance akun untuk saya.

c. Di Buat pengguna dan grup dan tambahkan ke App Studio, berikan nama dan tambahkan pengguna ke grup admin dan grup pembuat. Pengguna yang ditambahkan ke grup admin akan memiliki peran Admin di App Studio, dan pengguna yang ditambahkan ke grup pembuat akan memiliki peran Builder. Peran didefinisikan sebagai berikut:

   - Admin dapat mengelola pengguna dan grup dalam App Studio, menambah dan mengelola konektor, dan mengelola aplikasi yang dibuat oleh pembangun. Selain itu, pengguna dengan peran Admin memiliki semua izin yang disertakan dengan peran Builder.
   - Pembangun dapat membuat dan membangun aplikasi. Builder tidak dapat mengelola pengguna atau grup, menambah atau mengedit instance konektor, atau mengelola aplikasi pembangun lain.

> **Important**
>
> Anda harus menambahkan diri Anda sebagai pengguna grup admin untuk menyiapkan App Studio dan memiliki akses admin setelah pengaturan.

6. Dalam akses dan peran Layanan, tinjau peran layanan dan peran terkait layanan yang dibuat secara otomatis saat Anda menyiapkan App Studio untuk menyediakan layanan dengan izin yang diperlukan. Pilih Lihat izin untuk melihat izin persis yang diberikan untuk peran layanan, atau Lihat kebijakan untuk melihat kebijakan izin yang dilampirkan pada peran terkait layanan.
7. Dalam Pengakuan, akui pernyataan dengan memilih kotak centang mereka.
8. Pilih Siapkan untuk membuat instance Anda.

Menerima undangan untuk bergabung dengan App Studio

Akses ke App Studio dikelola oleh IAM Identity Center. Itu berarti bahwa setiap pengguna yang ingin menggunakan App Studio harus mengonfigurasi pengguna di Pusat Identitas IAM dan termasuk dalam grup yang telah ditambahkan ke App Studio oleh administrator. Ketika administrator mengundang Anda untuk bergabung dengan IAM Identity Center, Anda akan menerima email yang meminta Anda untuk menerima undangan dan mengaktifkan kredensi pengguna Anda. Setelah diaktifkan, Anda dapat menggunakan kredensialnya untuk masuk ke App Studio.

Untuk menerima undangan ke IAM Identity Center untuk mengakses App Studio

1. Saat Anda menerima email undangan, ikuti langkah-langkah untuk memberikan kata sandi dan mengaktifkan kredensi pengguna Anda di Pusat Identitas IAM. Untuk informasi selengkapnya, lihat Menerima undangan untuk bergabung dengan IAM Identity Center.
2. Setelah mengaktifkan kredensi pengguna, gunakan kredensialnya untuk masuk ke instans App Studio.

Memulai dengan AWS App Studio

Tutorial memulai berikut memandu Anda membangun aplikasi pertama Anda di App Studio.

- Direkomendasikan: Untuk menggunakan AI generatif untuk mendeskripsikan aplikasi yang ingin Anda buat, dan secara otomatis membuatnya dan sumber dayanya, lihatTutorial: Menghasilkan aplikasi menggunakan AI.
- Untuk mulai membangun dari aplikasi kosong, lihatTutorial: Mulai membangun dari aplikasi kosong.

Tutorial: Menghasilkan aplikasi menggunakan AI

AWS App Studio berisi fitur AI generatif di seluruh layanan untuk membantu mempercepat pembuatan aplikasi. Dalam tutorial ini, Anda akan mempelajari cara membuat aplikasi menggunakan AI dengan mendeskripsikan aplikasi Anda menggunakan bahasa alami.

Menggunakan AI untuk menghasilkan aplikasi adalah cara yang bagus untuk mulai membangun karena banyak sumber daya aplikasi dibuat untuk Anda. Biasanya jauh lebih mudah untuk mulai membangun dari aplikasi yang dihasilkan dengan sumber daya yang ada daripada memulai dari aplikasi kosong.

> **Note**
>
> Anda dapat melihat posting blog Membangun aplikasi kelas perusahaan dengan bahasa alami menggunakan AWS App Studio (pratinjau) untuk melihat panduan serupa yang menyertakan gambar. Posting blog juga berisi informasi tentang pengaturan dan konfigurasi sumber daya terkait administrator, tetapi Anda dapat melompat ke bagian tentang membangun aplikasi jika diinginkan.

Saat App Studio membuat aplikasi dengan AI, aplikasi akan dibuat dengan sumber daya berikut yang disesuaikan dengan aplikasi yang Anda jelaskan:

- Halaman dan komponen: Komponen adalah blok bangunan antarmuka pengguna aplikasi Anda. Mereka mewakili elemen visual seperti tabel, bentuk, dan tombol. Setiap komponen memiliki kumpulan propertinya sendiri, dan Anda dapat menyesuaikan komponen agar sesuai dengan kebutuhan spesifik Anda. Halaman adalah wadah untuk komponen.

- Otomatisasi: Anda menggunakan otomasi untuk menentukan logika dan alur kerja yang mengatur bagaimana aplikasi Anda berperilaku. Misalnya, Anda dapat menggunakan otomatisasi untuk membuat, memperbarui, membaca, atau menghapus baris dalam tabel data atau berinteraksi dengan objek di bucket Amazon S3. Anda juga dapat menggunakannya untuk menangani tugas-tugas seperti validasi data, notifikasi, atau integrasi dengan sistem lain.
- Entitas: Data adalah informasi yang mendukung aplikasi Anda. Aplikasi yang dihasilkan membuat entitas, yang mirip dengan tabel. Entitas mewakili berbagai jenis data yang perlu Anda simpan dan kerjakan, seperti pelanggan, produk, atau pesanan. Anda dapat menghubungkan model data ini ke berbagai sumber data, termasuk AWS layanan dan eksternal APIs, dengan menggunakan konektor App Studio.

Daftar Isi

Prasyarat

Sebelum Anda memulai, tinjau dan lengkapi prasyarat berikut:

- Akses ke AWS App Studio. Untuk informasi selengkapnya, lihat Menyiapkan dan masuk ke AWS App Studio.
- Opsional: Tinjau AWS Konsep App Studio untuk membiasakan diri dengan konsep App Studio yang penting.

Langkah 1: Buat aplikasi

Langkah pertama dalam membuat aplikasi adalah mendeskripsikan aplikasi yang ingin Anda buat ke asisten AI di App Studio. Anda dapat meninjau aplikasi yang akan dihasilkan, dan mengulangi seperti yang diinginkan sebelum membuatnya.

Untuk menghasilkan aplikasi Anda menggunakan AI

1. Masuk ke App Studio.
2. Di navigasi sebelah kiri, pilih Builder hub dan pilih + Buat aplikasi.
3. Pilih Hasilkan aplikasi dengan AI.
4. Di bidang Nama aplikasi, berikan nama untuk aplikasi Anda.
5. Dalam kotak dialog Pilih sumber data, pilih Lewati.
6. Anda dapat mulai mendefinisikan aplikasi yang ingin Anda hasilkan dengan menjelaskannya di kotak teks, atau dengan memilih Sesuaikan pada prompt sampel. Setelah mendeskripsikan aplikasi, App Studio akan menghasilkan persyaratan dan detail aplikasi untuk Anda tinjau. Ini termasuk kasus penggunaan, alur pengguna, dan model data.
7. Gunakan kotak teks untuk mengulangi aplikasi sesuai kebutuhan hingga Anda puas dengan persyaratan dan detailnya.
8. Saat Anda siap membuat aplikasi dan mulai membangun, pilih Hasilkan aplikasi.
9. Secara opsional, Anda dapat melihat video pendek yang merinci cara menavigasi di sekitar aplikasi baru Anda.
10. Pilih Edit aplikasi untuk masuk ke lingkungan Pengembangan aplikasi Anda.

Langkah 2: Jelajahi aplikasi baru Anda

Di lingkungan Pengembangan, Anda akan menemukan sumber daya berikut:

- Kanvas yang Anda gunakan untuk melihat atau mengedit aplikasi Anda. Kanvas berubah tergantung pada sumber daya yang dipilih.
- Tab navigasi di bagian atas kanvas. Tab dijelaskan dalam daftar berikut:
  - Pages: Tempat Anda menggunakan halaman dan komponen untuk mendesain UI aplikasi Anda.
  - Otomatisasi: Tempat Anda menggunakan tindakan dalam otomatisasi untuk menentukan logika bisnis aplikasi Anda.

  - Data: Tempat Anda menentukan entitas, bidangnya, data sampel, dan tindakan data untuk menentukan model data aplikasi Anda.
  - Pengaturan aplikasi: Tempat Anda menentukan setelan untuk aplikasi, termasuk peran aplikasi, yang Anda gunakan untuk menentukan visibilitas halaman berbasis peran bagi pengguna akhir.
- Menu navigasi sisi kiri, yang berisi sumber daya berdasarkan tab yang Anda lihat.
- Menu sisi kanan yang mencantumkan sumber daya dan properti untuk sumber daya yang dipilih di tab Halaman dan Otomasi.
- Konsol debug yang menampilkan peringatan dan kesalahan di bagian bawah pembuat. Mungkin ada kesalahan di aplikasi yang Anda buat. Ini mungkin karena otomatisasi yang memerlukan konektor yang dikonfigurasi untuk melakukan tindakan, seperti mengirim email dengan Amazon Simple Email Service.
- Jendela obrolan Ask AI untuk mendapatkan bantuan kontekstual dari asisten pembuat AI.

Mari kita lihat lebih dekat tab Pages, Automations, dan Data.

Jelajahi halaman dan komponen

Tab Pages menampilkan halaman dan komponennya yang dibuat untuk Anda.

Setiap halaman mewakili layar antarmuka pengguna (UI) aplikasi Anda yang akan berinteraksi dengan pengguna Anda. Pada halaman ini, Anda dapat menemukan berbagai komponen (seperti tabel, formulir, dan tombol) untuk membuat tata letak dan fungsionalitas yang diinginkan.

Luangkan waktu untuk melihat halaman dan komponennya dengan menggunakan menu navigasi sisi kiri. Ketika Anda memilih halaman atau komponen, Anda dapat memilih Properties di menu sisi kanan.

Jelajahi otomatisasi dan tindakan

Tab Otomasi menunjukkan otomatisasi dan tindakannya yang dibuat untuk Anda.

Otomatisasi menentukan logika bisnis aplikasi Anda, seperti membuat, melihat, memperbarui, atau menghapus entri data, mengirim email, atau bahkan memanggil APIs atau fungsi Lambda.

Luangkan waktu untuk melihat otomatisasi dengan menggunakan menu navigasi sisi kiri. Ketika Anda memilih otomatisasi, Anda dapat melihat propertinya di menu Properties sisi kanan. Otomatisasi berisi sumber daya berikut:

- Otomatisasi terdiri dari tindakan individual, yang merupakan blok bangunan logika bisnis aplikasi Anda. Anda dapat melihat tindakan otomatisasi di menu navigasi sisi kiri, atau di kanvas otomatisasi yang dipilih. Ketika Anda memilih tindakan, Anda dapat melihat propertinya di menu Properties sisi kanan.
- Parameter otomatisasi adalah bagaimana data diteruskan ke otomatisasi. Parameter bertindak sebagai placeholder yang diganti dengan nilai aktual saat otomatisasi dijalankan. Ini memungkinkan Anda untuk menggunakan otomatisasi yang sama dengan input yang berbeda setiap kali.
- Output otomatisasi adalah tempat Anda mengonfigurasi hasil otomatisasi. Secara default, otomatisasi tidak memiliki output, jadi untuk menggunakan hasil otomatisasi dalam komponen atau otomatisasi lainnya, Anda harus mendefinisikannya di sini.

Untuk informasi selengkapnya, lihat Konsep otomatisasi.

Jelajahi data dengan entitas

Tab Data menunjukkan entitas yang dibuat untuk Anda.

Entitas mewakili tabel yang menyimpan data aplikasi Anda, mirip dengan tabel dalam database. Alih-alih menghubungkan antarmuka pengguna (UI) aplikasi Anda dan otomatisasi langsung ke sumber data, mereka terhubung ke entitas terlebih dahulu. Entitas bertindak sebagai perantara antara sumber data aktual Anda dan aplikasi App Studio Anda. Ini menyediakan satu tempat untuk mengelola dan mengakses data Anda.

Luangkan waktu untuk melihat entitas yang dihasilkan dengan memilihnya dari menu navigasi sisi kiri. Anda dapat meninjau detail berikut:

- Tab Konfigurasi menunjukkan nama entitas dan bidangnya, yang mewakili kolom entitas Anda.
- Tab Tindakan data menunjukkan tindakan data yang dihasilkan dengan entitas Anda. Komponen dan otomatisasi dapat menggunakan tindakan data untuk mengambil data dari entitas Anda.
- Tab Data sampel menampilkan data sampel, yang dapat Anda gunakan untuk menguji aplikasi di lingkungan Pengembangan (yang tidak berkomunikasi dengan layanan eksternal). Untuk informasi selengkapnya tentang lingkungan, lihatLingkungan aplikasi.
- Tab Sambungan menampilkan informasi tentang sumber data eksternal yang terhubung dengan entitas. App Studio menyediakan solusi penyimpanan data terkelola yang menggunakan tabel DynamoDB. Untuk informasi selengkapnya, lihat Entitas data terkelola di AWS App Studio.

Langkah 3: Pratinjau aplikasi Anda

Anda dapat melihat pratinjau aplikasi di App Studio untuk melihat tampilannya bagi pengguna. Anda juga dapat menguji fungsinya dengan menggunakannya dan memeriksa log di panel debug.

Lingkungan pratinjau aplikasi tidak mendukung tampilan data langsung atau koneksi dengan sumber daya eksternal dengan konektor, seperti sumber data. Sebagai gantinya, Anda dapat menggunakan data sampel dan output tiruan untuk menguji fungsionalitas.

Untuk melihat pratinjau aplikasi Anda untuk pengujian

1. Di pojok kanan atas pembuat aplikasi, pilih Pratinjau.
2. Berinteraksi dengan halaman aplikasi Anda.

Langkah selanjutnya

Sekarang setelah Anda membuat aplikasi pertama Anda, berikut adalah beberapa langkah selanjutnya:

- Untuk panduan memulai lainnya yang menyertakan gambar, lihat posting blog Membangun aplikasi tingkat perusahaan dengan bahasa alami menggunakan AWS App Studio (pratinjau).
- Aplikasi menggunakan konektor untuk mengirim dan menerima data, atau untuk berkomunikasi dengan layanan eksternal (baik AWS layanan maupun layanan pihak ketiga). Anda perlu mempelajari lebih lanjut tentang konektor dan cara mengonfigurasinya untuk membangun aplikasi. Perhatikan bahwa Anda harus memiliki peran Admin untuk mengelola konektor. Untuk mempelajari selengkapnya, lihat Connect App Studio ke layanan lain dengan konektor.
- Untuk mempelajari lebih lanjut tentang pratinjau, penerbitan, dan akhirnya membagikan aplikasi Anda kepada pengguna akhir, lihatMempratinjau, menerbitkan, dan berbagi aplikasi.
- Terus jelajahi dan perbarui aplikasi yang Anda buat untuk pengalaman langsung.
- Untuk mempelajari lebih lanjut tentang membangun aplikasi, lihatDokumentasi pembangun. Secara khusus, topik-topik berikut mungkin berguna untuk dijelajahi:
  - Referensi tindakan otomatis
  - Referensi komponen
  - Berinteraksi dengan Amazon Simple Storage Service dengan komponen dan otomatisasi
  - Pertimbangan dan mitigasi keamanan

Tutorial: Mulai membangun dari aplikasi kosong

Dalam tutorial ini, Anda akan membangun aplikasi Permintaan Rapat Pelanggan internal menggunakan AWS App Studio. Anda akan belajar tentang cara membuat aplikasi di App Studio, sambil berfokus pada kasus penggunaan dunia nyata dan contoh langsung. Selain itu, Anda akan belajar tentang mendefinisikan struktur data, desain UI, dan penerapan aplikasi.

> **Note**
>
> Tutorial ini merinci cara membuat aplikasi dari awal, dimulai dengan aplikasi kosong. Biasanya, jauh lebih cepat dan lebih mudah untuk menggunakan AI untuk membantu menghasilkan aplikasi dan sumber dayanya dengan memberikan deskripsi aplikasi yang ingin Anda buat. Untuk informasi selengkapnya, lihat Tutorial: Menghasilkan aplikasi menggunakan AI.

Kunci untuk memahami cara membangun aplikasi dengan App Studio adalah memahami empat konsep inti berikut dan bagaimana mereka bekerja sama: komponen, otomatisasi, data, dan konektor.

- Komponen: Komponen adalah blok bangunan antarmuka pengguna aplikasi Anda. Mereka mewakili elemen visual seperti tabel, bentuk, dan tombol. Setiap komponen memiliki serangkaian propertinya sendiri, yang dapat Anda sesuaikan agar sesuai dengan kebutuhan spesifik Anda.
- Otomatisasi: Dengan otomasi, Anda dapat menentukan logika dan alur kerja yang mengatur bagaimana aplikasi Anda berperilaku. Anda dapat menggunakan otomatisasi untuk membuat, memperbarui, membaca, atau menghapus baris dalam tabel data atau berinteraksi dengan objek di bucket Amazon S3. Anda juga dapat menggunakannya untuk menangani tugas-tugas seperti validasi data, notifikasi, atau integrasi dengan sistem lain.
- Data: Data adalah informasi yang mendukung aplikasi Anda. Di App Studio, Anda dapat menentukan model data, yang disebut entitas. Entitas mewakili berbagai jenis data yang perlu Anda simpan dan kerjakan, seperti permintaan pertemuan pelanggan, agenda, atau peserta. Anda dapat menghubungkan model data ini ke berbagai sumber data, termasuk AWS layanan dan eksternal APIs, dengan menggunakan konektor App Studio.
- Konektor: App Studio menyediakan koneksi dengan berbagai sumber data, yang mencakup AWS layanan seperti Aurora, DynamoDB, dan Amazon Redshift. Sumber data juga mencakup layanan pihak ketiga seperti Salesforce, atau banyak lainnya yang menggunakan OpenAPI atau konektor API generik. Anda dapat menggunakan konektor App Studio untuk menggabungkan data dan

fungsionalitas dengan mudah dari layanan kelas perusahaan ini dan aplikasi eksternal ke dalam aplikasi Anda.

Saat Anda maju melalui tutorial, Anda akan mengeksplorasi bagaimana konsep kunci komponen, data, dan otomatisasi bersatu untuk membangun aplikasi Permintaan Pertemuan Pelanggan internal Anda.

Berikut ini adalah langkah-langkah tingkat tinggi yang menjelaskan apa yang akan Anda lakukan dalam tutorial ini:

1. Mulai dengan data: Banyak aplikasi dimulai dengan model data, jadi tutorial ini dimulai dengan data juga. Untuk membangun aplikasi Permintaan Rapat Pelanggan, Anda akan mulai dengan membuat `MeetingRequests` entitas. Entitas ini mewakili struktur data untuk menyimpan semua informasi permintaan pertemuan yang relevan, seperti nama pelanggan, tanggal rapat, agenda, dan peserta. Model data ini berfungsi sebagai fondasi untuk aplikasi Anda, dan memberi daya pada berbagai komponen dan otomatisasi yang akan Anda buat.
2. Buat antarmuka pengguna Anda (UI): Dengan model data di tempat, tutorial kemudian memandu Anda melalui membangun antarmuka pengguna (UI). Di App Studio, Anda membangun UI dengan menambahkan halaman dan menambahkan komponen ke dalamnya. Anda akan menambahkan komponen seperti Tabel, Tampilan detail, dan Kalender ke halaman dasbor permintaan rapat. Komponen-komponen ini akan dirancang untuk menampilkan dan berinteraksi dengan data yang disimpan dalam `MeetingRequests` entitas. Hal ini memungkinkan pengguna Anda untuk melihat, mengelola, dan menjadwalkan pertemuan pelanggan. Anda juga akan membuat halaman pembuatan permintaan rapat. Halaman ini mencakup komponen Formulir untuk mengumpulkan data dan komponen Tombol untuk mengirimkannya.
3. Tambahkan logika bisnis dengan otomasi: Untuk meningkatkan fungsionalitas aplikasi Anda, Anda akan mengonfigurasi beberapa komponen untuk mengaktifkan interaksi pengguna. Beberapa contoh menavigasi ke halaman atau membuat catatan permintaan pertemuan baru di entitas. `MeetingRequests`
4. Tingkatkan dengan validasi dan ekspresi: Untuk memastikan integritas dan keakuratan data Anda, Anda akan menambahkan aturan validasi ke komponen Formulir. Ini akan membantu memastikan bahwa pengguna memberikan informasi yang lengkap dan valid saat membuat catatan permintaan rapat baru. Selain itu, Anda akan menggunakan ekspresi untuk referensi dan memanipulasi data dalam aplikasi Anda sehingga Anda dapat menampilkan informasi dinamis dan kontekstual di seluruh antarmuka pengguna Anda.

5. Pratinjau dan uji: Sebelum menerapkan aplikasi Anda, Anda akan memiliki kesempatan untuk melihat pratinjau dan mengujinya secara menyeluruh. Ini akan memungkinkan Anda untuk memverifikasi bahwa komponen, data, dan otomatisasi semuanya bekerja sama dengan mulus. Ini memberi pengguna Anda pengalaman yang halus dan intuitif.
6. Publikasikan aplikasi: Terakhir, Anda akan menerapkan aplikasi Permintaan Rapat Pelanggan internal yang telah selesai dan membuatnya dapat diakses oleh pengguna Anda. Dengan kekuatan pendekatan low-code di App Studio, Anda akan membangun aplikasi khusus yang memenuhi kebutuhan spesifik organisasi Anda, tanpa perlu keahlian pemrograman yang luas.

Daftar Isi

Prasyarat

Sebelum Anda memulai, tinjau dan lengkapi prasyarat berikut:

- Akses ke AWS App Studio. Untuk informasi selengkapnya, lihat Menyiapkan dan masuk ke AWS App Studio.
- Opsional: Tinjau AWS Konsep App Studio untuk membiasakan diri dengan konsep App Studio yang penting.
- Opsional: Pemahaman tentang konsep pengembangan web dasar, seperti JavaScript sintaks.
- Opsional: Keakraban dengan AWS layanan.

Langkah 1: Buat aplikasi

1. Masuk ke App Studio.
2. Di navigasi sebelah kiri, pilih Builder hub dan pilih + Buat aplikasi.
3. Pilih Mulai dari awal.
4. Di bidang Nama aplikasi, berikan nama untuk aplikasi Anda, seperti**`Customer Meeting Requests`**.
5. Jika diminta untuk memilih sumber data atau konektor, pilih Lewati untuk keperluan tutorial ini.
6. Pilih Berikutnya untuk melanjutkan.
7. (Opsional): Tonton tutorial video untuk ikhtisar singkat tentang membangun aplikasi di App Studio.
8. Pilih Edit aplikasi, yang akan membawa Anda ke pembuat aplikasi App Studio.

Langkah 2: Buat entitas untuk menentukan data aplikasi

Entitas mewakili tabel yang menyimpan data aplikasi Anda, mirip dengan tabel dalam database. Alih-alih antarmuka pengguna (UI) aplikasi Anda dan otomatisasi yang terhubung langsung ke sumber data, mereka terhubung ke entitas terlebih dahulu. Entitas bertindak sebagai perantara antara sumber data aktual Anda dan aplikasi App Studio Anda, dan menyediakan satu tempat untuk mengelola dan mengakses data Anda.

Ada empat cara untuk membuat entitas. Untuk tutorial ini, Anda akan menggunakan entitas terkelola App Studio.

Buat entitas terkelola

Membuat entitas terkelola juga membuat tabel DynamoDB terkait yang dikelola App Studio. Saat perubahan dilakukan pada entitas di app App Studio, tabel DynamoDB akan diperbarui secara otomatis. Dengan opsi ini, Anda tidak perlu membuat, mengelola, atau terhubung secara manual ke sumber data pihak ketiga, atau menetapkan pemetaan dari bidang entitas ke kolom tabel.

Saat membuat entitas, Anda harus menentukan bidang kunci utama. Kunci primer berfungsi sebagai pengidentifikasi unik untuk setiap catatan atau baris dalam entitas. Ini memastikan bahwa setiap catatan dapat dengan mudah diidentifikasi dan diambil tanpa ambiguitas. Kunci primer terdiri dari properti berikut:

- Nama kunci primer: Nama untuk bidang kunci utama entitas.

- Tipe data kunci primer: Jenis bidang kunci utama. Di App Studio, tipe kunci utama yang didukung adalah String untuk teks dan Float untuk angka. Kunci primer teks (seperti*meetingName*) akan memiliki tipe String, dan kunci primer numerik (seperti*meetingId*) akan memiliki tipe Float.

Kunci utama adalah komponen penting dari suatu entitas karena memberlakukan integritas data, mencegah duplikasi data, dan memungkinkan pengambilan dan kueri data yang efisien.

Untuk membuat entitas terkelola

1. Pilih Data dari menu bilah atas.
2. Pilih + Buat entitas.
3. Pilih Buat entitas terkelola App Studio.
4. Di bidang Nama entitas, berikan nama untuk entitas Anda. Untuk tutorial ini, masukkan **`MeetingRequests`**.
5. Di bidang kunci utama, masukkan label nama kunci utama untuk diberikan ke kolom kunci utama di entitas Anda. Untuk tutorial ini, masukkan **`requestID`**.
6. Untuk tipe data kunci Primer, pilih Float.
7. Pilih Buat entitas.

Menambahkan bidang ke entitas Anda

Untuk setiap bidang, Anda akan menentukan nama tampilan, yang merupakan label yang terlihat oleh pengguna aplikasi. Nama tampilan dapat berisi spasi dan karakter khusus, tetapi harus unik di dalam entitas. Nama tampilan berfungsi sebagai label yang mudah digunakan untuk bidang tersebut, dan membantu pengguna dengan mudah mengidentifikasi dan memahami tujuannya.

Selanjutnya, Anda akan memberikan nama sistem, pengenal unik yang digunakan secara internal oleh aplikasi untuk mereferensikan bidang. Nama sistem harus ringkas, tanpa spasi atau karakter khusus. Nama sistem memungkinkan aplikasi untuk membuat perubahan pada data bidang. Ini bertindak sebagai titik referensi unik untuk bidang dalam aplikasi.

Terakhir, Anda akan memilih tipe data yang paling mewakili jenis data yang ingin Anda simpan di bidang, seperti String (text), Boolean (true/false), Date, Decimal, Float, Integer, atau. DateTime Mendefinisikan tipe data yang sesuai memastikan integritas data dan memungkinkan penanganan dan pemrosesan nilai bidang yang tepat. Misalnya, jika Anda menyimpan nama pelanggan dalam permintaan rapat, Anda akan memilih tipe `String` data untuk mengakomodasi nilai teks.

Untuk menambahkan bidang ke **`MeetingRequests`** entitas Anda

- Pilih + Tambahkan bidang untuk menambahkan empat bidang berikut:

  a. Tambahkan bidang yang mewakili nama pelanggan dengan informasi berikut:

     - Nama tampilan: **`Customer name`**
     - Nama sistem: **`customerName`**
     - Tipe data: **`String`**

  b. Tambahkan bidang yang mewakili tanggal pertemuan dengan informasi berikut:

     - Nama tampilan: **`Meeting date`**
     - Nama sistem: **`meetingDate`**
     - Tipe data: **`DateTime`**

  c. Tambahkan bidang yang mewakili agenda rapat dengan informasi berikut:

     - Nama tampilan: **`Agenda`**
     - Nama sistem: **`agenda`**
     - Tipe data: **`String`**

  d. Tambahkan bidang untuk mewakili peserta rapat dengan informasi berikut:

     - Nama tampilan: **`Attendees`**
     - Nama sistem: **`attendees`**
     - Tipe data: **`String`**

Anda dapat menambahkan data sampel ke entitas yang dapat Anda gunakan untuk menguji dan melihat pratinjau aplikasi Anda sebelum mempublikasikannya. Dengan menambahkan hingga 500 baris data tiruan, Anda dapat mensimulasikan skenario dunia nyata dan memeriksa bagaimana aplikasi Anda menangani dan menampilkan berbagai jenis data, tanpa mengandalkan data aktual atau menghubungkan ke layanan eksternal. Ini membantu Anda mengidentifikasi dan menyelesaikan masalah atau ketidakkonsistenan di awal proses pengembangan. Ini memastikan bahwa aplikasi Anda berfungsi sebagaimana dimaksud saat berhadapan dengan data aktual.

Untuk menambahkan data sampel ke entitas Anda

1. Pilih tab Sampel data di spanduk.

2. Pilih Hasilkan lebih banyak data sampel.
3. Pilih Simpan.

Secara opsional, pilih Koneksi di spanduk untuk meninjau detail tentang konektor dan tabel DynamoDB yang dibuat untuk Anda.

Langkah 3: Desain antarmuka pengguna (UI) dan logika

Tambahkan halaman dasbor permintaan rapat

Di App Studio, setiap halaman mewakili layar antarmuka pengguna (UI) aplikasi yang akan berinteraksi dengan pengguna Anda. Dalam halaman ini, Anda dapat menambahkan berbagai komponen seperti tabel, formulir, dan tombol untuk membuat tata letak dan fungsionalitas yang diinginkan.

Aplikasi yang baru dibuat dilengkapi dengan halaman default, jadi Anda akan mengganti namanya alih-alih menambahkan yang baru untuk digunakan sebagai halaman dasbor permintaan rapat sederhana.

Untuk mengganti nama halaman default

1. Di menu navigasi bilah atas, pilih Halaman.
2. Di panel sisi kiri, klik dua kali Page1, ganti nama menjadi, dan tekan Enter. **`MeetingRequestsDashboard`**

Sekarang, tambahkan komponen tabel ke halaman yang akan digunakan untuk menampilkan permintaan rapat.

Untuk menambahkan komponen tabel ke halaman dasbor permintaan rapat

1. Di panel Components sebelah kanan, cari komponen Table dan seret ke kanvas.
2. Pilih tabel di kanvas untuk memilihnya.
3. Di panel Properties sisi kanan, perbarui pengaturan berikut:
    a. Pilih ikon pensil untuk mengganti nama tabel menjadi**`meetingRequestsTable`**.
    b. Di dropdown Source, pilih Entity.
    c. Di menu tarik-turun Tindakan data, pilih entitas yang Anda buat (**`MeetingRequests`**) dan pilih + Tambahkan tindakan data.

4. Jika diminta, pilih GetAll.

> **Note**
>
> Tindakan data getAll adalah jenis tindakan data tertentu yang mengambil semua catatan (baris) dari entitas tertentu. Ketika Anda mengaitkan tindakan data getAll dengan komponen tabel, misalnya, tabel secara otomatis terisi dengan semua data dari entitas yang terhubung, dan menampilkan setiap catatan sebagai baris dalam tabel.

Tambahkan halaman pembuatan permintaan rapat

Selanjutnya, buat halaman yang berisi formulir yang akan digunakan pengguna akhir untuk membuat permintaan rapat. Anda juga akan menambahkan tombol kirim yang membuat catatan di `MeetingRequests` entitas, dan kemudian menavigasi pengguna akhir kembali ke `MeetingRequestsDashboard` halaman.

Untuk menambahkan halaman pembuatan permintaan rapat

1. Di spanduk atas, pilih Pages.
2. Di panel sisi kiri, pilih + Tambah.
3. Di panel properti sisi kanan, pilih ikon pensil dan ganti nama halaman menjadi. **`CreateMeetingRequest`**

Sekarang setelah halaman ditambahkan, Anda akan menambahkan formulir ke halaman yang akan digunakan pengguna akhir untuk memasukkan informasi untuk membuat permintaan pertemuan di `MeetingRequests` entitas. App Studio menawarkan metode untuk menghasilkan formulir dari entitas yang ada, yang mengisi otomatis bidang formulir berdasarkan bidang entitas dan juga menghasilkan tombol kirim untuk membuat catatan di entitas dengan input formulir.

Untuk secara otomatis menghasilkan formulir dari entitas pada halaman pembuatan permintaan pertemuan

1. Pada menu Components sisi kanan, temukan komponen Form dan seret ke kanvas.
2. Pilih Menghasilkan formulir.
3. Dari dropdown, pilih entitas. `MeetingRequests`
4. Pilih Hasilkan.

5. Pilih tombol Kirim pada kanvas untuk memilihnya.
6. Di panel properti sisi kanan, di bagian Pemicu, pilih + Tambah.
7. Pilih Navigasi.
8. Di panel properti sisi kanan, ubah nama Action menjadi sesuatu yang deskriptif, seperti. **`Navigate to MeetingRequestsDashboard`**
9. Ubah jenis Navigasi ke halaman. Di menu tarik-turun Navigasi ke, pilih. **`MeetingRequestsDashboard`**

Sekarang setelah kami memiliki halaman dan formulir pembuatan permintaan rapat, kami ingin membuatnya mudah untuk menavigasi ke halaman ini dari `MeetingRequestsDashboard` halaman, sehingga pengguna akhir yang meninjau dasbor dapat dengan mudah membuat permintaan rapat. Gunakan prosedur berikut untuk membuat tombol pada `MeetingRequestsDashboard` halaman yang menavigasi ke `CreateMeetingRequest` halaman.

Untuk menambahkan tombol untuk menavigasi dari **`MeetingRequestsDashboard`** ke **`CreateMeetingRequest`**

1. Di spanduk atas, pilih Pages.
2. Pilih `MeetingRequestsDashboard` halaman.
3. Di panel Components sisi kanan, temukan komponen Button, seret ke kanvas, dan letakkan di atas tabel.
4. Pilih tombol yang baru ditambahkan untuk memilihnya.
5. Di panel Properties sisi kanan, perbarui pengaturan berikut:

    a. Pilih ikon pensil untuk mengganti nama tombol menjadi**`createMeetingRequestButton`**.
    b. Label tombol:**`Create Meeting Request`**. Ini adalah nama yang akan dilihat pengguna akhir.
    c. Di dropdown Icon, pilih + Plus.
    d. Buat pemicu yang menavigasi pengguna akhir ke `MeetingRequestsDashboard` halaman:

        1. Di bagian Pemicu, pilih + Tambah.
        2. Di Jenis Tindakan, pilih Navigasi.
        3. Pilih pemicu yang baru saja Anda buat untuk mengonfigurasinya.
        4. Dalam nama Action, berikan nama deskriptif seperti**`NavigateToCreateMeetingRequest`**.

5. Di dropdown tipe Navigate, pilih Page.
6. Di menu tarik-turun Navigate to, pilih halaman. `CreateMeetingRequest`

Langkah 4: Pratinjau aplikasi

Anda dapat melihat pratinjau aplikasi di App Studio untuk melihat tampilannya bagi pengguna. Selain itu, Anda dapat menguji fungsinya dengan menggunakannya dan memeriksa log di panel debug.

Lingkungan pratinjau aplikasi tidak mendukung tampilan data langsung. Ini juga tidak mendukung koneksi dengan sumber daya eksternal dengan konektor, seperti sumber data. Sebagai gantinya, Anda dapat menggunakan data sampel dan output tiruan untuk menguji fungsionalitas.

Untuk melihat pratinjau aplikasi Anda untuk pengujian

1. Di pojok kanan atas pembuat aplikasi, pilih Pratinjau.
2. Berinteraksi dengan `MeetingRequestsDashboard` halaman, dan uji tabel, formulir, dan tombol.

Langkah 5: Publikasikan aplikasi ke lingkungan Pengujian

Setelah selesai membuat, mengonfigurasi, dan menguji aplikasi Anda, saatnya mempublikasikannya ke lingkungan Pengujian untuk melakukan pengujian akhir dan kemudian membagikannya dengan pengguna.

Untuk memublikasikan aplikasi Anda ke lingkungan Pengujian

1. Di pojok kanan atas pembuat aplikasi, pilih Publikasikan.
2. Tambahkan deskripsi versi untuk lingkungan Pengujian.
3. Tinjau dan pilih kotak centang mengenai SLA.
4. Pilih Mulai. Penerbitan bisa memakan waktu hingga 15 menit.
5. (Opsional) Saat Anda siap, Anda dapat memberi orang lain akses dengan memilih Bagikan dan mengikuti petunjuknya.

> **Note**
>
> Untuk berbagi aplikasi, Admin harus membuat grup pengguna akhir.

Setelah pengujian, pilih Publikasikan lagi untuk mempromosikan aplikasi ke lingkungan Produksi. Untuk informasi selengkapnya tentang lingkungan aplikasi yang berbeda, lihatLingkungan aplikasi.

Langkah selanjutnya

Sekarang setelah Anda membuat aplikasi pertama Anda, berikut adalah beberapa langkah selanjutnya:

1. Terus membangun aplikasi tutorial. Setelah memiliki data, beberapa halaman, dan otomatisasi yang dikonfigurasi, Anda dapat menambahkan halaman tambahan dan menambahkan komponen untuk mempelajari lebih lanjut tentang membangun aplikasi.
2. Untuk mempelajari lebih lanjut tentang membuat aplikasi, lihatDokumentasi pembangun. Secara khusus, topik-topik berikut mungkin berguna untuk dijelajahi:
   - Referensi tindakan otomatis
   - Referensi komponen
   - Berinteraksi dengan Amazon Simple Storage Service dengan komponen dan otomatisasi
   - Pertimbangan dan mitigasi keamanan

   Selain itu, topik-topik berikut berisi informasi lebih lanjut tentang konsep yang dibahas dalam tutorial:
   - Mempratinjau, menerbitkan, dan berbagi aplikasi
   - Membuat entitas di aplikasi App Studio

Dokumentasi administrator

Topik berikut berisi informasi untuk membantu pengguna yang mengelola koneksi dan akses layanan pihak ketiga, pengguna, dan peran di App Studio.

Topik

Mengelola akses dan peran di App Studio

Salah satu tanggung jawab administrator di App Studio adalah mengelola akses, peran, dan izin. Topik berikut berisi informasi tentang peran di App Studio, dan cara menambahkan pengguna, menghapus pengguna, atau mengubah peran mereka.

Akses ke AWS App Studio dikelola menggunakan grup Pusat Identitas IAM. Untuk menambahkan pengguna ke instance App Studio, Anda harus:

- Tambahkan mereka ke grup Pusat Identitas IAM yang sudah ada yang ditambahkan ke App Studio.
- Tambahkan mereka ke grup Pusat Identitas IAM baru atau yang sudah ada yang tidak ditambahkan ke App Studio, lalu tambahkan ke App Studio.

Karena peran diterapkan ke grup, grup Pusat Identitas IAM harus mewakili hak akses (atau peran) yang ingin Anda tetapkan kepada anggota grup. Untuk informasi selengkapnya tentang Pusat Identitas IAM, termasuk informasi tentang mengelola pengguna dan grup, lihat Panduan Pengguna Pusat Identitas IAM.

Peran dan izin

Ada tiga peran di App Studio. Daftar berikut berisi setiap peran dan deskripsinya.

- Admin: Admin dapat mengelola pengguna dan grup dalam App Studio, menambah dan mengelola konektor, dan mengelola aplikasi yang dibuat oleh pembangun. Selain itu, pengguna dengan peran Admin memiliki semua izin yang disertakan dengan peran Builder.

- Builder: Builder dapat membuat dan membangun aplikasi. Builder tidak dapat mengelola pengguna atau grup, menambah atau mengedit instance konektor, atau mengelola aplikasi pembangun lain.
- Pengguna Aplikasi: Pengguna Aplikasi dapat mengakses dan menggunakan aplikasi yang dipublikasikan, tetapi tidak dapat mengakses instans App Studio Anda untuk membuat aplikasi atau mengelola sumber daya.

Di App Studio, peran ditetapkan ke grup, oleh karena itu setiap anggota grup Pusat Identitas IAM yang ditambahkan akan ditetapkan peran yang ditetapkan ke grup.

Melihat grup

Lakukan langkah-langkah berikut untuk melihat grup yang ditambahkan ke instance App Studio Anda.

> **Note**
>
> Anda harus menjadi Admin untuk melihat grup di instans App Studio Anda.

Untuk melihat grup yang ditambahkan ke instance App Studio

- Di panel navigasi, pilih Peran di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar grup yang ada serta peran yang ditetapkan masing-masing grup.

  Untuk informasi tentang mengelola grup, lihatMenambahkan pengguna atau grup,Mengubah peran grup, atauMenghapus pengguna atau grup dari App Studio.

Menambahkan pengguna atau grup

Untuk menambahkan pengguna ke App Studio, Anda harus menambahkannya ke grup Pusat Identitas IAM dan menambahkan grup tersebut ke App Studio. Lakukan langkah-langkah berikut untuk menambahkan pengguna ke App Studio dengan menambahkan grup Pusat Identitas IAM dan menetapkan peran.

> **Note**
>
> Anda harus menjadi Admin untuk menambahkan pengguna ke instans App Studio Anda.

Untuk menambahkan pengguna atau grup ke instans App Studio

1. Untuk menambahkan pengguna ke instans App Studio, Anda harus menambahkannya ke grup Pusat Identitas IAM yang sudah ada yang telah ditambahkan ke App Studio, atau membuat grup Pusat Identitas IAM baru, menambahkan pengguna baru ke dalamnya, dan menambahkan grup baru ke App Studio.

   Untuk informasi tentang mengelola pengguna dan grup Pusat Identitas IAM, lihat Mengelola identitas di Pusat Identitas IAM di Panduan Pengguna.AWS IAM Identity Center
2. Jika Anda menambahkan pengguna ke grup Pusat Identitas IAM yang sudah ditambahkan ke App Studio, pengguna baru dapat mengakses App Studio dengan izin yang ditentukan setelah menyelesaikan penyiapan izin Pusat Identitas IAM mereka. Jika Anda membuat grup Pusat Identitas IAM baru, lakukan langkah-langkah berikut untuk menambahkan grup ke App Studio dan menetapkan peran untuk anggota grup.
3. Di panel navigasi, pilih Peran di bagian Kelola.
4. Pada halaman Peran, pilih + Tambah grup. Ini akan membuka kotak dialog Tambah grup bagi Anda untuk memasukkan informasi tentang grup.
5. Dalam kotak dialog Tambahkan grup, masukkan informasi berikut:

   - Pilih grup Pusat Identitas IAM yang ada di dropdown.
   - Pilih peran untuk grup.
     - Admin: Admin dapat mengelola pengguna dan grup dalam App Studio, menambah dan mengelola konektor, dan mengelola aplikasi yang dibuat oleh pembangun. Selain itu, pengguna dengan peran Admin memiliki semua izin yang disertakan dengan peran Builder.
     - Builder: Builder dapat membuat dan membangun aplikasi. Builder tidak dapat mengelola pengguna atau grup, menambah atau mengedit instance konektor, atau mengelola aplikasi pembangun lain.
     - Pengguna Aplikasi: Pengguna Aplikasi dapat mengakses dan menggunakan aplikasi yang dipublikasikan, tetapi tidak dapat mengakses instans App Studio Anda untuk membuat aplikasi atau mengelola sumber daya.
6. Pilih Tetapkan untuk menambahkan grup ke App Studio dan berikan peran yang dikonfigurasi kepada anggotanya.

Mengubah peran grup

Ikuti langkah-langkah ini untuk mengubah peran yang ditetapkan ke grup di App Studio. Mengubah peran grup akan mengubah peran setiap anggota dalam grup itu.

> **Note**
>
> Anda harus menjadi Admin untuk mengubah peran grup di App Studio.

Untuk mengubah peran kelompok

1. Di panel navigasi, pilih Peran di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar grup yang ada serta peran yang ditetapkan masing-masing grup.
2. Pilih ikon elips (...) dan pilih Ubah peran.
3. Di kotak dialog Ubah peran, pilih peran baru untuk grup:
   - Administrator: Admin dapat mengelola pengguna dan grup dalam App Studio, menambah dan mengelola konektor, dan mengelola aplikasi yang dibuat oleh pembangun. Selain itu, pengguna dengan peran Admin memiliki semua izin yang disertakan dengan peran Builder.
   - Builder: Builder dapat membuat dan membangun aplikasi. Builder tidak dapat mengelola pengguna atau grup, menambah atau mengedit instance konektor, atau mengelola aplikasi pembangun lain.
   - Pengguna Aplikasi: Pengguna Aplikasi dapat mengakses dan menggunakan aplikasi yang dipublikasikan, tetapi tidak dapat mengakses instans App Studio Anda untuk membuat aplikasi atau mengelola sumber daya.
4. Pilih Ubah ubah peran grup.

Menghapus pengguna atau grup dari App Studio

Anda tidak dapat menghapus grup Pusat Identitas IAM dari App Studio. Melakukan instruksi berikut akan menurunkan peran grup ke Pengguna Aplikasi. Anggota grup masih dapat mengakses aplikasi App Studio yang dipublikasikan.

Untuk menghapus semua akses ke App Studio dan aplikasinya, Anda harus menghapus grup Pusat Identitas IAM atau pengguna di AWS IAM Identity Center konsol. Untuk informasi tentang mengelola

pengguna dan grup Pusat Identitas IAM, lihat Mengelola identitas di Pusat Identitas IAM di Panduan Pengguna.AWS IAM Identity Center

> **Note**
>
> Anda harus menjadi Admin untuk menurunkan akses grup di App Studio.

Untuk menghapus grup

1. Di panel navigasi, pilih Peran di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar grup yang ada serta peran yang ditetapkan masing-masing grup.
2. Pilih ikon elips (...) dan pilih Cabut peran.
3. Di kotak dialog Cabut peran, pilih Cabut untuk menurunkan peran grup ke Pengguna Aplikasi.

Connect App Studio ke layanan lain dengan konektor

Konektor adalah koneksi antara App Studio dan AWS layanan lainnya, seperti AWS Lambda Amazon Redshift, atau layanan pihak ketiga. Setelah konektor dibuat dan dikonfigurasi, builder dapat menggunakannya dan sumber daya yang terhubung ke App Studio dalam aplikasi mereka.

Hanya pengguna dengan peran Admin yang dapat membuat, mengelola, atau menghapus konektor.

Topik

- Connect ke AWS layanan
- Connect ke layanan pihak ketiga
- Melihat, mengedit, dan menghapus konektor

Connect ke AWS layanan

Topik

- Connect ke Amazon Redshift
- Connect ke Amazon DynamoDB
- Connect ke AWS Lambda
- Connect ke Amazon Simple Storage Service (Amazon S3)
- Connect ke Amazon Aurora

- Connect ke Amazon Bedrock
- Connect ke Amazon Simple Email Service
- Connect to AWS services menggunakan konektor Other AWS Services
- Gunakan sumber data terenkripsi dengan CMKs

Connect ke Amazon Redshift

Untuk menghubungkan App Studio dengan Amazon Redshift agar pembuat dapat mengakses dan menggunakan sumber daya Amazon Redshift dalam aplikasi, Anda harus melakukan langkah-langkah berikut:

1. Langkah 1: Buat dan konfigurasikan sumber daya Amazon Redshift
2. Langkah 2: Buat kebijakan dan peran IAM dengan izin Amazon Redshift yang sesuai
3. Langkah 3: Buat konektor Amazon Redshift

Langkah 1: Buat dan konfigurasikan sumber daya Amazon Redshift

Gunakan prosedur berikut untuk membuat dan mengonfigurasi resource Amazon Redshift untuk digunakan dengan App Studio.

Untuk mengatur Amazon Redshift untuk digunakan dengan App Studio

1. Masuk ke AWS Management Console dan buka konsol Amazon Redshift di. https://console.aws.amazon.com/redshiftv2/

   Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Buat gudang data Redshift Tanpa Server atau klaster provisiond. Untuk informasi selengkapnya, lihat Membuat gudang data dengan Redshift Tanpa Server atau Membuat klaster di Panduan Pengguna Amazon Redshift.
3. Setelah penyediaan selesai, pilih Data Kueri untuk membuka editor kueri. Hubungi basis data Anda.
4. Ubah pengaturan berikut:

   1. Setel sakelar sesi Terisolasi ke. `OFF` Ini diperlukan agar Anda dapat melihat perubahan data yang dilakukan oleh pengguna lain, seperti dari aplikasi App Studio yang sedang berjalan.

2. Pilih ikon “roda gigi”. Pilih Pengaturan akun. Tingkatkan Koneksi bersamaan maksimum ke`10`. Ini adalah batas jumlah sesi editor kueri yang dapat terhubung ke database Amazon Redshift. Ini tidak berlaku untuk klien lain seperti aplikasi App Studio.

5. Buat tabel data Anda di bawah `public` skema. `INSERT`data awal apa pun ke dalam tabel ini.

6. Jalankan perintah berikut di editor query:

   Perintah berikut membuat pengguna database dan menghubungkannya dengan peran IAM bernama *`AppBuilderDataAccessRole`* yang digunakan oleh App Studio. Anda akan membuat peran IAM di langkah selanjutnya, dan nama di sini harus cocok dengan nama yang diberikan untuk peran itu.

   ```
   CREATE USER "IAMR:AppBuilderDataAccessRole" WITH PASSWORD DISABLE;
   ```

   Perintah berikut memberikan semua izin pada semua tabel ke App Studio.

   > **Note**
   >
   > Untuk praktik keamanan terbaik, Anda harus memasukkan izin di sini ke izin minimal yang diperlukan pada tabel yang sesuai. Untuk informasi selengkapnya tentang `GRANT` perintah, lihat GRANT di Panduan Pengembang Database Amazon Redshift.

   ```
   GRANT ALL ON ALL TABLES IN SCHEMA public to "IAMR:AppBuilderDataAccessRole";
   ```

Langkah 2: Buat kebijakan dan peran IAM dengan izin Amazon Redshift yang sesuai

Untuk menggunakan resource Amazon Redshift dengan App Studio, administrator harus membuat kebijakan dan peran IAM untuk memberikan izin App Studio untuk mengakses sumber daya. Kebijakan IAM mengontrol ruang lingkup data yang dapat digunakan pembangun dan operasi apa yang dapat dipanggil terhadap data tersebut, seperti Buat, Baca, Perbarui, atau Hapus. Kebijakan IAM kemudian dilampirkan ke peran IAM yang digunakan oleh App Studio.

Sebaiknya buat setidaknya satu peran IAM per layanan dan kebijakan. Misalnya, jika builder membuat dua aplikasi yang didukung oleh tabel berbeda di Amazon Redshift, administrator harus membuat dua kebijakan dan peran IAM, satu untuk setiap tabel di Amazon Redshift.

Langkah 2a: Buat kebijakan IAM dengan izin Amazon Redshift yang sesuai

Kebijakan IAM yang Anda buat dan gunakan dengan App Studio hanya boleh berisi izin minimal yang diperlukan pada sumber daya yang sesuai agar aplikasi dapat mengikuti praktik keamanan terbaik.

Untuk membuat kebijakan IAM dengan izin Amazon Redshift yang sesuai

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat kebijakan IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi sisi kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Di bagian Editor kebijakan, pilih opsi JSON.
5. Ketik atau tempel dalam dokumen kebijakan JSON. Tab berikut berisi contoh kebijakan untuk Amazon Redshift yang disediakan dan tanpa server.

> **Note**
>
> Kebijakan berikut berlaku untuk semua resource Amazon Redshift menggunakan wildcard (). * Untuk praktik keamanan terbaik, Anda harus mengganti wildcard dengan Nama Sumber Daya Amazon (ARN) dari sumber daya yang ingin Anda gunakan dengan App Studio.

Provisioned

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
      "Sid": "ProvisionedRedshiftForAppStudio",
      "Effect": "Allow",
         "Action": [
            "redshift:DescribeClusters",
            "redshift:GetClusterCredentialsWithIAM",
            "redshift-data:ListDatabases",
            "redshift-data:ListTables",
            "redshift-data:DescribeTable",
            "redshift-data:DescribeStatement",
```

```
            "redshift-data:ExecuteStatement",
            "redshift-data:GetStatementResult"
         ],
         "Resource": "*"
      }
   ]
}
```

Serverless

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
      "Sid": "ServerlessRedshiftForAppStudio",
      "Effect": "Allow",
         "Action": [
            "redshift-serverless:ListNamespaces",
            "redshift-serverless:GetCredentials",
            "redshift-serverless:ListWorkgroups",
            "redshift-data:ListDatabases",
            "redshift-data:ListTables",
            "redshift-data:DescribeTable",
            "redshift-data:DescribeStatement",
            "redshift-data:ExecuteStatement",
            "redshift-data:GetStatementResult"
         ],
         "Resource": "*"
      }
   ]
}
```

6. Pilih Berikutnya.
7. Pada halaman Tinjau dan buat, berikan nama Kebijakan, seperti **`RedshiftServerlessForAppStudio`** atau**`RedshiftProvisionedForAppStudio`**, dan Deskripsi (opsional).
8. Pilih Buat kebijakan untuk membuat kebijakan.

Langkah 2b: Buat peran IAM untuk memberi App Studio akses ke sumber daya Amazon Redshift

Sekarang, buat peran IAM yang menggunakan kebijakan yang Anda buat sebelumnya. App Studio akan menggunakan kebijakan ini untuk mendapatkan akses ke resource Amazon Redshift yang dikonfigurasi.

Untuk membuat peran IAM untuk memberi App Studio akses ke resource Amazon Redshift

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat peran IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi sisi kiri, pilih Peran
3. Pilih Buat peran.
4. Di Jenis entitas Tepercaya, pilih Kebijakan kepercayaan khusus.
5. Ganti kebijakan default dengan kebijakan berikut untuk mengizinkan aplikasi App Studio mengambil peran ini di akun Anda.

   Anda harus mengganti placeholder berikut dalam polis. Nilai yang akan digunakan dapat ditemukan di App Studio, di halaman Pengaturan akun.

   - Ganti *111122223333* dengan nomor AWS akun akun yang digunakan untuk menyiapkan instans App Studio, yang terdaftar sebagai ID AWS Akun di pengaturan akun.
   - Ganti *11111111-2222-3333-4444-555555555555* dengan ID tim App Studio Anda, yang dicantumkan sebagai ID Tim di setelan akun di instance App Studio Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
```

```
            }
        }
    ]
}
```

Pilih Berikutnya.

6. Di Tambahkan izin, cari dan pilih kebijakan yang Anda buat di langkah sebelumnya (**`RedshiftServerlessForAppStudio`**atau**`RedshiftProvisionedForAppStudio`**). Memilih + di samping kebijakan akan memperluas kebijakan untuk menampilkan izin yang diberikan olehnya dan memilih kotak centang akan memilih kebijakan tersebut.

   Pilih Berikutnya.

7. Pada halaman Nama, tinjau, dan buat, berikan nama Peran dan Deskripsi.

   > **Important**
   >
   > Nama peran di sini harus cocok dengan nama peran yang digunakan dalam `GRANT` perintah di Langkah 1: Buat dan konfigurasikan sumber daya Amazon Redshift (*`AppBuilderDataAccessRole`*).

8. Pada Langkah 3: Tambahkan tag, pilih Tambahkan tag baru untuk menambahkan tag berikut untuk memberikan akses App Studio:

   - Kunci: `IsAppStudioDataAccessRole`
   - Nilai: `true`

9. Pilih Buat peran dan catat Nama Sumber Daya Amazon (ARN) yang dihasilkan, Anda akan membutuhkannya saat membuat konektor Amazon Redshift di App Studio.

Langkah 3: Buat konektor Amazon Redshift

Setelah sumber daya Amazon Redshift serta kebijakan dan peran IAM dikonfigurasi, gunakan informasi tersebut untuk membuat konektor di App Studio yang dapat digunakan builder untuk menghubungkan aplikasi mereka ke Amazon Redshift.

> **Note**
>
> Anda harus memiliki peran Admin di App Studio untuk membuat konektor.

Untuk membuat konektor untuk Amazon Redshift

1. Arahkan ke App Studio.
2. Di panel navigasi sisi kiri, pilih Konektor di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar konektor yang ada dengan beberapa detail tentang masing-masing.
3. Pilih + Buat konektor.
4. Pilih konektor Amazon Redshift.
5. Konfigurasikan konektor Anda dengan mengisi kolom berikut:
   - Nama: Berikan nama untuk konektor Anda.
   - Deskripsi: Berikan deskripsi untuk konektor Anda.
   - Peran IAM: Masukkan Nama Sumber Daya Amazon (ARN) dari peran IAM yang dibuat di. Langkah 2b: Buat peran IAM untuk memberi App Studio akses ke sumber daya Amazon Redshift Untuk informasi lebih lanjut tentang IAM, lihat Panduan Pengguna IAM.
   - Wilayah: Pilih AWS Wilayah tempat sumber daya Amazon Redshift Anda berada.
   - Jenis komputasi: Pilih apakah Anda menggunakan Amazon Redshift Tanpa Server atau klaster yang disediakan.
   - Pemilihan cluster atau Workgroup: Jika Provisioned dipilih, pilih klaster yang ingin Anda sambungkan ke App Studio. Jika Serverless dipilih, pilih workgroup.
   - Pemilihan database: Pilih database yang ingin Anda sambungkan ke App Studio.
   - Tabel yang tersedia: Pilih tabel yang ingin Anda sambungkan ke App Studio.
6. Pilih Berikutnya. Tinjau informasi koneksi dan pilih Buat.
7. Konektor yang baru dibuat akan muncul di daftar konektor.

Connect ke Amazon DynamoDB

Untuk menghubungkan App Studio dengan DynamoDB agar pembangun dapat mengakses dan menggunakan sumber daya DynamoDB dalam aplikasi, Anda harus melakukan langkah-langkah berikut:

1. Langkah 1: Buat dan konfigurasikan sumber daya DynamoDB
2. Langkah 2: Buat kebijakan dan peran IAM dengan izin DynamoDB yang sesuai
3. Buat konektor DynamoDB

Langkah 1: Buat dan konfigurasikan sumber daya DynamoDB

Gunakan prosedur berikut untuk membuat dan mengonfigurasi sumber daya DynamoDB yang akan digunakan dengan App Studio.

Untuk menyiapkan DynamoDB untuk digunakan dengan App Studio

1. Masuk ke AWS Management Console dan buka konsol DynamoDB di. https://console.aws.amazon.com/dynamodb/

   Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi kiri, pilih Tabel.
3. Pilih Buat tabel.
4. Masukkan nama dan kunci untuk tabel Anda.
5. Pilih Buat tabel.
6. Setelah tabel Anda dibuat, tambahkan beberapa item ke dalamnya sehingga akan muncul setelah tabel terhubung ke App Studio.
   a. Pilih tabel Anda, pilih Tindakan, dan pilih Jelajahi item.
   b. Di Item yang dikembalikan, pilih Buat item.
   c. (Opsional): Pilih Tambahkan atribut baru untuk menambahkan lebih banyak atribut ke tabel Anda.
   d. Masukkan nilai untuk setiap atribut dan pilih Buat item.

Langkah 2: Buat kebijakan dan peran IAM dengan izin DynamoDB yang sesuai

Untuk menggunakan sumber daya DynamoDB dengan App Studio, administrator harus membuat kebijakan dan peran IAM untuk memberikan izin App Studio untuk mengakses sumber daya. Kebijakan IAM mengontrol ruang lingkup data yang dapat digunakan pembangun dan operasi apa yang dapat dipanggil terhadap data tersebut, seperti Buat, Baca, Perbarui, atau Hapus. Kebijakan IAM kemudian dilampirkan ke peran IAM yang digunakan oleh App Studio.

Sebaiknya buat setidaknya satu peran IAM per layanan dan kebijakan. Misalnya, jika pembangun membuat dua aplikasi yang didukung oleh tabel yang sama di DynamoDB, satu yang hanya memerlukan akses baca, dan satu yang memerlukan baca, buat, perbarui, dan hapus; administrator

harus membuat dua peran IAM, satu menggunakan izin baca saja, dan satu dengan izin CRUD penuh ke tabel yang berlaku di DynamoDB.

Langkah 2a: Buat kebijakan IAM dengan izin DynamoDB yang sesuai

Kebijakan IAM yang Anda buat dan gunakan dengan App Studio hanya boleh berisi izin minimal yang diperlukan pada sumber daya yang sesuai agar aplikasi dapat mengikuti praktik keamanan terbaik.

Untuk membuat kebijakan IAM dengan izin DynamoDB yang sesuai

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat kebijakan IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi sisi kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Di bagian Editor kebijakan, pilih opsi JSON.
5. Ketik atau tempel dalam dokumen kebijakan JSON. Tab berikut berisi contoh kebijakan untuk baca saja dan akses penuh ke tabel DynamoDB, bersama dengan contoh kebijakan yang menyertakan AWS KMS izin untuk tabel DynamoDB yang dienkripsi dengan kunci yang dikelola pelanggan (CMK). AWS KMS

   > **Note**
   >
   > Kebijakan berikut berlaku untuk semua resource DynamoDB menggunakan wildcard (). * Untuk praktik keamanan terbaik, Anda harus mengganti wildcard dengan Nama Sumber Daya Amazon (ARN) dari sumber daya yang ingin Anda gunakan dengan App Studio.

   Read only

   Kebijakan berikut memberikan akses baca ke sumber daya DynamoDB yang dikonfigurasi.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
      "Sid": "ReadOnlyDDBForAppStudio",
      "Effect": "Allow",
         "Action": [
```

```
            "dynamodb:ListTables",
            "dynamodb:DescribeTable",
            "dynamodb:PartiQLSelect"
         ],
         "Resource": "*"
      }
   ]
}
```

Full access

Kebijakan berikut memberikan akses buat, baca, perbarui, dan hapus ke sumber daya DynamoDB yang dikonfigurasi.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
      "Sid": "FullAccessDDBForAppStudio",
      "Effect": "Allow",
         "Action": [
           "dynamodb:ListTables",
           "dynamodb:DescribeTable",
           "dynamodb:PartiQLSelect",
           "dynamodb:PartiQLInsert",
           "dynamodb:PartiQLUpdate",
           "dynamodb:PartiQLDelete"
         ],
         "Resource": "*"
      }
   ]
}
```

Read only - KMS encrypted

Kebijakan berikut memberikan akses baca ke sumber daya DynamoDB terenkripsi yang dikonfigurasi dengan memberikan izin. AWS KMS Anda harus mengganti ARN dengan ARN kunci. AWS KMS

```
{
   "Version": "2012-10-17",
   "Statement": [
```

```
      {
      "Sid": "ReadOnlyDDBForAppStudio",
      "Effect": "Allow",
         "Action": [
            "dynamodb:ListTables",
            "dynamodb:DescribeTable",
            "dynamodb:PartiQLSelect"
         ],
         "Resource": "*"
      },
      {
      "Sid": "KMSPermissionsForEncryptedTable",
      "Effect": "Allow",
         "Action": [
            "kms:Decrypt",
            "kms:DescribeKey"
         ],
         "Resource": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      },

   ]
}
```

Full access - KMS encrypted

Kebijakan berikut memberikan akses baca ke sumber daya DynamoDB terenkripsi yang dikonfigurasi dengan memberikan izin. AWS KMS Anda harus mengganti ARN dengan ARN kunci. AWS KMS

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
      "Sid": "ReadOnlyDDBForAppStudio",
      "Effect": "Allow",
         "Action": [
           "dynamodb:ListTables",
           "dynamodb:DescribeTable",
           "dynamodb:PartiQLSelect",
           "dynamodb:PartiQLInsert",
           "dynamodb:PartiQLUpdate",
           "dynamodb:PartiQLDelete"
```

```
         ],
         "Resource": "*"
      },
      {
      "Sid": "KMSPermissionsForEncryptedTable",
      "Effect": "Allow",
         "Action": [
            "kms:Decrypt",
            "kms:DescribeKey"
         ],
         "Resource": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
      },

   ]
}
```

6. Pilih Berikutnya.
7. Pada halaman Tinjau dan buat, berikan nama Kebijakan, seperti **`ReadOnlyDDBForAppStudio`** atau**`FullAccessDDBForAppStudio`**, dan Deskripsi (opsional).
8. Pilih Buat kebijakan untuk membuat kebijakan.

Langkah 2b: Buat peran IAM untuk memberikan akses App Studio ke sumber daya DynamoDB

Sekarang, buat peran IAM yang menggunakan kebijakan yang Anda buat sebelumnya. App Studio akan menggunakan kebijakan ini untuk mendapatkan akses ke sumber daya DynamoDB yang dikonfigurasi.

Untuk membuat peran IAM untuk memberikan akses App Studio ke sumber daya DynamoDB

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat peran IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi konsol, pilih Peran dan kemudian pilih Buat peran.
3. Di Jenis entitas Tepercaya, pilih Kebijakan kepercayaan khusus.
4. Ganti kebijakan default dengan kebijakan berikut untuk mengizinkan aplikasi App Studio mengambil peran ini di akun Anda.

   Anda harus mengganti placeholder berikut dalam polis. Nilai yang akan digunakan dapat ditemukan di App Studio, di halaman Pengaturan akun.

- Ganti *111122223333* dengan nomor AWS akun akun yang digunakan untuk menyiapkan instans App Studio, yang terdaftar sebagai ID AWS Akun di pengaturan akun.
- Ganti *11111111-2222-3333-4444-555555555555* dengan ID tim App Studio Anda, yang dicantumkan sebagai ID Tim di setelan akun di instance App Studio Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

Pilih Berikutnya.

5. Di Tambahkan izin, cari dan pilih kebijakan yang Anda buat di langkah sebelumnya (**`ReadOnlyDDBForAppStudio`**atau**`FullAccessDDBForAppStudio`**). Memilih + di samping kebijakan akan memperluas kebijakan untuk menampilkan izin yang diberikan olehnya dan memilih kotak centang akan memilih kebijakan tersebut.

   Pilih Berikutnya.

6. Pada halaman Nama, tinjau, dan buat, berikan nama Peran dan Deskripsi.
7. Pada Langkah 3: Tambahkan tag, pilih Tambahkan tag baru untuk menambahkan tag berikut untuk memberikan akses App Studio:

   - Kunci: `IsAppStudioDataAccessRole`
   - Nilai: `true`

8. Pilih Buat peran dan catat Nama Sumber Daya Amazon (ARN) yang dihasilkan, Anda akan memerlukannya saat membuat konektor DynamoDB di App Studio.

Buat konektor DynamoDB

Setelah sumber daya DynamoDB serta kebijakan dan peran IAM Anda dikonfigurasi, gunakan informasi tersebut untuk membuat konektor di App Studio yang dapat digunakan builder untuk menghubungkan aplikasi mereka ke DynamoDB.

> **Note**
>
> Anda harus memiliki peran Admin di App Studio untuk membuat konektor.

Untuk membuat konektor untuk DynamoDB

1. Arahkan ke App Studio.
2. Di panel navigasi sisi kiri, pilih Konektor di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar konektor yang ada dengan beberapa detail tentang masing-masing.
3. Pilih + Buat konektor.
4. Pilih Amazon DynamoDB dari daftar jenis konektor.
5. Konfigurasikan konektor Anda dengan mengisi kolom berikut:
   - Nama: Masukkan nama untuk konektor DynamoDB Anda.
   - Deskripsi: Masukkan deskripsi untuk konektor DynamoDB Anda.
   - Peran IAM: Masukkan Nama Sumber Daya Amazon (ARN) dari peran IAM yang dibuat di. Langkah 2b: Buat peran IAM untuk memberikan akses App Studio ke sumber daya DynamoDB Untuk informasi lebih lanjut tentang IAM, lihat Panduan Pengguna IAM.
   - Wilayah: Pilih AWS Wilayah tempat sumber daya DynamoDB Anda berada.
   - Tabel yang tersedia: Pilih tabel yang ingin Anda sambungkan ke App Studio.
6. Pilih Berikutnya. Tinjau informasi koneksi dan pilih Buat.
7. Konektor yang baru dibuat akan muncul di daftar Konektor.

Connect ke AWS Lambda

Untuk menghubungkan App Studio dengan Lambda agar pembuat dapat mengakses dan menggunakan sumber daya Lambda dalam aplikasi, Anda harus melakukan langkah-langkah berikut:

1. Langkah 1: Buat dan konfigurasikan fungsi Lambda
2. Langkah 2: Buat peran IAM untuk memberi App Studio akses ke sumber daya Lambda
3. Langkah 3: Buat konektor Lambda

Langkah 1: Buat dan konfigurasikan fungsi Lambda

Jika Anda tidak memiliki fungsi Lambda yang ada, Anda harus membuatnya terlebih dahulu. Untuk mempelajari selengkapnya tentang membuat fungsi Lambda, lihat Panduan AWS Lambda Pengembang.

Langkah 2: Buat peran IAM untuk memberi App Studio akses ke sumber daya Lambda

Untuk menggunakan resource Lambda dengan App Studio, administrator harus membuat peran IAM untuk memberikan izin App Studio untuk mengakses sumber daya. Peran IAM mengontrol sumber daya atau operasi yang dapat diakses aplikasi dari Lambda.

Sebaiknya buat setidaknya satu peran IAM per layanan dan kebijakan.

Untuk membuat peran IAM untuk memberikan akses App Studio ke sumber daya Lambda

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat peran IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi konsol, pilih Peran dan kemudian pilih Buat peran.
3. Di Jenis entitas Tepercaya, pilih Kebijakan kepercayaan khusus.
4. Ganti kebijakan default dengan kebijakan berikut untuk mengizinkan aplikasi App Studio mengambil peran ini di akun Anda.

   Anda harus mengganti placeholder berikut dalam polis. Nilai yang akan digunakan dapat ditemukan di App Studio, di halaman Pengaturan akun.

   - Ganti *111122223333* dengan nomor AWS akun akun yang digunakan untuk menyiapkan instans App Studio, yang terdaftar sebagai ID AWS Akun di pengaturan akun.

- Ganti *11111111-2222-3333-4444-555555555555* dengan ID tim App Studio Anda, yang dicantumkan sebagai ID Tim di setelan akun di instance App Studio Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

Pilih Berikutnya.

5. Di Tambahkan izin, cari dan pilih kebijakan yang memberikan izin yang sesuai untuk peran tersebut. Memilih + di samping kebijakan akan memperluas kebijakan untuk menampilkan izin yang diberikan olehnya dan memilih kotak centang akan memilih kebijakan. Untuk Lambda, Anda dapat mempertimbangkan untuk menambahkan `AWSLambdaRole` kebijakan, yang memberikan izin untuk menjalankan fungsi Lambda.

   Untuk informasi selengkapnya tentang penggunaan kebijakan IAM dengan Lambda, termasuk daftar kebijakan terkelola dan deskripsinya, lihat Identity and Access Management AWS Lambda untuk di AWS Lambda Panduan Pengembang.

   Pilih Berikutnya.

6. Pada halaman Nama, tinjau, dan buat, berikan nama Peran dan Deskripsi.
7. Pada Langkah 3: Tambahkan tag, pilih Tambahkan tag baru untuk menambahkan tag berikut untuk memberikan akses App Studio:

   - Kunci: `IsAppStudioDataAccessRole`

- Nilai: `true`

8. Pilih Buat peran dan catat Nama Sumber Daya Amazon (ARN) yang dihasilkan, Anda akan membutuhkannya saat membuat konektor Lambda di App Studio.

Langkah 3: Buat konektor Lambda

Setelah sumber daya Lambda serta kebijakan dan peran IAM Anda dikonfigurasi, gunakan informasi tersebut untuk membuat konektor di App Studio yang dapat digunakan builder untuk menghubungkan aplikasi mereka ke Lambda.

> **Note**
>
> Anda harus memiliki peran Admin di App Studio untuk membuat konektor.

Untuk membuat konektor untuk Lambda

1. Arahkan ke App Studio.
2. Di panel navigasi sisi kiri, pilih Konektor di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar konektor yang ada dengan beberapa detail tentang masing-masing.
3. Pilih + Buat konektor.
4. Pilih AWS Layanan Lain dari daftar jenis konektor.
5. Konfigurasikan konektor Anda dengan mengisi kolom berikut:
   - Nama: Masukkan nama untuk konektor Lambda Anda.
   - Deskripsi: Masukkan deskripsi untuk konektor Lambda Anda.
   - Peran IAM: Masukkan Nama Sumber Daya Amazon (ARN) dari peran IAM yang dibuat di. Langkah 2: Buat peran IAM untuk memberi App Studio akses ke sumber daya Lambda Untuk informasi lebih lanjut tentang IAM, lihat Panduan Pengguna IAM.
   - Layanan: Pilih Lambda.
   - Wilayah: Pilih AWS Wilayah tempat sumber daya Lambda Anda berada.
6. Pilih Buat.
7. Konektor yang baru dibuat akan muncul di daftar Konektor.

Connect ke Amazon Simple Storage Service (Amazon S3)

Untuk menghubungkan App Studio dengan Amazon S3 agar pembuat dapat mengakses dan menggunakan sumber daya Amazon S3 dalam aplikasi, lakukan langkah-langkah berikut:

1. Langkah 1: Buat dan konfigurasikan sumber daya Amazon S3
2. Langkah 2: Buat kebijakan dan peran IAM dengan izin Amazon S3 yang sesuai
3. Langkah 3: Buat konektor Amazon S3

Setelah Anda menyelesaikan langkah-langkah dan membuat konektor dengan izin yang tepat, pembangun dapat menggunakan konektor untuk membuat aplikasi yang berinteraksi dengan sumber daya Amazon S3. Untuk informasi selengkapnya tentang berinteraksi dengan Amazon S3 di aplikasi App Studio, lihat. Berinteraksi dengan Amazon Simple Storage Service dengan komponen dan otomatisasi

Langkah 1: Buat dan konfigurasikan sumber daya Amazon S3

Bergantung pada kebutuhan aplikasi dan sumber daya yang ada, Anda mungkin perlu membuat bucket Amazon S3 agar aplikasi dapat menulis dan membaca. Untuk informasi tentang membuat sumber daya Amazon S3, termasuk bucket, lihat Memulai Amazon S3 di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Untuk menggunakan Unggah S3 komponen di aplikasi, Anda harus menambahkan konfigurasi cross-origin resource sharing (CORS) ke bucket Amazon S3 yang ingin Anda unggah. Konfigurasi CORS memberikan izin App Studio untuk mendorong objek ke bucket. Prosedur berikut merinci cara menambahkan konfigurasi CORS ke bucket Amazon S3 menggunakan konsol. Untuk informasi selengkapnya tentang CORS dan mengonfigurasinya, lihat Menggunakan berbagi sumber daya lintas asal (CORS) di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Untuk menambahkan konfigurasi CORS ke bucket Amazon S3 di konsol

1. Arahkan ke ember Anda di https://console.aws.amazon.com/s3/.
2. Pilih tab Izin.
3. Di Berbagi sumber daya lintas asal (CORS), pilih Edit.
4. Tambahkan cuplikan berikut:

```
[
```

```
    {
        "AllowedHeaders": [
            "*"
        ],
        "AllowedMethods": [
            "PUT",
            "POST"
        ],
        "AllowedOrigins": [
            "*"
        ],
        "ExposeHeaders": []
    }
]
```

5. Pilih Simpan perubahan.

Langkah 2: Buat kebijakan dan peran IAM dengan izin Amazon S3 yang sesuai

Untuk menggunakan sumber daya Amazon S3 dengan App Studio, administrator harus membuat kebijakan dan peran IAM untuk memberikan izin App Studio untuk mengakses sumber daya. Kebijakan IAM mengontrol ruang lingkup data yang dapat digunakan pembangun dan operasi apa yang dapat dipanggil terhadap data tersebut, seperti Buat, Baca, Perbarui, atau Hapus. Kebijakan IAM kemudian dilampirkan ke peran IAM yang digunakan oleh App Studio.

Sebaiknya buat setidaknya satu peran IAM per layanan dan kebijakan. Misalnya, jika builder membuat dua aplikasi yang didukung oleh bucket berbeda di Amazon S3, administrator harus membuat dua kebijakan dan peran IAM, satu untuk setiap bucket.

Langkah 2a: Buat kebijakan IAM dengan izin Amazon S3 yang sesuai

Kebijakan IAM yang Anda buat dan gunakan dengan App Studio hanya boleh berisi izin minimal yang diperlukan pada sumber daya yang sesuai agar aplikasi dapat mengikuti praktik keamanan terbaik.

Untuk membuat kebijakan IAM dengan izin Amazon S3 yang sesuai

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat kebijakan IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi sisi kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.

4. Di bagian Editor kebijakan, pilih opsi JSON.
5. Ketik atau tempel dalam dokumen kebijakan JSON. Tab berikut berisi contoh kebijakan untuk hanya baca dan akses penuh ke sumber daya Amazon S3.

> **Note**
>
> Kebijakan berikut berlaku untuk semua resource Amazon S3 menggunakan wildcard (). * Untuk praktik keamanan terbaik, Anda harus mengganti wildcard dengan Nama Sumber Daya Amazon (ARN) sumber daya, seperti bucket atau folder, yang ingin Anda gunakan dengan App Studio.

Read only

Kebijakan berikut memberikan akses hanya baca (dapatkan dan daftar) ke bucket atau folder Amazon S3 yang dikonfigurasi.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "S3ReadOnlyForAppStudio",
         "Effect": "Allow",
         "Action": [
            "s3:GetObject",
            "s3:ListBucket"
         ],
         "Resource": "*"
      }
   ]
}
```

Full access

Kebijakan berikut memberikan akses penuh (put, get, list, dan delete) ke bucket atau folder Amazon S3 yang dikonfigurasi.

```
{
   "Version": "2012-10-17",
   "Statement": [
```

```
      {
         "Sid": "S3FullAccessForAppStudio",
         "Effect": "Allow",
         "Action": [
            "s3:PutObject",
            "s3:GetObject",
            "s3:ListBucket",
            "s3:DeleteObject"
         ],
         "Resource": "*"
      }
   ]
}
```

6. Pilih Berikutnya.
7. Pada halaman Tinjau dan buat, berikan nama Kebijakan, seperti**`AWSAppStudioS3FullAccess`**, dan Deskripsi (opsional).
8. Pilih Buat kebijakan untuk membuat kebijakan.

Langkah 2b: Buat peran IAM untuk memberi App Studio akses ke sumber daya Amazon S3

Untuk menggunakan sumber daya Amazon S3 dengan App Studio, administrator harus membuat peran IAM untuk memberikan izin App Studio untuk mengakses sumber daya. Peran IAM mengontrol ruang lingkup data yang dapat digunakan pembangun dan operasi apa yang dapat dipanggil terhadap data tersebut, seperti Buat, Baca, Perbarui, atau Hapus.

Sebaiknya buat setidaknya satu peran IAM per layanan dan kebijakan.

Untuk membuat peran IAM untuk memberikan akses App Studio ke sumber daya Amazon S3

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat peran IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi konsol, pilih Peran dan kemudian pilih Buat peran.
3. Di Jenis entitas Tepercaya, pilih Kebijakan kepercayaan khusus.
4. Ganti kebijakan default dengan kebijakan berikut untuk mengizinkan aplikasi App Studio mengambil peran ini di akun Anda.

   Anda harus mengganti placeholder berikut dalam polis. Nilai yang akan digunakan dapat ditemukan di App Studio, di halaman Pengaturan akun.

- Ganti *111122223333* dengan nomor AWS akun akun yang digunakan untuk menyiapkan instans App Studio, yang terdaftar sebagai ID AWS Akun di pengaturan akun.
- Ganti *11111111-2222-3333-4444-555555555555* dengan ID tim App Studio Anda, yang dicantumkan sebagai ID Tim di setelan akun di instance App Studio Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

Pilih Berikutnya.

5. Di Tambahkan izin, cari dan pilih kebijakan yang Anda buat di langkah sebelumnya (**`S3ReadOnlyForAppStudio`**atau**`S3FullAccessForAppStudio`**). Memilih + di samping kebijakan akan memperluas kebijakan untuk menampilkan izin yang diberikan olehnya dan memilih kotak centang akan memilih kebijakan.

   Pilih Berikutnya.

6. Pada halaman Nama, tinjau, dan buat, berikan nama Peran dan Deskripsi.
7. Pada Langkah 3: Tambahkan tag, pilih Tambahkan tag baru untuk menambahkan tag berikut untuk memberikan akses App Studio:

   - Kunci: `IsAppStudioDataAccessRole`
   - Nilai: `true`

8. Pilih Buat peran dan catat Nama Sumber Daya Amazon (ARN) yang dihasilkan, Anda akan memerlukannya untuk membuat konektor Amazon S3 di App Studio pada langkah berikutnya.

Langkah 3: Buat konektor Amazon S3

Setelah sumber daya Amazon S3 serta kebijakan dan peran IAM Anda dikonfigurasi, gunakan informasi tersebut untuk membuat konektor di App Studio yang dapat digunakan builder untuk menghubungkan aplikasi mereka ke Amazon S3.

> **Note**
>
> Anda harus memiliki peran Admin di App Studio untuk membuat konektor.

Untuk membuat konektor untuk Amazon S3

1. Arahkan ke App Studio.
2. Di panel navigasi sisi kiri, pilih Konektor di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar konektor yang ada dengan beberapa detail tentang masing-masing.
3. Pilih + Buat konektor.
4. Pilih konektor Amazon S3.
5. Konfigurasikan konektor Anda dengan mengisi kolom berikut:
   - Nama: Masukkan nama untuk konektor Amazon S3 Anda.
   - Deskripsi: Masukkan deskripsi untuk konektor Amazon S3 Anda.
   - Peran IAM: Masukkan Nama Sumber Daya Amazon (ARN) dari peran IAM yang dibuat di. Langkah 2b: Buat peran IAM untuk memberi App Studio akses ke sumber daya Amazon S3 Untuk informasi lebih lanjut tentang IAM, lihat Panduan Pengguna IAM.
   - Wilayah: Pilih AWS Wilayah tempat sumber daya Amazon S3 Anda berada.
6. Pilih Buat.
7. Konektor yang baru dibuat akan muncul di daftar Konektor.

Connect ke Amazon Aurora

Untuk menghubungkan App Studio dengan Aurora agar pembangun dapat mengakses dan menggunakan sumber daya Aurora dalam aplikasi, Anda harus melakukan langkah-langkah berikut:

1. Langkah 1: Buat dan konfigurasikan sumber daya Aurora
2. Langkah 2: Buat kebijakan dan peran IAM dengan izin Aurora yang sesuai
3. Langkah 3: Buat konektor Aurora di App Studio

App Studio mendukung versi Aurora berikut:

- Aurora MySQL Tanpa Server V1:5.72
- Aurora PostgreSQL Tanpa Server V1:11.18, 13.9
- Aurora MySQL Serverless V2:13.11 atau lebih tinggi, 14.8 atau lebih tinggi, dan 15.3 atau lebih tinggi
- Aurora PostgreSQL Serverless V2:13.11 atau lebih tinggi, 14.8 atau lebih tinggi, dan 15.3 atau lebih tinggi

Langkah 1: Buat dan konfigurasikan sumber daya Aurora

Untuk menggunakan database Aurora dengan App Studio, Anda harus terlebih dahulu membuatnya dan mengonfigurasinya dengan tepat. Ada dua jenis database Aurora yang didukung oleh App Studio: Aurora PostgreSQL dan Aurora MySQL. Untuk membandingkan jenisnya, lihat Apa perbedaan antara MySQL dan PostgreSQL? . Pilih tab yang sesuai dan ikuti prosedur untuk menyiapkan Aurora untuk digunakan dengan aplikasi App Studio.

Aurora PostgreSQL

Gunakan prosedur berikut untuk membuat dan mengonfigurasi klaster database PostgreSQL Aurora untuk digunakan dengan App Studio.

Untuk mengatur Aurora untuk digunakan dengan App Studio

1. Masuk ke AWS Management Console dan buka konsol Amazon RDS di https://console.aws.amazon.com/rds/.
2. Pilih Buat basis data.
3. Pilih Aurora (PostgreSQL Kompatibel).
4. Dalam versi yang tersedia, pilih versi yang lebih besar dari atau sama dengan versi`13.11`,`14.8`, dan`15.3`.
5. Di Pengaturan, masukkan pengidentifikasi cluster DB.
6. Dalam konfigurasi Instans, pilih Serverless v2 dan pilih kapasitas yang sesuai.

7. Di Konektivitas, pilih Aktifkan API Data RDS.
8. Dalam otentikasi Database, pilih autentikasi database IAM.
9. Dalam konfigurasi tambahan, dalam nama database awal, masukkan nama database awal untuk database Anda.

Aurora MySQL

Gunakan prosedur berikut untuk membuat dan mengonfigurasi cluster database MySQL Aurora untuk digunakan dengan App Studio.

Aurora MySQL tidak mendukung pembuatan dari UI untuk versi yang mendukung Data API atau Serverless v1. Untuk membuat cluster MySQL Aurora yang mendukung API Data, Anda harus menggunakan file. AWS CLI

> **Note**
>
> Untuk menggunakan database Aurora MySQL dengan App Studio, mereka harus berada di cloud pribadi virtual (VPC). Untuk informasi selengkapnya, lihat Bekerja dengan cluster DB di VPC di Panduan Pengguna Amazon Aurora.

Untuk mengatur Aurora MySQL untuk digunakan dengan App Studio

1. Jika perlu, instal AWS CLI dengan mengikuti petunjuk di Instal atau perbarui ke versi terbaru AWS CLI di Panduan AWS Command Line Interface Pengguna.
2. Masuk ke AWS Management Console dan buka konsol Amazon RDS di https://console.aws.amazon.com/rds/.
3. Di navigasi sisi kiri, pilih grup Subnet.
4. Pilih Buat grup subnet DB.
5. Isi informasi dan buat grup sunbnet. Untuk informasi selengkapnya tentang grup subnet dan penggunaan VPCs, lihat Bekerja dengan klaster DB di VPC di Panduan Pengguna Amazon Aurora.
6. Jalankan AWS CLI perintah berikut:

```
aws rds create-db-cluster --database-name db_name \
    --db-cluster-identifier db_cluster_identifier \
    --engine aurora-mysql \
```

```
    --engine-version 5.7.mysql_aurora.2.08.3 \
    --engine-mode serverless \
    --scaling-configuration
MinCapacity=4,MaxCapacity=32,SecondsUntilAutoPause=1000,AutoPause=true \
    --master-username userName \
    --master-user-password userPass \
    --availability-zones us-west-2b us-west-2c \
    --db-subnet-group-name subnet-group-name
```

Ganti bidang berikut:

- Ganti *db_name* dengan nama database yang diinginkan.
- Ganti *db_cluster_identifier* dengan pengidentifikasi cluster database yang diinginkan.
- (Opsional) Ganti angka di `scaling-configuration` bidang sesuai keinginan.
- Ganti *userName* dengan nama pengguna yang diinginkan.
- Ganti *userPass* dengan kata sandi yang diinginkan.
- Di`availability-zones`, tambahkan zona ketersediaan dari grup subnet yang Anda buat.
- Ganti *subnet-group-name* dengan nama grup subnet yang Anda buat.

Langkah 2: Buat kebijakan dan peran IAM dengan izin Aurora yang sesuai

Untuk menggunakan sumber daya Aurora dengan App Studio, administrator harus membuat kebijakan IAM dan melampirkannya ke peran IAM yang digunakan untuk memberikan izin App Studio untuk mengakses sumber daya yang dikonfigurasi. Kebijakan dan peran IAM mengontrol ruang lingkup data yang dapat digunakan pembangun dan operasi apa yang dapat dipanggil terhadap data tersebut, seperti Buat, Baca, Perbarui, atau Hapus.

Sebaiknya buat setidaknya satu peran IAM per layanan dan kebijakan.

Langkah 2a: Buat kebijakan IAM dengan izin Aurora yang sesuai

Kebijakan IAM yang Anda buat dan gunakan dengan App Studio hanya boleh berisi izin minimal yang diperlukan pada sumber daya yang sesuai agar aplikasi dapat mengikuti praktik keamanan terbaik.

Untuk membuat kebijakan IAM dengan izin Aurora yang sesuai

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat peran IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi sisi kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Di bagian Editor kebijakan, pilih opsi JSON.
5. Ganti cuplikan yang ada dengan cuplikan berikut, ganti *111122223333* dengan nomor AWS akun tempat sumber daya Amazon Redshift dan Aurora terkandung.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "BaselineAuroraForAppStudio",
            "Effect": "Allow",
            "Action": [
                "rds-data:ExecuteStatement",
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws:rds:*:111122223333:cluster:*",
                "arn:aws:secretsmanager:*:111122223333:secret:rds*"
            ]
        }
    ]
}
```

6. Pilih Berikutnya.
7. Pada halaman Tinjau dan buat, berikan nama Kebijakan, seperti **`Aurora_AppStudio`** dan Deskripsi (opsional).
8. Pilih Buat kebijakan untuk membuat kebijakan.

Langkah 2b: Buat peran IAM untuk memberi App Studio akses ke sumber daya Aurora

Sekarang, buat peran IAM yang menggunakan kebijakan yang Anda buat sebelumnya. App Studio akan menggunakan kebijakan ini untuk mendapatkan akses ke sumber daya Aurora yang dikonfigurasi.

Untuk membuat peran IAM untuk memberikan akses App Studio ke sumber daya Aurora

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat peran IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi konsol, pilih Peran dan kemudian pilih Buat peran.
3. Di Jenis entitas Tepercaya, pilih Kebijakan kepercayaan khusus.
4. Ganti kebijakan default dengan kebijakan berikut untuk mengizinkan aplikasi App Studio mengambil peran ini di akun Anda.

   Anda harus mengganti placeholder berikut dalam polis. Nilai yang akan digunakan dapat ditemukan di App Studio, di halaman Pengaturan akun.

   - Ganti *111122223333* dengan nomor AWS akun akun yang digunakan untuk mengatur instance App Studio, yang terdaftar sebagai ID AWS Akun di pengaturan akun.
   - Ganti *11111111-2222-3333-4444-555555555555* dengan ID tim App Studio Anda, yang dicantumkan sebagai ID Tim di setelan akun di instance App Studio Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

   Pilih Berikutnya.

5. Di Tambahkan izin, cari dan pilih kebijakan yang Anda buat sebelumnya (**`Aurora_AppStudio`**). Memilih + di samping kebijakan akan memperluas kebijakan untuk menampilkan izin yang diberikan olehnya dan memilih kotak centang akan memilih kebijakan.

   Pilih Berikutnya.
6. Pada halaman Nama, tinjau, dan buat, berikan nama Peran dan Deskripsi.
7. Pada Langkah 3: Tambahkan tag, pilih Tambahkan tag baru untuk menambahkan tag berikut untuk memberikan akses App Studio:

   - Kunci: `IsAppStudioDataAccessRole`
   - Nilai: `true`
8. Pilih Buat peran dan catat Nama Sumber Daya Amazon (ARN) yang dihasilkan, Anda akan membutuhkannya saat membuat konektor Aurora di App Studio.

Langkah 3: Buat konektor Aurora di App Studio

Sekarang setelah sumber daya Aurora serta kebijakan dan peran IAM Anda dikonfigurasi, gunakan informasi tersebut untuk membuat konektor di App Studio yang dapat digunakan pembangun untuk menghubungkan aplikasi mereka ke Aurora.

> **Note**
>
> Anda harus memiliki peran Admin di App Studio untuk membuat konektor.

Untuk membuat konektor untuk Aurora

1. Arahkan ke App Studio.
2. Di panel navigasi sisi kiri, pilih Konektor di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar konektor yang ada dengan beberapa detail tentang masing-masing.
3. Pilih + Buat konektor.
4. Pilih konektor Amazon Aurora.
5. Konfigurasikan konektor Anda dengan mengisi kolom berikut:

   - Nama: Masukkan nama untuk konektor Aurora Anda.
   - Deskripsi: Masukkan deskripsi untuk konektor Aurora Anda.

- Peran IAM: Masukkan Nama Sumber Daya Amazon (ARN) dari peran IAM yang dibuat di. Langkah 2b: Buat peran IAM untuk memberi App Studio akses ke sumber daya Aurora Untuk informasi lebih lanjut tentang IAM, lihat Panduan Pengguna IAM.
- Rahasia ARN: Masukkan ARN rahasia dari cluster database. Untuk informasi tentang tempat menemukan ARN rahasia, lihat Melihat detail tentang rahasia untuk cluster DB di Panduan Pengguna Amazon Aurora.
- Wilayah: Pilih AWS Wilayah tempat sumber daya Aurora Anda berada.
- Database ARN: Masukkan ARN dari cluster database. ARN dapat ditemukan di tab Konfigurasi cluster database, mirip dengan ARN rahasia.
- Jenis database: Pilih jenis database, MySQL atau PostgreSQL, yang cocok dengan jenis database yang dibuat. Langkah 1: Buat dan konfigurasikan sumber daya Aurora
- Nama database: Masukkan nama database, yang juga dapat ditemukan di tab Konfigurasi cluster database.
- Tabel yang tersedia: Pilih tabel yang ingin Anda gunakan dengan App Studio menggunakan konektor ini.

6. Pilih Berikutnya untuk meninjau atau menentukan pemetaan entitas.
7. Pilih Buat untuk membuat konektor Aurora. Konektor yang baru dibuat akan muncul di daftar Konektor.

Connect ke Amazon Bedrock

Untuk menghubungkan App Studio dengan Amazon Bedrock sehingga builder dapat mengakses dan menggunakan Amazon Bedrock dalam aplikasi, Anda harus melakukan langkah-langkah berikut:

1. Langkah 1: Aktifkan model Amazon Bedrock
2. Langkah 2: Buat kebijakan dan peran IAM dengan izin Amazon Bedrock yang sesuai
3. Langkah 3: Buat konektor Amazon Bedrock

Langkah 1: Aktifkan model Amazon Bedrock

Gunakan prosedur berikut untuk mengaktifkan model Amazon Bedrock.

Untuk mengaktifkan model Amazon Bedrock

1. Masuk ke AWS Management Console dan buka konsol Amazon Bedrock di https://console.aws.amazon.com/bedrock/.
2. Di panel navigasi kiri, pilih Akses model.
3. Aktifkan model yang ingin Anda gunakan. Untuk informasi selengkapnya, lihat Mengelola akses ke model foundation Amazon Bedrock.

Langkah 2: Buat kebijakan dan peran IAM dengan izin Amazon Bedrock yang sesuai

Untuk menggunakan sumber daya Amazon Bedrock dengan App Studio, administrator harus membuat kebijakan dan peran IAM untuk memberikan izin App Studio untuk mengakses sumber daya. Kebijakan IAM mengontrol sumber daya apa dan operasi apa yang dapat dipanggil terhadap sumber daya tersebut, seperti`InvokeModel`. Kebijakan IAM kemudian dilampirkan ke peran IAM yang digunakan oleh App Studio.

Langkah 2a: Buat kebijakan IAM dengan izin Amazon Bedrock yang sesuai

Kebijakan IAM yang Anda buat dan gunakan dengan App Studio hanya boleh berisi izin minimal yang diperlukan pada sumber daya yang sesuai agar aplikasi dapat mengikuti praktik keamanan terbaik.

Untuk membuat kebijakan IAM dengan izin Amazon Bedrock yang sesuai

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat kebijakan IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi sisi kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Di bagian Editor kebijakan, pilih opsi JSON.
5. Ketik atau tempel dalam dokumen kebijakan JSON. Kebijakan contoh berikut menyediakan `InvokeModel` pada semua resource Amazon Bedrock, menggunakan wildcard ()*.

   Untuk praktik keamanan terbaik, Anda harus mengganti wildcard dengan Nama Sumber Daya Amazon (ARN) dari sumber daya yang ingin Anda gunakan dengan App Studio.

```
{
   "Version": "2012-10-17",
```

```
   "Statement": [
      {
         "Sid": "BedrockAccessForAppStudio",
         "Effect": "Allow",
         "Action": [
            "bedrock:InvokeModel"
         ],
         "Resource": "*"
      }
   ]
}
```

6. Pilih Berikutnya.
7. Pada halaman Tinjau dan buat, berikan nama Kebijakan, seperti**`BedrockAccessForAppStudio`**, dan Deskripsi (opsional).
8. Pilih Buat kebijakan untuk membuat kebijakan.

Langkah 2b: Buat peran IAM untuk memberi App Studio akses ke Amazon Bedrock

Untuk menggunakan Amazon Bedrock dengan App Studio, administrator harus membuat peran IAM untuk memberikan izin App Studio untuk mengakses sumber daya. Peran IAM mengontrol cakupan izin untuk aplikasi App Studio yang akan digunakan, dan digunakan saat membuat konektor. Sebaiknya buat setidaknya satu peran IAM per layanan dan kebijakan.

Untuk membuat peran IAM untuk memberikan App Studio akses ke Amazon Bedrock

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat peran IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi konsol, pilih Peran dan kemudian pilih Buat peran.
3. Di Jenis entitas Tepercaya, pilih Kebijakan kepercayaan khusus.
4. Ganti kebijakan default dengan kebijakan berikut untuk mengizinkan aplikasi App Studio mengambil peran ini di akun Anda.

   Anda harus mengganti placeholder berikut dalam polis. Nilai yang akan digunakan dapat ditemukan di App Studio, di halaman Pengaturan akun.

   - Ganti *111122223333* dengan nomor AWS akun akun yang digunakan untuk menyiapkan instans App Studio, yang terdaftar sebagai ID AWS Akun di pengaturan akun.

- Ganti *11111111-2222-3333-4444-555555555555* dengan ID tim App Studio Anda, yang dicantumkan sebagai ID Tim di setelan akun di instance App Studio Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

Pilih Berikutnya.

5. Di Tambahkan izin, cari dan pilih kebijakan yang Anda buat di langkah sebelumnya (**`BedrockAccessForAppStudio`**). Memilih + di samping kebijakan akan memperluas kebijakan untuk menampilkan izin yang diberikan olehnya dan memilih kotak centang akan memilih kebijakan.

   Pilih Berikutnya.

6. Pada halaman Nama, tinjau, dan buat, berikan nama Peran dan Deskripsi.
7. Pada Langkah 3: Tambahkan tag, pilih Tambahkan tag baru untuk menambahkan tag berikut untuk memberikan akses App Studio:

   - Kunci: `IsAppStudioDataAccessRole`
   - Nilai: `true`

8. Pilih Buat peran dan catat Nama Sumber Daya Amazon (ARN) yang dihasilkan, Anda akan membutuhkannya saat membuat konektor Amazon Bedrock di App Studio pada langkah berikutnya.

Langkah 3: Buat konektor Amazon Bedrock

Setelah sumber daya Amazon Bedrock serta kebijakan dan peran IAM Anda dikonfigurasi, gunakan informasi tersebut untuk membuat konektor di App Studio yang dapat digunakan builder untuk menghubungkan aplikasi mereka ke Amazon Bedrock.

> **Note**
>
> Anda harus memiliki peran Admin di App Studio untuk membuat konektor.

Untuk membuat konektor untuk Amazon Bedrock

1. Arahkan ke App Studio.
2. Di panel navigasi sisi kiri, pilih Konektor di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar konektor yang ada dengan beberapa detail tentang masing-masing.
3. Pilih + Buat konektor.
4. Pilih AWS Layanan lain dari daftar jenis konektor.
5. Konfigurasikan konektor Anda dengan mengisi kolom berikut:
   - Nama: Masukkan nama untuk konektor Amazon Bedrock Anda.
   - Deskripsi: Masukkan deskripsi untuk konektor Amazon Bedrock Anda.
   - Peran IAM: Masukkan Nama Sumber Daya Amazon (ARN) dari peran IAM yang dibuat di. Langkah 2b: Buat peran IAM untuk memberi App Studio akses ke Amazon Bedrock Untuk informasi lebih lanjut tentang IAM, lihat Panduan Pengguna IAM.
   - Layanan: Pilih Bedrock Runtime.

     > **Note**
     >
     > Bedrock Runtime digunakan untuk membuat permintaan inferensi untuk model yang dihosting di Amazon Bedrock, sedangkan Bedrock digunakan untuk mengelola, melatih, dan menerapkan model.

   - Wilayah: Pilih AWS Wilayah tempat sumber daya Amazon Bedrock Anda berada.
6. Pilih Buat.
7. Konektor yang baru dibuat akan muncul di daftar Konektor.

Connect ke Amazon Simple Email Service

Untuk menghubungkan App Studio dengan Amazon SES agar pembuat dapat menggunakannya untuk mengirim pemberitahuan email dari aplikasi mereka, Anda harus melakukan langkah-langkah berikut:

1. Langkah 1: Konfigurasikan sumber daya Amazon SES
2. Langkah 2: Buat kebijakan dan peran IAM dengan izin Amazon SES yang sesuai
3. Langkah 3: Buat konektor Amazon SES

Langkah 1: Konfigurasikan sumber daya Amazon SES

Jika belum, Anda harus terlebih dahulu mengonfigurasi Amazon SES untuk menggunakannya untuk mengirim email. Untuk mempelajari selengkapnya tentang menyiapkan Amazon SES, lihat Memulai Layanan Email Sederhana Amazon di Panduan Pengembang Layanan Email Sederhana Amazon.

Langkah 2: Buat kebijakan dan peran IAM dengan izin Amazon SES yang sesuai

Untuk menggunakan sumber daya Amazon SES dengan App Studio, administrator harus membuat peran IAM untuk memberikan izin App Studio untuk mengakses sumber daya. Peran IAM mengontrol fungsi atau sumber daya Amazon SES yang dapat digunakan di aplikasi App Studio.

Sebaiknya buat setidaknya satu peran IAM per layanan dan kebijakan.

Langkah 2a: Buat kebijakan IAM dengan izin Amazon SES yang sesuai

Kebijakan IAM yang Anda buat dan gunakan dengan App Studio hanya boleh berisi izin minimal yang diperlukan pada sumber daya yang sesuai agar aplikasi dapat mengikuti praktik keamanan terbaik.

Untuk membuat kebijakan IAM dengan izin Amazon SES yang sesuai

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat kebijakan IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi sisi kiri, pilih Kebijakan.
3. Pilih Buat kebijakan.
4. Di bagian Editor kebijakan, pilih opsi JSON.
5. Ketik atau tempel di dokumen kebijakan JSON berikut.

> **Note**
>
> Kebijakan berikut berlaku untuk semua resource Amazon SES menggunakan wildcard (*). Untuk praktik keamanan terbaik, Anda harus mengganti wildcard dengan Nama Sumber Daya Amazon (ARN) dari sumber daya yang ingin Anda gunakan dengan App Studio.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "ses:SendEmail",
            "Resource": "*"
        }
    ]
}
```

6. Pilih Berikutnya.
7. Pada halaman Tinjau dan buat, berikan nama Kebijakan, seperti**`SESForAppStudioPolicy`**, dan Deskripsi (opsional).
8. Pilih Buat kebijakan untuk membuat kebijakan.

Langkah 2b: Buat peran IAM untuk memberi App Studio akses ke Amazon SES

Sekarang, buat peran IAM yang menggunakan kebijakan yang Anda buat sebelumnya. App Studio akan menggunakan kebijakan ini untuk mendapatkan akses ke Amazon SES.

Untuk membuat peran IAM untuk memberikan akses App Studio ke Amazon SES

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat peran IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi sisi kiri, pilih Peran
3. Pilih Buat peran.

4. Di Jenis entitas Tepercaya, pilih Kebijakan kepercayaan khusus.
5. Ganti kebijakan default dengan kebijakan berikut untuk mengizinkan aplikasi App Studio mengambil peran ini di akun Anda.

   Anda harus mengganti placeholder berikut dalam polis. Nilai yang akan digunakan dapat ditemukan di App Studio, di halaman Pengaturan akun.

   - Ganti *111122223333* dengan nomor AWS akun akun yang digunakan untuk menyiapkan instans App Studio, yang terdaftar sebagai AWS Acccount Id di setelan akun.
   - Ganti *11111111-2222-3333-4444-555555555555* dengan ID tim App Studio Anda, yang dicantumkan sebagai ID Tim di setelan akun di instance App Studio Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

   Pilih Berikutnya.
6. Di Tambahkan izin, cari dan pilih kebijakan yang Anda buat di langkah sebelumnya (**`SESForAppStudioPolicy`**). Memilih + di samping kebijakan akan memperluas kebijakan untuk menampilkan izin yang diberikan olehnya dan memilih kotak centang akan memilih kebijakan.

   Pilih Berikutnya.
7. Pada halaman Nama, tinjau, dan buat, berikan nama Peran dan Deskripsi.

8. Pada Langkah 3: Tambahkan tag, pilih Tambahkan tag baru untuk menambahkan tag berikut untuk memberikan akses App Studio:
   - Kunci: `IsAppStudioDataAccessRole`
   - Nilai: `true`
9. Pilih Buat peran dan catat Nama Sumber Daya Amazon (ARN) yang dihasilkan, Anda akan membutuhkannya saat membuat konektor Amazon SES di App Studio.

Langkah 3: Buat konektor Amazon SES

Setelah Amazon SES serta kebijakan dan peran IAM dikonfigurasi, gunakan informasi tersebut untuk membuat konektor di App Studio yang dapat digunakan builder untuk menggunakan Amazon SES di aplikasi mereka.

> **Note**
>
> Anda harus memiliki peran Admin di App Studio untuk membuat konektor.

Untuk membuat konektor untuk Amazon SES

1. Arahkan ke App Studio.
2. Di panel navigasi sisi kiri, pilih Konektor di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar konektor yang ada dengan beberapa detail tentang masing-masing.
3. Pilih + Buat konektor.
4. Pilih AWS Layanan Lain dari daftar jenis konektor.
5. Konfigurasikan konektor Anda dengan mengisi kolom berikut:
   - Nama: Masukkan nama untuk konektor Amazon SES Anda.
   - Deskripsi: Masukkan deskripsi untuk konektor Amazon SES Anda.
   - Peran IAM: Masukkan Nama Sumber Daya Amazon (ARN) dari peran IAM yang dibuat di. Langkah 2b: Buat peran IAM untuk memberi App Studio akses ke Amazon SES Untuk informasi lebih lanjut tentang IAM, lihat Panduan Pengguna IAM.
   - Layanan: Pilih Layanan Email Sederhana.
   - Wilayah: Pilih AWS Wilayah tempat sumber daya Amazon SES Anda berada.
6. Pilih Buat.

7. Konektor yang baru dibuat akan muncul di daftar Konektor.

Connect to AWS services menggunakan konektor Other AWS Services

Meskipun App Studio menawarkan beberapa konektor yang khusus untuk AWS layanan tertentu, Anda juga dapat terhubung ke AWS layanan lain menggunakan konektor AWS Layanan Lain.

> **Note**
>
> Disarankan untuk menggunakan konektor khusus untuk AWS layanan jika tersedia.

Untuk menghubungkan App Studio dengan AWS layanan agar pembuat dapat mengakses dan menggunakan sumber daya layanan dalam aplikasi, Anda harus melakukan langkah-langkah berikut:

1. Membuat peran IAM untuk memberikan akses App Studio ke sumber daya AWS
2. Buat konektor AWS layanan lain

Membuat peran IAM untuk memberikan akses App Studio ke sumber daya AWS

Untuk menggunakan AWS layanan dan sumber daya dengan App Studio, administrator harus membuat peran IAM untuk memberikan izin App Studio untuk mengakses sumber daya. Peran IAM mengontrol ruang lingkup sumber daya yang dapat diakses oleh pembangun dan operasi apa yang dapat disebut melawan sumber daya. Sebaiknya buat setidaknya satu peran IAM per layanan dan kebijakan.

Untuk membuat peran IAM untuk memberikan akses App Studio ke sumber daya AWS

1. Masuk ke konsol IAM dengan pengguna yang memiliki izin untuk membuat peran IAM. Sebaiknya gunakan pengguna administratif yang dibuat diBuat pengguna administratif untuk mengelola AWS sumber daya.
2. Di panel navigasi konsol, pilih Peran dan kemudian pilih Buat peran.
3. Di Jenis entitas Tepercaya, pilih Kebijakan kepercayaan khusus.
4. Ganti kebijakan default dengan kebijakan berikut untuk mengizinkan aplikasi App Studio mengambil peran ini di akun Anda.

   Anda harus mengganti placeholder berikut dalam polis. Nilai yang akan digunakan dapat ditemukan di App Studio, di halaman Pengaturan akun.

- Ganti *111122223333* dengan nomor AWS akun akun yang digunakan untuk menyiapkan instans App Studio, yang terdaftar sebagai ID AWS Akun di pengaturan akun.
- Ganti *11111111-2222-3333-4444-555555555555* dengan ID tim App Studio Anda, yang dicantumkan sebagai ID Tim di setelan akun di instance App Studio Anda.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId": "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

Pilih Berikutnya.

5. Di Tambahkan izin, cari dan pilih kebijakan yang memberikan izin yang sesuai untuk peran tersebut. Memilih + di samping kebijakan akan memperluas kebijakan untuk menampilkan izin yang diberikan olehnya dan memilih kotak centang akan memilih kebijakan tersebut. Untuk informasi lebih lanjut tentang IAM, lihat Panduan Pengguna IAM.

   Pilih Berikutnya.

6. Dalam Rincian peran, berikan nama dan deskripsi.
7. Pada Langkah 3: Tambahkan tag, pilih Tambahkan tag baru untuk menambahkan tag berikut untuk memberikan akses App Studio:

   - Kunci: `IsAppStudioDataAccessRole`
   - Nilai: `true`

8. Pilih Buat peran dan catat Nama Sumber Daya Amazon (ARN) yang dihasilkan, Anda akan memerlukannya saat membuat konektor AWS Layanan lain di App Studio.

Buat konektor AWS layanan lain

Setelah peran IAM Anda dikonfigurasi, gunakan informasi tersebut untuk membuat konektor di App Studio yang dapat digunakan builder untuk menghubungkan aplikasi mereka ke layanan dan sumber daya.

> **Note**
>
> Anda harus memiliki peran Admin di App Studio untuk membuat konektor.

Untuk terhubung ke AWS layanan menggunakan konektor AWS Layanan lain

1. Arahkan ke App Studio.
2. Di panel navigasi sisi kiri, pilih Konektor di bagian Kelola.
3. Pilih + Buat konektor.
4. Pilih AWS Layanan lain di bagian AWS konektor dari daftar layanan yang didukung.
5. Konfigurasikan konektor AWS layanan Anda dengan mengisi kolom berikut:
   - Nama: Berikan nama untuk konektor Anda.
   - Deskripsi: Berikan deskripsi untuk konektor Anda.
   - Peran IAM: Masukkan Nama Sumber Daya Amazon (ARN) dari peran IAM yang dibuat di. Membuat peran IAM untuk memberikan akses App Studio ke sumber daya AWS
   - Layanan: Pilih AWS layanan yang ingin Anda sambungkan ke App Studio.
   - Wilayah: Pilih AWS Wilayah tempat AWS sumber daya Anda berada.
6. Pilih Buat. Konektor yang baru dibuat akan muncul di daftar konektor.

Gunakan sumber data terenkripsi dengan CMKs

Topik ini berisi informasi tentang menyiapkan dan menghubungkan App Studio ke sumber data yang dienkripsi menggunakan Kunci Terkelola AWS KMS Pelanggan (CMK).

Daftar Isi

- Menggunakan tabel penyimpanan data terkelola terenkripsi
- Menggunakan tabel DynamoDB terenkripsi

Menggunakan tabel penyimpanan data terkelola terenkripsi

Gunakan prosedur berikut untuk mengenkripsi tabel DynamoDB yang digunakan oleh entitas penyimpanan terkelola di aplikasi App Studio Anda. Untuk informasi selengkapnya tentang entitas data terkelola, lihatEntitas data terkelola di AWS App Studio.

Untuk menggunakan tabel penyimpanan data terkelola terenkripsi

1. Jika perlu, buat entitas data terkelola dalam aplikasi di App Studio. Untuk informasi selengkapnya, lihat Membuat entitas dengan sumber data terkelola App Studio.
2. Tambahkan pernyataan kebijakan dengan izin untuk mengenkripsi dan mendekripsi data tabel dengan CMK Anda ke peran `AppStudioManagedStorageDDBAccess` IAM dengan melakukan langkah-langkah berikut:

   a. Buka konsol IAM di https://console.aws.amazon.com/iam/.

   > **Important**
   >
   > Anda harus menggunakan akun yang sama yang digunakan untuk membuat instance App Studio.

   b. Di panel navigasi konsol IAM, pilih Peran.

   c. Pilih `AppStudioManagedStorageDDBAccess`.

   d. Di Kebijakan izin, pilih Tambahkan izin, lalu pilih Buat kebijakan sebaris.

   e. Pilih JSON dan ganti isinya dengan kebijakan berikut, ganti yang berikut:

   - Ganti *`team_account_id`* dengan ID tim App Studio Anda, yang dapat ditemukan di pengaturan akun Anda.
   - Ganti *`CMK_id`* dengan ID CMK. Untuk menemukannya, lihat Temukan ID kunci dan kunci ARN.

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Sid": "connector_cmk_support",
      "Effect": "Allow",
      "Action": [ "kms:Decrypt", "kms:Encrypt"],
      "Resource": "arn:aws:kms:us-west-2:team_account_id:key/CMK_id"
    }
  ]
}
```

3. Enkripsi tabel DynamoDB yang digunakan oleh entitas data terkelola App Studio Anda dengan melakukan langkah-langkah berikut:

   a. Buka konsol Amazon DynamoDB di. https://console.aws.amazon.com/dynamodbv2/

   b. Pilih tabel yang ingin Anda enkripsi. Anda dapat menemukan nama tabel di tab Connection dari entitas terkait di App Studio.

   c. Pilih Pengaturan tambahan.

   d. Di Enkripsi, pilih Kelola enkripsi.

   e. Pilih Disimpan di akun Anda, dan dimiliki dan dikelola oleh Anda dan pilih CMK Anda.

4. Uji perubahan Anda dengan menerbitkan ulang aplikasi Anda dan memastikan bahwa membaca dan menulis data berfungsi di lingkungan Pengujian dan Produksi, dan menggunakan tabel ini di entitas lain berfungsi seperti yang diharapkan.

> **Note**
>
> Setiap entitas data terkelola yang baru ditambahkan menggunakan kunci terkelola DynamoDB secara default, dan harus diperbarui untuk menggunakan CMK dengan mengikuti langkah-langkah sebelumnya.

Menggunakan tabel DynamoDB terenkripsi

Gunakan prosedur berikut untuk mengonfigurasi tabel DynamoDB terenkripsi yang akan digunakan di aplikasi App Studio Anda.

Untuk menggunakan tabel DynamoDB terenkripsi

1. Ikuti instruksi Langkah 1: Buat dan konfigurasikan sumber daya DynamoDB dengan perubahan berikut:

- Konfigurasikan tabel Anda untuk dienkripsi. Untuk informasi selengkapnya, lihat Menentukan kunci enkripsi untuk tabel baru di Panduan Pengembang Amazon DynamoDB.

2. Ikuti petunjuk diLangkah 2: Buat kebijakan dan peran IAM dengan izin DynamoDB yang sesuai, lalu perbarui kebijakan izin pada peran baru dengan menambahkan pernyataan kebijakan baru dengan mengizinkannya mengenkripsi dan mendekripsi data tabel menggunakan CMK Anda dengan melakukan langkah-langkah berikut:

   a. Jika perlu, navigasikan ke peran Anda di konsol IAM.

   b. Di Kebijakan izin, pilih Tambahkan izin, lalu pilih Buat kebijakan sebaris.

   c. Pilih JSON dan ganti isinya dengan kebijakan berikut, ganti yang berikut:

      - Ganti *team_account_id* dengan ID tim App Studio Anda, yang dapat ditemukan di pengaturan akun Anda.
      - Ganti *CMK_id* dengan ID CMK. Untuk menemukannya, lihat Temukan ID kunci dan kunci ARN.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "connector_cmk_support",
      "Effect": "Allow",
      "Action": [ "kms:Decrypt", "kms:Encrypt"],
      "Resource": "arn:aws:kms:us-west-2:team_account_id:key/CMK_id"
    }
  ]
}
```

3. Buat konektor dengan mengikuti petunjuk Buat konektor DynamoDB dan menggunakan peran yang Anda buat sebelumnya.

4. Uji konfigurasi dengan memublikasikan aplikasi yang menggunakan konektor dan tabel DynamoDB ke Pengujian atau Produksi. Pastikan bahwa membaca dan menulis data berfungsi, dan menggunakan tabel ini untuk membuat entitas lain juga berfungsi.

> **Note**
>
> Ketika tabel DynamoDB baru dibuat, Anda harus mengonfigurasinya untuk dienkripsi menggunakan CMK dengan mengikuti langkah-langkah sebelumnya.

Connect ke layanan pihak ketiga

Topik

Konektor OpenAPI vs Konektor API

Untuk mengirim permintaan API ke layanan pihak ketiga dari aplikasi App Studio, Anda harus membuat dan mengonfigurasi konektor yang digunakan aplikasi untuk mengautentikasi dengan layanan dan mengonfigurasi panggilan API. App Studio menyediakan tipe `OpenAPI Connector` konektor `API Connector` dan konektor untuk mencapai hal ini, yang dijelaskan sebagai berikut:

- Konektor API: Digunakan untuk mengonfigurasi otentikasi dan meminta informasi untuk semua jenis REST API.
- Konektor OpenAPI: Digunakan untuk mengonfigurasi otentikasi dan meminta informasi APIs yang telah mengadopsi Spesifikasi OpenAPI (OAS). APIs yang mematuhi OAS memberikan beberapa manfaat, termasuk standardisasi, keamanan, tata kelola, dan dokumentasi.

App Studio merekomendasikan penggunaan `OpenAPI Connector` untuk semua APIs yang mematuhi OAS, dan menyediakan File Spesifikasi OpenAPI. Untuk informasi selengkapnya tentang OpenAPI, lihat Apa itu OpenAPI? dalam dokumentasi Swagger.

Connect ke layanan pihak ketiga dan APIs (generik)

Gunakan prosedur berikut untuk membuat Konektor API generik di App Studio. Konektor API digunakan untuk menyediakan aplikasi App Studio akses ke layanan, sumber daya, atau operasi pihak ketiga.

Untuk terhubung ke layanan pihak ketiga dengan Konektor API

1. Di panel navigasi sisi kiri, pilih konektor di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar konektor yang ada dengan beberapa detail tentang masing-masing.
2. Pilih + Buat konektor.
3. Pilih Konektor API. Sekarang, konfigurasikan konektor Anda dengan mengisi bidang berikut.
4. Nama konektor: Berikan nama untuk konektor Anda.
5. Deskripsi konektor: Berikan deskripsi untuk konektor Anda.
6. URL Dasar: Situs web atau host koneksi pihak ketiga. Misalnya, `www.slack.com`.
7. Metode otentikasi: Pilih metode untuk otentikasi dengan layanan target.
    - Tidak ada: Akses layanan target tanpa otentikasi.
    - Dasar: Akses layanan target menggunakan Username dan Password yang diperoleh dari layanan yang terhubung.
    - Token Pembawa: Akses layanan target menggunakan nilai Token dari token otentikasi yang diperoleh dari akun pengguna layanan atau pengaturan API.
    - OAuth 2.0: Akses layanan target menggunakan protokol OAuth 2.0, yang memberikan App Studio akses ke layanan dan sumber daya tanpa membagikan kredensi atau identitas apa pun. Untuk menggunakan metode otentikasi OAuth 2.0, Anda harus terlebih dahulu membuat aplikasi dari layanan yang terhubung ke yang mewakili App Studio untuk mendapatkan informasi yang diperlukan. Dengan informasi tersebut, isi bidang-bidang berikut:

        a. Aliran kredensial klien: Ideal untuk system-to-system interaksi di mana aplikasi bertindak atas namanya sendiri tanpa interaksi pengguna. Misalnya, aplikasi CRM yang memperbarui catatan Salesforce secara otomatis berdasarkan catatan baru yang ditambahkan oleh pengguna, atau aplikasi yang mengambil dan menampilkan data transaksi dalam laporan.

            1. Di Client ID, masukkan ID yang diperoleh dari OAuth aplikasi yang dibuat di layanan target.
            2. Dalam rahasia Klien, masukkan rahasia yang diperoleh dari OAuth aplikasi yang dibuat di layanan target.
            3. Di URL token akses, masukkan URL token yang diperoleh dari OAuth aplikasi yang dibuat di layanan target.

4. Secara opsional, di Scopes, masukkan cakupan untuk aplikasi. Cakupan adalah izin atau tingkat akses yang diperlukan oleh aplikasi. Lihat dokumentasi API layanan target untuk memahami cakupannya, dan konfigurasikan hanya yang dibutuhkan aplikasi App Studio Anda.

Pilih Verifikasi koneksi untuk menguji otentikasi dan koneksi.

b. Alur kode otorisasi: Ideal untuk aplikasi yang memerlukan tindakan atas nama pengguna. Misalnya, aplikasi dukungan pelanggan tempat pengguna masuk dan melihat serta memperbarui tiket dukungan, atau aplikasi penjualan tempat setiap anggota tim masuk untuk melihat dan mengelola data penjualan mereka.

1. Di Client ID, masukkan ID yang diperoleh dari OAuth aplikasi yang dibuat di layanan target.
2. Dalam rahasia Klien, masukkan rahasia yang diperoleh dari OAuth aplikasi yang dibuat di layanan target.
3. Di URL Otorisasi, masukkan URL otorisasi dari layanan target.
4. Di URL token akses, masukkan URL token yang diperoleh dari OAuth aplikasi yang dibuat di layanan target.
5. Secara opsional, di Scopes, masukkan cakupan untuk aplikasi. Cakupan adalah izin atau tingkat akses yang diperlukan oleh aplikasi. Lihat dokumentasi API layanan target untuk memahami cakupannya, dan konfigurasikan hanya yang dibutuhkan aplikasi App Studio Anda.

8. Header: Tambahkan header HTTP yang digunakan untuk menyediakan metadata tentang permintaan atau respons. Anda dapat menambahkan kunci dan nilai, atau hanya memberikan kunci yang pembangun dapat memberikan nilai dalam aplikasi.
9. Parameter kueri: Tambahkan parameter kueri yang digunakan untuk meneruskan opsi, filter, atau data sebagai bagian dari URL permintaan. Seperti header, Anda dapat memberikan kunci dan nilai, atau hanya memberikan kunci yang pembangun dapat memberikan nilai dalam aplikasi.
10. Pilih Buat. Konektor yang baru dibuat akan muncul di daftar Konektor.

Sekarang setelah konektor dibuat, pembangun dapat menggunakannya di aplikasi mereka.

Connect ke layanan dengan OpenAPI

Untuk menghubungkan App Studio dengan layanan menggunakan OpenAPI untuk memungkinkan pembangun membangun aplikasi yang mengirim permintaan dan menerima tanggapan dari layanan, lakukan langkah-langkah berikut:

1. Dapatkan file Spesifikasi OpenAPI dan kumpulkan informasi layanan
2. Buat konektor OpenAPI

Dapatkan file Spesifikasi OpenAPI dan kumpulkan informasi layanan

Untuk menghubungkan layanan ke App Studio dengan OpenAPI, lakukan langkah-langkah berikut:

1. Buka layanan yang ingin Anda sambungkan ke App Studio dan temukan file JSON Spesifikasi OpenAPI.

   > **Note**
   >
   > App Studio mendukung file Spesifikasi OpenAPI yang sesuai dengan versi Spesifikasi OpenAPI Versi 3.0.0 atau lebih tinggi.

2. Kumpulkan data yang diperlukan untuk mengonfigurasi konektor OpenAPI, termasuk yang berikut ini:

   - URL dasar untuk menghubungkan ke layanan.
   - Kredensi otentikasi, seperti token atau nama pengguna/kata sandi.
   - Jika berlaku, header apa pun.
   - Jika berlaku, parameter kueri apa pun.

Buat konektor OpenAPI

Untuk membuat konektor untuk OpenAPI

1. Arahkan ke App Studio.
2. Di panel navigasi sisi kiri, pilih Konektor di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar konektor yang ada dengan beberapa detail tentang masing-masing.
3. Pilih + Buat konektor.

4. Pilih OpenAPI Connector dari daftar jenis konektor. Sekarang, konfigurasikan konektor Anda dengan mengisi bidang berikut.
5. Nama: Masukkan nama untuk konektor OpenAPI Anda.
6. Deskripsi: Masukkan deskripsi untuk konektor OpenAPI Anda.
7. URL dasar: Masukkan URL dasar untuk menghubungkan ke layanan.
8. Metode otentikasi: Pilih metode untuk otentikasi dengan layanan target.
    - Tidak ada: Akses layanan target tanpa otentikasi.
    - Dasar: Akses layanan target menggunakan Username dan Password yang diperoleh dari layanan yang terhubung.
    - Token Pembawa: Akses layanan target menggunakan nilai Token dari token otentikasi yang diperoleh dari akun pengguna layanan atau pengaturan API.
    - OAuth 2.0: Akses layanan target menggunakan protokol OAuth 2.0, yang memberikan App Studio akses ke layanan dan sumber daya tanpa membagikan kredensi atau identitas apa pun. Untuk menggunakan metode otentikasi OAuth 2.0, Anda harus terlebih dahulu membuat aplikasi dari layanan yang terhubung ke yang mewakili App Studio untuk mendapatkan informasi yang diperlukan. Dengan informasi tersebut, isi bidang-bidang berikut:

        a. Aliran kredensi klien:

            1. Di Client ID, masukkan ID dari layanan target.
            2. Dalam rahasia Klien, masukkan rahasia dari layanan target.
            3. Di URL token akses, masukkan URL token dari layanan target.
            4. Secara opsional, di Scopes, masukkan cakupan untuk aplikasi. Cakupan adalah izin atau tingkat akses yang diperlukan oleh aplikasi. Lihat dokumentasi API layanan target untuk memahami cakupannya, dan konfigurasikan hanya yang dibutuhkan aplikasi App Studio Anda.

            Tambahkan Variabel apa pun yang akan dikirim bersama layanan dengan setiap panggilan, dan pilih Verifikasi koneksi untuk menguji otentikasi dan koneksi.

        b. Alur kode otorisasi:

            1. Di Client ID, masukkan ID dari layanan target.
            2. Dalam rahasia Klien, masukkan rahasia dari layanan target.

3. Di URL Otorisasi, masukkan URL otorisasi dari layanan target.
4. Di URL token akses, masukkan URL token dari layanan target.
5. Secara opsional, di Scopes, masukkan cakupan untuk aplikasi. Cakupan adalah izin atau tingkat akses yang diperlukan oleh aplikasi. Lihat dokumentasi API layanan target untuk memahami cakupannya, dan konfigurasikan hanya yang dibutuhkan aplikasi App Studio Anda.

9. Variabel: Tambahkan variabel yang akan dikirim ke layanan dengan setiap panggilan. Variabel yang ditambahkan selama konfigurasi disimpan dengan aman dan hanya diakses selama runtime aplikasi yang menggunakan koneksi.
10. Header: Tambahkan header HTTP yang digunakan untuk menyediakan metadata tentang permintaan atau respons. Anda dapat menambahkan kunci dan nilai, atau hanya memberikan kunci yang pembangun dapat memberikan nilai dalam aplikasi.
11. Parameter kueri: Tambahkan parameter kueri yang digunakan untuk meneruskan opsi, filter, atau data sebagai bagian dari URL permintaan. Seperti header, Anda dapat memberikan kunci dan nilai, atau hanya memberikan kunci yang pembangun dapat memberikan nilai dalam aplikasi.
12. File Spesifikasi OpenAPI: Unggah file JSON Spesifikasi OpenAPI dengan menyeret dan melepaskan, atau memilih Pilih file untuk menavigasi sistem file lokal Anda dan pilih file yang akan diunggah.

    Setelah ditambahkan, file diproses dan daftar opsi yang tersedia ditampilkan. Pilih operasi yang diperlukan untuk konektor Anda.
13. Pilih Buat. Konektor yang baru dibuat akan muncul di daftar Konektor.

Sekarang setelah konektor dibuat, pembangun dapat menggunakannya di aplikasi mereka.

Connect ke Salesforce

Untuk menghubungkan App Studio dengan Salesforce untuk memungkinkan pembangun mengakses dan menggunakan sumber daya Salesforce dalam aplikasi, Anda harus membuat dan mengonfigurasi aplikasi yang terhubung di Salesforce dan membuat konektor Salesforce di App Studio.

Untuk menghubungkan Salesforce dengan App Studio

1. Di App Studio, di panel navigasi, pilih Konektor di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar konektor yang ada dengan beberapa detail tentang masing-masing.
2. Pilih + Buat konektor.
3. Pilih Salesforce dari daftar jenis konektor untuk membuka halaman pembuatan konektor.
4. Perhatikan URL Pengalihan, yang akan Anda gunakan untuk mengonfigurasi Salesforce dalam langkah-langkah berikut.
5. Langkah selanjutnya adalah membuat aplikasi yang terhubung di Salesforce. Di tab atau jendela lain, arahkan ke instans Salesforce Anda.
6. Di kotak Pencarian Cepat, cari **`App Manager`** lalu pilih Manajer Aplikasi.
7. Pilih Aplikasi Terhubung Baru.
8. Di Nama Aplikasi Terhubung dan Nama API, masukkan nama untuk aplikasi Anda. Itu tidak harus cocok dengan nama aplikasi App Studio Anda.
9. Berikan informasi kontak sesuai kebutuhan.
10. Di bagian API (Aktifkan OAuth Pengaturan), aktifkan Aktifkan OAuth Pengaturan.
11. Di URL Callback, masukkan URL Pengalihan yang Anda catat sebelumnya dari App Studio.
12. Di OAuth Lingkup Terpilih, tambahkan cakupan izin yang diperlukan dari daftar. App Studio dapat berinteraksi dengan Salesforce REST APIs untuk melakukan operasi CRUD pada lima objek: Akun, Kasus, Kontak, Prospek, dan Peluang. Disarankan untuk menambahkan Akses penuh (penuh) untuk memastikan bahwa aplikasi App Studio Anda memiliki semua izin atau cakupan yang relevan.
13. Nonaktifkan opsi Require Proof Key for Code Exchange (PKCE) untuk Alur Otorisasi yang Didukung. PKCE tidak didukung oleh App Studio.
14. Aktifkan Require Secret for Web Server Flow dan Require Secret for Refresh Token Flow untuk mengikuti praktik keamanan terbaik.
15. App Studio mendukung kedua alur autentikasi berikut:

    - Aliran Kredensial Klien: Ideal untuk server-to-server interaksi di mana aplikasi bertindak atas namanya sendiri tanpa interaksi pengguna. Misalnya, daftar semua informasi prospek untuk tim karyawan sementara yang tidak memiliki akses Salesforce.
    - Alur Kode Otorisasi: Sesuai untuk aplikasi yang bertindak atas nama pengguna, seperti akses atau tindakan data pribadi. Misalnya, daftar prospek setiap manajer penjualan yang bersumber atau dimiliki oleh mereka untuk melakukan tugas lain melalui aplikasi ini.

- Untuk Alur Kredensial Klien:

    a. Aktifkan Aktifkan Alur Kredensial Klien. Tinjau dan konfirmasikan pesan.

    b. Simpan aplikasi.

    c. Anda harus memilih pengguna eksekusi, meskipun tidak ada interaksi pengguna dalam alur. Dengan memilih pengguna eksekusi, Salesforce mengembalikan token akses atas nama pengguna.

        1. Di Manajer Aplikasi, dari daftar aplikasi, pilih panah aplikasi App Studio dan pilih Kelola.
        2. Pilih Edit Kebijakan
        3. Di Client Credentials Flow, tambahkan pengguna yang sesuai.

- Untuk Alur Kode Otorisasi, aktifkan Aktifkan Kode Otorisasi dan Alur Kredensial

16. Salesforce menyediakan ID Klien dan Rahasia Klien, yang harus digunakan untuk mengonfigurasi konektor di App Studio dalam langkah-langkah berikut.

    a. Di Manajer Aplikasi, pilih panah aplikasi App Studio dan pilih Lihat.

    b. Di bagian API (Aktifkan OAuth Pengaturan), pilih Kelola Detail Konsumen. Ini dapat mengirim email untuk kunci verifikasi, yang harus Anda masukkan untuk konfirmasi.

    c. Perhatikan Kunci Konsumen (ID Klien) dan Rahasia Konsumen (Rahasia Klien).

17. Kembali ke App Studio, konfigurasikan dan buat konektor Anda dengan mengisi kolom berikut.
18. Dalam Nama, masukkan nama untuk konektor Salesforce Anda.
19. Dalam Deskripsi, masukkan deskripsi untuk konektor Salesforce Anda.
20. Di URL Dasar, masukkan URL dasar untuk instance Salesforce Anda. Seharusnya terlihat seperti ini:`https://hostname.salesforce.com/services/data/v60.0`, mengganti *hostname* dengan nama instance Salesforce Anda.
21. Dalam metode otentikasi, pastikan OAuth 2.0 dipilih.
22. Di OAuth 2.0 Flow, pilih metode OAuth otentikasi dan isi bidang terkait:

    - Pilih alur kredensi Klien untuk digunakan dalam aplikasi yang bertindak atas nama mereka sendiri, untuk system-to-system integrasi.

        a. Di Client ID, masukkan Consumer Key yang diperoleh sebelumnya dari Salesforce.

b. Dalam rahasia Klien, masukkan Rahasia Konsumen, yang diperoleh sebelumnya dari Salesforce.

c. Di URL token akses, masukkan titik akhir token OAuth 2.0. Seharusnya terlihat seperti ini:`https://hostname/services/oauth2/token`, mengganti *hostname* dengan nama instance Salesforce Anda. Untuk informasi selengkapnya, lihat dokumentasi Titik OAuth Akhir Salesforce.

d. Pilih Verifikasi koneksi untuk menguji otentikasi dan koneksi.

- Pilih Alur kode otorisasi untuk digunakan dalam aplikasi yang bertindak atas nama pengguna.

a. Di Client ID, masukkan Consumer Key yang diperoleh sebelumnya dari Salesforce.

b. Dalam rahasia Klien, masukkan Rahasia Konsumen, yang diperoleh sebelumnya dari Salesforce.

c. Di URL Otorisasi, masukkan titik akhir otorisasi. Seharusnya terlihat seperti ini:`https://hostname/services/oauth2/authorize`, mengganti *hostname* dengan nama instance Salesforce Anda. Untuk informasi selengkapnya, lihat dokumentasi Titik OAuth Akhir Salesforce.

d. Di URL token akses, masukkan titik akhir token OAuth 2.0. Seharusnya terlihat seperti ini:`https://hostname/services/oauth2/token`, mengganti *hostname* dengan nama instance Salesforce Anda. Untuk informasi selengkapnya, lihat dokumentasi Titik OAuth Akhir Salesforce.

23. Dalam Operasi, pilih operasi Salesforce yang akan didukung konektor Anda. Operasi dalam daftar ini telah ditentukan sebelumnya dan mewakili tugas umum dalam Salesforce, seperti membuat, mengambil, memperbarui, atau menghapus catatan dari objek umum.
24. Pilih Buat. Konektor yang baru dibuat akan muncul di daftar Konektor.

Melihat, mengedit, dan menghapus konektor

Untuk melihat, mengedit, atau menghapus konektor yang ada

1. Di panel navigasi, pilih Konektor di bagian Kelola. Anda akan dibawa ke halaman yang menampilkan daftar konektor yang ada dengan rincian berikut untuk setiap konektor:

   - Nama: Nama konektor yang disediakan selama pembuatan.
   - Deskripsi: Deskripsi konektor yang disediakan selama pembuatan.

- Terhubung ke: Layanan yang dihubungkan oleh konektor ke App Studio. Nilai API mewakili koneksi ke layanan pihak ketiga.
- Dibuat oleh: Pengguna yang membuat konektor.
- Tanggal dibuat: Tanggal konektor dibuat.

2. Untuk melihat detail lebih lanjut tentang konektor, atau mengedit atau menghapus konektor, gunakan petunjuk berikut:
   - Untuk melihat informasi selengkapnya tentang konektor tertentu, pilih Lihat untuk konektor tersebut.
   - Untuk mengedit konektor, pilih menu tarik-turun di sebelah View dan pilih Edit.
   - Untuk menghapus konektor, pilih menu tarik-turun di sebelah Lihat dan pilih Hapus.

Menghapus instans App Studio

Gunakan prosedur dalam topik ini untuk menghapus instance App Studio Anda. Jika Anda membuat sumber daya di layanan lain untuk digunakan dengan App Studio, tinjau dan hapus seperlunya agar tidak dikenakan biaya.

Anda mungkin ingin menghapus instans App Studio karena alasan berikut:

- Anda tidak lagi ingin menggunakan App Studio.
- Anda ingin membuat instance App Studio di AWS Region yang berbeda. Karena App Studio hanya mendukung memiliki instance di satu Wilayah pada satu waktu, Anda harus menghapus instans yang ada untuk membuat instance lain.

> **Warning**
>
> Menghapus instance App Studio juga menghapus semua resource App Studio, seperti aplikasi dan konektor. Menghapus instance tidak dapat dibatalkan.

Untuk menghapus instance App Studio

1. Buka konsol App Studio di https://console.aws.amazon.com/appstudio/.
2. Pilih Wilayah tempat instance App Studio Anda ada.
3. Di panel navigasi, pilih Instance.

4. Pilih Tindakan untuk membuka dropdown dengan tindakan instance tambahan.
5. Pilih Hapus instans App Studio.
6. Masuk **`confirm`** dan pilih Hapus.
7. Mungkin perlu beberapa saat agar penghapusan instans Anda diproses. Setelah dihapus, Anda akan menerima email konfirmasi. Setelah Anda menerima email, Anda dapat membuat contoh lain jika diinginkan.

Dokumentasi pembangun

Topik berikut berisi informasi untuk membantu pengguna di App Studio yang membuat, mengedit, dan menerbitkan aplikasi.

Topik

Tutorial

Topik

Buat aplikasi peringkas teks AI dengan Amazon Bedrock

Dalam tutorial ini, Anda akan membangun aplikasi di App Studio yang menggunakan Amazon Bedrock untuk memberikan ringkasan ringkas input teks dari pengguna akhir. Aplikasi ini berisi antarmuka pengguna sederhana di mana pengguna dapat memasukkan teks apa pun yang ingin

diringkas. Ini bisa berupa catatan pertemuan, konten artikel, temuan penelitian, atau informasi tekstual lainnya. Setelah pengguna memasukkan teks, mereka dapat menekan tombol untuk mengirim teks ke Amazon Bedrock, yang akan memprosesnya menggunakan model Claude 3 Sonnet dan mengembalikan versi yang diringkas.

Daftar Isi

Prasyarat

Sebelum Anda memulai, tinjau dan lengkapi prasyarat berikut:

- Akses ke AWS App Studio. Perhatikan bahwa Anda harus memiliki peran Admin untuk membuat konektor dalam tutorial ini.
- Opsional: Tinjau AWS Konsep App Studio dan Tutorial: Mulai membangun dari aplikasi kosong untuk membiasakan diri dengan konsep App Studio yang penting.

Langkah 1: Buat dan konfigurasikan peran IAM dan konektor App Studio

Untuk menyediakan akses App Studio ke model Amazon Bedrock, Anda harus:

1. Aktifkan model Amazon Bedrock yang ingin Anda gunakan di aplikasi Anda. Untuk tutorial ini, Anda akan menggunakan Claude 3 Sonnet, jadi pastikan Anda mengaktifkan model itu.
2. Buat peran IAM dengan izin yang sesuai untuk Amazon Bedrock.
3. Buat konektor App Studio dengan peran IAM yang akan digunakan di aplikasi Anda.

Pergi ke Connect ke Amazon Bedrock untuk petunjuk rinci, dan kembali ke tutorial ini setelah Anda mengikuti langkah-langkah dan membuat konektor.

Langkah 2: Buat aplikasi

Gunakan prosedur berikut untuk membuat aplikasi kosong di App Studio yang akan Anda buat ke dalam aplikasi peringkas teks.

1. Masuk ke App Studio.
2. Arahkan ke hub pembuat dan pilih + Buat aplikasi.
3. Pilih Mulai dari awal.
4. Di bidang Nama aplikasi, berikan nama untuk aplikasi Anda, seperti**`Text Summarizer`**.
5. Jika Anda diminta untuk memilih sumber data atau konektor, pilih Lewati untuk keperluan tutorial ini.
6. Pilih Berikutnya untuk melanjutkan.
7. (Opsional): Tonton tutorial video untuk ikhtisar singkat tentang membangun aplikasi di App Studio.
8. Pilih Edit aplikasi, yang akan membawa Anda ke studio aplikasi.

Langkah 3: Buat dan konfigurasikan otomatisasi

Anda menentukan logika dan perilaku aplikasi App Studio dalam otomatisasi. Otomatisasi terdiri dari langkah-langkah individual yang dikenal sebagai tindakan, parameter yang digunakan untuk meneruskan data ke tindakan dari sumber daya lain, dan output yang dapat digunakan oleh otomatisasi atau komponen lain. Pada langkah ini, Anda akan membuat otomatisasi yang menangani interaksi dengan Amazon Bedrock dengan yang berikut:

- Input: Parameter untuk meneruskan input teks dari pengguna ke otomatisasi.
- Tindakan: Satu tindakan GenAI Prompt yang mengirimkan input teks ke Amazon Bedrock dan mengembalikan ringkasan teks keluaran.
- Output: Output otomatisasi yang terdiri dari ringkasan yang diproses dari Amazon Bedrock, yang dapat digunakan di aplikasi Anda.

Untuk membuat dan mengonfigurasi otomatisasi yang mengirimkan prompt ke Amazon Bedrock dan memproses serta mengembalikan ringkasan

1. Pilih tab Automations di bagian atas kanvas.
2. Pilih + Tambahkan otomatisasi.
3. Di panel sebelah kanan, pilih Properties.
4. Perbarui nama otomatisasi dengan memilih ikon pensil. Masuk **`InvokeBedrock`** dan tekan Enter.
5. Tambahkan parameter ke otomatisasi yang akan digunakan untuk meneruskan input prompt teks dari pengguna ke otomatisasi yang akan digunakan dalam permintaan ke Amazon Bedrock dengan melakukan langkah-langkah berikut:

   a. Di kanvas, di kotak parameter, pilih + Tambah.
   b. Di Nama, masukkan **`input`**.
   c. Dalam Deskripsi, masukkan deskripsi apa pun, seperti**`Text to be sent to Amazon Bedrock`**.
   d. Di Type, pilih String.
   e. Pilih Tambah untuk membuat parameter.
6. Tambahkan tindakan GenAI Prompt dengan melakukan langkah-langkah berikut:

   a. Di panel sebelah kanan, pilih Tindakan.
   b. Pilih GenAI Prompt untuk menambahkan tindakan.
7. Konfigurasikan tindakan dengan melakukan langkah-langkah berikut:

   a. Pilih tindakan dari kanvas untuk membuka menu Properties sebelah kanan.
   b. Ubah nama tindakan menjadi **`PromptBedrock`** dengan memilih ikon pensil, memasukkan nama, dan menekan enter.
   c. Di Connector, pilih konektor yang dibuat diLangkah 1: Buat dan konfigurasikan peran IAM dan konektor App Studio.
   d. Di Model, pilih model Amazon Bedrock yang ingin Anda gunakan untuk memproses prompt. Dalam tutorial ini, Anda akan memilih Claude 3.5 Sonnet.
   e. Di Prompt pengguna, masukkan`{{params.input}}`. Ini mewakili `input` parameter yang Anda buat sebelumnya, dan akan berisi input teks oleh pengguna aplikasi Anda.
   f. Dalam Prompt sistem, masukkan instruksi prompt sistem yang ingin Anda kirim ke Amazon Bedrock. Untuk tutorial ini, masukkan yang berikut ini:

```
You are a highly efficient text summarizer. Provide a concise summary of the
 prompted text, capturing the key points and main ideas.
```

g. Pilih Minta setelan untuk memperluasnya, dan perbarui bidang berikut:

- Dalam Suhu, masukkan`0`. Tempearture menentukan keacakan atau kreativitas output pada skala 0 hingga 10. Semakin tinggi angkanya, semakin kreatif responsnya.
- Di Token Maks, masukkan `4096` untuk membatasi panjang respons.

8. Output dari otomatisasi ini akan menjadi teks yang diringkas, namun, secara default otomatisasi tidak membuat output. Konfigurasikan otomatisasi untuk membuat output otomatisasi dengan melakukan langkah-langkah berikut:

a. Di navigasi sebelah kiri, pilih otomatisasi. InvokeBedrock

b. Di menu Properties sebelah kanan, di Output, pilih + Tambah.

c. Di Output, masukkan**`{{results.PromptBedrock.text}}`**. Ekspresi ini mengembalikan isi `processResults` tindakan.

Langkah 4: Buat halaman dan komponen

Di App Studio, setiap halaman mewakili layar antarmuka pengguna (UI) aplikasi yang akan berinteraksi dengan pengguna Anda. Dalam halaman ini, Anda dapat menambahkan berbagai komponen seperti tabel, formulir, tombol, dan lainnya untuk membuat tata letak dan fungsionalitas yang diinginkan.

Ganti nama halaman default

Aplikasi ringkasan teks dalam tutorial ini hanya akan berisi satu halaman. Aplikasi yang baru dibuat dilengkapi dengan halaman default, jadi Anda akan mengganti namanya alih-alih menambahkannya.

Untuk mengganti nama halaman default

1. Di menu navigasi bilah atas, pilih Halaman.
2. Di panel sisi kiri, pilih Page1 dan pilih panel Properties di panel sisi kanan.
3. Pilih ikon pensil, masukkan**`TextSummarizationTool`**, dan tekan Enter.
4. Di label Navigasi masukkan**`TextSummarizationTool`**.

Tambahkan komponen ke halaman

Untuk tutorial ini, aplikasi text summarizer memiliki satu halaman yang berisi komponen-komponen berikut:

- Komponen input Teks yang digunakan pengguna akhir untuk memasukkan prompt yang akan diringkas.
- Komponen Tombol yang digunakan untuk mengirim prompt ke Amazon Bedrock.
- Komponen area Teks yang menampilkan ringkasan dari Amazon Bedrock.

Tambahkan komponen input Teks ke halaman yang akan digunakan pengguna untuk memasukkan prompt teks yang akan diringkas.

Untuk menambahkan komponen input teks

1. Di panel Components sebelah kanan, cari komponen input Teks dan seret ke kanvas.
2. Pilih input teks di kanvas untuk memilihnya.
3. Di panel Properties sisi kanan, perbarui pengaturan berikut:
    a. Pilih ikon pensil untuk mengganti nama input teks menjadi**`inputPrompt`**.
    b. Di Label, masukkan**`Prompt`**.
    c. Di Placeholder, masukkan. **`Enter text to be summarized`**

Sekarang, tambahkan komponen Tombol yang akan dipilih pengguna untuk mengirim prompt ke Amazon Bedrock.

Untuk menambahkan komponen tombol

1. Di panel Components sebelah kanan, cari komponen Button dan seret ke kanvas.
2. Pilih tombol di kanvas untuk memilihnya.
3. Di panel Properties sisi kanan, perbarui pengaturan berikut:
    a. Pilih ikon pensil untuk mengganti nama tombol menjadi**`sendButton`**.
    b. Di Label Tombol, masukkan**`Send`**.

Sekarang, tambahkan komponen area Teks yang akan menampilkan ringkasan yang dikembalikan oleh Amazon Bedrock.

Untuk menambahkan komponen area teks

1. Di panel Components sebelah kanan, cari komponen area Teks dan seret ke kanvas.
2. Pilih area teks di kanvas untuk memilihnya.
3. Di panel Properties sisi kanan, perbarui pengaturan berikut:
   a. Pilih ikon pensil untuk mengganti nama tombol menjadi**`textSummary`**.
   b. Di Label, masukkan**`Summary`**.

Konfigurasikan komponen halaman

Sekarang aplikasi berisi halaman dengan komponen, langkah selanjutnya adalah mengonfigurasi komponen untuk melakukan perilaku yang sesuai. Untuk mengonfigurasi komponen, seperti tombol, untuk mengambil tindakan saat berinteraksi dengannya, Anda harus menambahkan pemicu ke dalamnya. Untuk aplikasi dalam tutorial ini, Anda akan menambahkan dua pemicu ke `sendButton` tombol untuk melakukan hal berikut:

- Pemicu pertama mengirimkan teks dalam `textPrompt` komponen ke Amazon Bedrock untuk dianalisis.
- Pemicu kedua menampilkan ringkasan yang dikembalikan dari Amazon Bedrock di `textSummary` komponen.

Untuk menambahkan pemicu yang mengirimkan prompt ke Amazon Bedrock

1. Pilih tombol di kanvas untuk memilihnya.
2. Di panel Properties sisi kanan, di bagian Pemicu, pilih + Tambah.
3. Pilih Invoke Automation.
4. Pilih pemicu InvokeAutomation1 yang dibuat untuk mengonfigurasinya.
5. Di Nama Tindakan, masukkan**`invokeBedrockAutomation`**.
6. Di Invoke Automation, pilih InvokeBedrockotomatisasi yang dibuat sebelumnya.
7. Di kotak parameter, dalam parameter input yang dibuat sebelumnya, masukkan**`{{ui.inputPrompt.value}}`**, yang melewati konten dalam komponen input `inputPrompt` teks.

8. Pilih panah kiri di bagian atas panel untuk kembali ke menu properti komponen.

Sekarang, Anda telah mengonfigurasi pemicu yang memanggil otomatisasi untuk mengirim permintaan ke Amazon Bedrock saat tombol diklik, langkah selanjutnya adalah mengonfigurasi pemicu kedua yang menampilkan hasil dalam komponen. `textSummary`

Untuk menambahkan pemicu yang menampilkan hasil Amazon Bedrock di komponen area teks

1. Di panel Properti sisi kanan tombol, di bagian Pemicu, pilih + Tambah.
2. Pilih Jalankan tindakan komponen.
3. Pilih pemicu Runcomponentaction1 yang dibuat untuk mengonfigurasinya.
4. Di Nama Tindakan, masukkan**`setTextSummary`**.
5. Di Komponen, pilih komponen TextSummary.
6. Dalam Tindakan, pilih Tetapkan nilai.
7. Di Tetapkan nilai ke, masukkan**`{{results.invokeBedrockAutomation}}`**.

Langkah 5: Publikasikan aplikasi ke lingkungan Pengujian

Biasanya, saat Anda sedang membangun aplikasi, praktik yang baik adalah mempratinjaunya untuk melihat tampilannya dan melakukan pengujian awal pada fungsinya. Namun, karena aplikasi tidak berinteraksi dengan layanan eksternal di lingkungan pratinjau, Anda akan memublikasikan aplikasi ke lingkungan Pengujian untuk dapat menguji permintaan pengiriman dan menerima tanggapan dari Amazon Bedrock.

Untuk memublikasikan aplikasi Anda ke lingkungan Pengujian

1. Di pojok kanan atas pembuat aplikasi, pilih Publikasikan.
2. Tambahkan deskripsi versi untuk lingkungan Pengujian.
3. Tinjau dan pilih kotak centang mengenai SLA.
4. Pilih Mulai. Penerbitan dapat memakan waktu hingga 15 menit.
5. (Opsional) Saat Anda siap, Anda dapat memberi orang lain akses dengan memilih Bagikan dan mengikuti petunjuknya. Untuk informasi selengkapnya tentang berbagi aplikasi App Studio, lihatBerbagi aplikasi yang diterbitkan.

Setelah menguji aplikasi Anda, pilih Publish lagi untuk mempromosikan aplikasi ke lingkungan Produksi. Perhatikan bahwa aplikasi di lingkungan Produksi tidak tersedia untuk pengguna akhir hingga aplikasi tersebut dibagikan. Untuk informasi selengkapnya tentang lingkungan aplikasi yang berbeda, lihatLingkungan aplikasi.

(Opsional) Bersihkan

Anda sekarang telah berhasil menyelesaikan tutorial dan membangun aplikasi ringkasan teks di App Studio dengan Amazon Bedrock. Anda dapat terus menggunakan aplikasi Anda, atau Anda dapat membersihkan sumber daya yang dibuat dalam tutorial ini. Daftar berikut berisi daftar sumber daya yang akan dibersihkan:

- Konektor Amazon Bedrock dibuat di App Studio. Untuk informasi selengkapnya, lihat Melihat, mengedit, dan menghapus konektor.
- Aplikasi peringkas teks di App Studio. Untuk informasi selengkapnya, lihat Menghapus aplikasi.
- Peran IAM dibuat di konsol IAM. Untuk informasi selengkapnya, lihat Menghapus peran atau profil instance di Panduan AWS Identity and Access Management Pengguna.
- Jika Anda meminta akses model untuk menggunakan Claude 3 Sonnet dan ingin mengembalikan akses, lihat Mengelola akses ke model foundation Amazon Bedrock di Panduan Pengguna Amazon Bedrock.

Berinteraksi dengan Amazon Simple Storage Service dengan komponen dan otomatisasi

Anda dapat menjalankan berbagai operasi Amazon S3 dari aplikasi App Studio. Misalnya, Anda dapat membuat panel admin sederhana untuk mengelola pengguna dan pesanan Anda dan menampilkan media Anda dari Amazon S3. Meskipun Anda dapat menjalankan operasi Amazon S3 apa pun menggunakan tindakan AWS Invoke, ada empat tindakan Amazon S3 khusus yang dapat Anda tambahkan ke otomatisasi di aplikasi untuk melakukan operasi umum pada bucket dan objek Amazon S3. Keempat tindakan dan operasinya adalah sebagai berikut:

- Put Object: Menggunakan `Amazon S3 PutObject` operasi untuk menambahkan objek bucket Amazon S3.
- Dapatkan Objek: Menggunakan `Amazon S3 GetObject` operasi untuk mengambil objek dari bucket Amazon S3.

- Daftar Objek: Menggunakan `Amazon S3 ListObjects` operasi untuk mencantumkan objek di bucket Amazon S3.
- Hapus Objek: Menggunakan `Amazon S3 DeleteObject` operasi untuk menghapus objek dari bucket Amazon S3.

Selain tindakan, ada komponen unggah S3 yang dapat Anda tambahkan ke halaman dalam aplikasi. Pengguna dapat menggunakan komponen ini untuk memilih file yang akan diunggah, dan komponen memanggil `Amazon S3 PutObject` untuk mengunggah file ke bucket dan folder yang dikonfigurasi. Tutorial ini akan menggunakan komponen ini sebagai pengganti tindakan otomatisasi Put Object mandiri. (Tindakan mandiri harus digunakan dalam skenario yang lebih kompleks yang melibatkan logika atau tindakan tambahan yang harus diambil sebelum atau setelah mengunggah.)

Prasyarat

Panduan ini mengasumsikan Anda telah menyelesaikan prasyarat berikut:

1. Membuat dan mengonfigurasi bucket Amazon S3, peran dan kebijakan IAM, serta konektor Amazon S3 agar berhasil mengintegrasikan Amazon S3 dengan App Studio. Untuk membuat konektor, Anda harus memiliki peran Administrator. Untuk informasi selengkapnya, lihat Connect ke Amazon Simple Storage Service (Amazon S3).

Buat aplikasi kosong

Buat aplikasi kosong untuk digunakan di seluruh panduan ini dengan melakukan langkah-langkah berikut.

Untuk membuat aplikasi kosong

1. Di panel navigasi, pilih Aplikasi saya.
2. Pilih + Buat aplikasi.
3. Di kotak dialog Buat aplikasi, beri nama aplikasi Anda, pilih Mulai dari awal, dan pilih Berikutnya.
4. Di kotak dialog Connect to existing data, pilih Lewati untuk membuat aplikasi.
5. Pilih Edit aplikasi untuk dibawa ke kanvas aplikasi baru Anda, di mana Anda dapat menggunakan komponen, otomatisasi, dan data untuk mengonfigurasi tampilan dan fungsi aplikasi Anda.

Buat halaman

Buat tiga halaman dalam aplikasi Anda untuk mengumpulkan atau menampilkan informasi.

Untuk membuat halaman

1. Jika perlu, pilih tab Halaman di bagian atas kanvas.
2. Di navigasi sebelah kiri, ada satu halaman yang dibuat dengan aplikasi Anda. Pilih + Tambahkan dua kali untuk membuat dua halaman lagi. Panel navigasi harus menampilkan tiga halaman total.
3. Perbarui nama halaman Page1 dengan melakukan langkah-langkah berikut:
   a. Pilih ikon elips dan pilih properti Halaman.
   b. Di menu Properties sebelah kanan, pilih ikon pensil untuk mengedit nama.
   c. Masuk **`FileList`** dan tekan Enter.
4. Ulangi langkah sebelumnya untuk memperbarui halaman kedua dan ketiga sebagai berikut:
   - Ubah nama Page2 menjadi. **`UploadFile`**
   - Ubah nama Page3 menjadi. **`FailUpload`**

Sekarang, aplikasi Anda harus memiliki tiga halaman bernama FileList, UploadFile, dan FailUpload, yang ditampilkan di panel Pages sebelah kiri.

Selanjutnya, Anda akan membuat dan mengonfigurasi otomatisasi yang berinteraksi dengan Amazon S3.

Buat dan konfigurasikan otomatisasi

Buat otomatisasi aplikasi Anda yang berinteraksi dengan Amazon S3. Gunakan prosedur berikut untuk membuat otomatisasi berikut:

- Otomatisasi GetFiles yang mencantumkan objek di bucket Amazon S3 Anda, yang akan digunakan untuk mengisi komponen tabel.
- Otomatisasi DeleteFile yang menghapus objek dari bucket Amazon S3 Anda, yang akan digunakan untuk menambahkan tombol hapus ke komponen tabel.
- Otomatisasi ViewFile yang mendapatkan objek dari bucket Amazon S3 Anda dan menampilkannya, yang akan digunakan untuk menampilkan detail selengkapnya tentang satu objek yang dipilih dari komponen tabel.

Buat **`getFiles`** otomatisasi

Buat otomatisasi yang akan mencantumkan file dalam bucket Amazon S3 tertentu.

1. Pilih tab Automations di bagian atas kanvas.
2. Pilih + Tambahkan otomatisasi.
3. Di panel sebelah kanan, pilih Properties.
4. Perbarui nama otomatisasi dengan memilih ikon pensil. Masuk **`getFiles`** dan tekan Enter.
5. Tambahkan tindakan objek Daftar dengan melakukan langkah-langkah berikut:
   a. Di panel sebelah kanan, pilih Tindakan.
   b. Pilih Daftar objek untuk menambahkan tindakan. Tindakan itu harus diberi nama`ListObjects1`.
6. Konfigurasikan tindakan dengan melakukan langkah-langkah berikut:
   a. Pilih tindakan dari kanvas untuk membuka menu Properties sebelah kanan.
   b. Untuk Konektor, pilih konektor Amazon S3 yang Anda buat dari prasyarat.
   c. Untuk Konfigurasi, masukkan teks berikut, ganti *`bucket_name`* dengan bucket yang Anda buat di prasyarat:

```
{
  "Bucket": "bucket_name",
  "Prefix": ""
}
```

> **Note**
>
> Anda dapat menggunakan `Prefix` bidang untuk membatasi respons terhadap objek yang dimulai dengan string yang ditentukan.

7. Output otomatisasi ini akan digunakan untuk mengisi komponen tabel dengan objek dari bucket Amazon S3 Anda. Namun, secara default, otomatisasi tidak membuat output. Konfigurasikan otomatisasi untuk membuat output otomatisasi dengan melakukan langkah-langkah berikut:
   a. Di navigasi sebelah kiri, pilih otomatisasi GetFiles.
   b. Di menu Properties sebelah kanan, di Output otomatisasi, pilih + Tambahkan output.

c. Untuk Output, masukkan**{{results.ListObjects1.Contents}}**. Ekspresi ini mengembalikan isi tindakan, dan sekarang dapat digunakan untuk mengisi komponen tabel.

Buat **deleteFile** otomatisasi

Buat otomatisasi yang menghapus objek dari bucket Amazon S3 tertentu.

1. Di panel Otomatisasi sebelah kiri, pilih + Tambah.
2. Pilih + Tambahkan otomatisasi.
3. Di panel sebelah kanan, pilih Properties.
4. Perbarui nama otomatisasi dengan memilih ikon pensil. Masuk **deleteFile** dan tekan Enter.
5. Tambahkan parameter otomatisasi, yang digunakan untuk meneruskan data ke otomatisasi, dengan melakukan langkah-langkah berikut:
   a. Di menu Properties sebelah kanan, di parameter Otomasi, pilih + Tambah.
   b. Pilih ikon pensil untuk mengedit parameter otomatisasi. Perbarui nama parameter ke **fileName** dan tekan Enter.
6. Tambahkan tindakan Hapus objek dengan melakukan langkah-langkah berikut:
   a. Di panel sebelah kanan, pilih Tindakan.
   b. Pilih Hapus objek untuk menambahkan tindakan. Tindakan itu harus diberi nama`DeleteObject1`.
7. Konfigurasikan tindakan dengan melakukan langkah-langkah berikut:
   a. Pilih tindakan dari kanvas untuk membuka menu Properties sebelah kanan.
   b. Untuk Konektor, pilih konektor Amazon S3 yang Anda buat dari prasyarat.
   c. Untuk Konfigurasi, masukkan teks berikut, ganti *bucket_name* dengan bucket yang Anda buat di prasyarat:

```
{
  "Bucket": "bucket_name",
  "Key": params.fileName
}
```

Buat **`viewFile`** otomatisasi

Buat otomatisasi yang mengambil satu objek dari bucket Amazon S3 tertentu. Nanti, Anda akan mengonfigurasi otomatisasi ini dengan komponen penampil file untuk menampilkan objek.

1. Di panel Otomatisasi sebelah kiri, pilih + Tambah.
2. Pilih + Tambahkan otomatisasi.
3. Di panel sebelah kanan, pilih Properties.
4. Perbarui nama otomatisasi dengan memilih ikon pensil. Masuk **`viewFile`** dan tekan Enter.
5. Tambahkan parameter otomatisasi, yang digunakan untuk meneruskan data ke otomatisasi, dengan melakukan langkah-langkah berikut:
   a. Di menu Properties sebelah kanan, di parameter Otomasi, pilih + Tambah.
   b. Pilih ikon pensil untuk mengedit parameter otomatisasi. Perbarui nama parameter ke **`fileName`** dan tekan Enter.
6. Tambahkan tindakan Get object dengan melakukan langkah-langkah berikut:
   a. Di panel sebelah kanan, pilih Tindakan.
   b. Pilih Dapatkan objek untuk menambahkan tindakan. Tindakan itu harus diberi nama`GetObject1`.
7. Konfigurasikan tindakan dengan melakukan langkah-langkah berikut:
   a. Pilih tindakan dari kanvas untuk membuka menu Properties sebelah kanan.
   b. Untuk Konektor, pilih konektor Amazon S3 yang Anda buat dari prasyarat.
   c. Untuk Konfigurasi, masukkan teks berikut, ganti *`bucket_name`* dengan bucket yang Anda buat di prasyarat:

```
{
  "Bucket": "bucket_name",
  "Key": params.fileName
}
```

8. Secara default, otomatisasi tidak membuat output. Konfigurasikan otomatisasi untuk membuat output otomatisasi dengan melakukan langkah-langkah berikut:
   a. Di navigasi sebelah kiri, pilih otomatisasi ViewFile.
   b. Di menu Properties sebelah kanan, di Output otomatisasi, pilih + Tambahkan output.

c. Untuk Output, masukkan**{{results.GetObject1.Body.transformToWebStream()}}**. Ekspresi ini mengembalikan isi dari tindakan.

> **Note**
>
> Anda dapat membaca tanggapan dengan `S3 GetObject` cara-cara berikut:
>
> - `transformToWebStream`: Mengembalikan aliran, yang harus dikonsumsi untuk mengambil data. Jika digunakan sebagai output otomatisasi, otomatisasi menangani ini, dan output dapat digunakan sebagai sumber data dari gambar atau komponen penampil PDF. Ini juga dapat digunakan sebagai input ke operasi lain, seperti`S3 PutObject`.
> - `transformToString`: Mengembalikan data mentah otomatisasi, dan harus digunakan dalam JavaScript tindakan jika file Anda berisi konten teks, seperti data JSON. Harus ditunggu, misalnya: `await results.GetObject1.Body.transformToString();`
> - `transformToByteArray`: Mengembalikan array 8-bit integer unsigned. Respons ini melayani tujuan array byte, yang memungkinkan penyimpanan dan manipulasi data biner. Harus ditunggu, misalnya: `await results.GetObject1.Body.transformToByteArray();`

Selanjutnya, Anda akan menambahkan komponen ke halaman yang Anda buat sebelumnya, dan mengonfigurasinya dengan otomatisasi Anda sehingga pengguna dapat menggunakan aplikasi Anda untuk melihat dan menghapus file.

Tambahkan dan konfigurasikan komponen halaman

Sekarang setelah Anda membuat otomatisasi yang menentukan logika bisnis dan fungsionalitas aplikasi Anda, Anda akan membuat komponen dan menghubungkan keduanya.

Tambahkan komponen ke FileListhalaman

FileListHalaman yang Anda buat sebelumnya akan digunakan untuk menampilkan daftar file di bucket Amazon S3 yang dikonfigurasi dan detail lebih lanjut tentang file apa pun yang dipilih dari daftar. Untuk melakukan itu, Anda akan melakukan hal berikut:

1. Buat komponen tabel untuk menampilkan daftar file. Anda akan mengonfigurasi baris tabel yang akan diisi dengan output otomatisasi GetFiles yang sebelumnya Anda buat.
2. Buat komponen penampil PDF untuk menampilkan satu PDF. Anda akan mengonfigurasi komponen untuk melihat file yang dipilih dari tabel, menggunakan otomatisasi ViewFile yang sebelumnya Anda buat untuk mengambil file dari bucket Anda.

Untuk menambahkan komponen ke FileListhalaman

1. Pilih tab Pages di bagian atas kanvas.
2. Di panel Pages sebelah kiri, pilih halaman. FileList
3. Di halaman Components sebelah kanan, temukan komponen Table dan seret ke tengah kanvas.
4. Pilih komponen tabel yang baru saja Anda tambahkan ke halaman.
5. Di menu Properties sebelah kanan, pilih dropdown Source dan pilih Automation.
6. Pilih dropdown Otomasi dan pilih otomatisasi GetFiles. Tabel akan menggunakan output otomatisasi GetFiles sebagai isinya.
7. Tambahkan kolom yang akan diisi dengan nama file.
    a. Di menu Properties sebelah kanan, di sebelah Kolom, pilih + Tambah.
    b. Pilih ikon panah di sebelah kanan kolom Column1 yang baru saja ditambahkan.
    c. Untuk label Kolom, ganti nama kolom menjadi**`Filename`**.
    d. Untuk Nilai, masukkan **`{{currentRow.Key}}`**.
    e. Pilih ikon panah di bagian atas panel untuk kembali ke panel Properties utama.
8. Tambahkan tindakan tabel untuk menghapus file berturut-turut.
    a. Di menu Properties sebelah kanan, di samping Actions, pilih + Add.
    b. Di Actions, ganti nama Button menjadi**`Delete`**.
    c. Pilih ikon panah di sebelah kanan tindakan Hapus yang baru saja diganti namanya.
    d. Di On click, pilih + Add action dan pilih Invoke automation.
    e. Pilih tindakan yang ditambahkan untuk mengonfigurasinya.
    f. Untuk nama Action, masukkan**`DeleteRecord`**.
    g. Di Invoke automation, pilih**`deleteFile`**.
    h. Di kotak teks parameter, masukkan**`{{currentRow.Key}}`**.
    i. Untuk Nilai, masukkan **`{{currentRow.Key}}`**.

9. Di panel sebelah kanan, pilih Komponen untuk melihat menu komponen. Ada dua pilihan untuk menampilkan file:

   - Penampil gambar untuk melihat file dengan `.png`, `.jpeg`, atau `.jpg` ekstensi.
   - Komponen penampil PDF untuk melihat file PDF.

   Dalam tutorial ini, Anda akan menambahkan dan mengkonfigurasi komponen penampil PDF.

10. Tambahkan komponen penampil PDF.

    a. Di halaman Komponen sebelah kanan, temukan komponen penampil PDF dan seret ke kanvas, di bawah komponen tabel.

    b. Pilih komponen penampil PDF yang baru saja ditambahkan.

    c. Di menu Properties sebelah kanan, pilih dropdown Source dan pilih Automation.

    d. Pilih dropdown Otomasi dan pilih otomatisasi ViewFile. Tabel akan menggunakan output dari otomatisasi ViewFile sebagai isinya.

    e. Di kotak teks parameter, masukkan **`{{ui.table1.selectedRow["Filename"]}}`**.

    f. Di panel sebelah kanan, ada juga bidang Nama file. Nilai bidang ini digunakan sebagai header untuk komponen penampil PDF. Masukkan teks yang sama dengan langkah sebelumnya: **`{{ui.table1.selectedRow["Filename"]}}`**.

Tambahkan komponen ke UploadFilehalaman

UploadFileHalaman akan berisi pemilih file yang dapat digunakan untuk memilih dan mengunggah file ke bucket Amazon S3 yang dikonfigurasi. Anda akan menambahkan komponen unggah S3 ke halaman, yang dapat digunakan pengguna untuk memilih dan mengunggah file.

1. Di panel Pages sebelah kiri, pilih halaman. UploadFile
2. Di halaman Components sebelah kanan, temukan komponen upload S3 dan seret ke tengah kanvas.
3. Pilih komponen upload S3 yang baru saja Anda tambahkan ke halaman.
4. Di menu Properties sebelah kanan, konfigurasikan komponen:

   a. Di dropdown Connector, pilih konektor Amazon S3 yang dibuat dalam prasyarat.

   b. Untuk Bucket, masukkan nama bucket Amazon S3 Anda.

   c. Untuk nama File, masukkan **`{{ui.s3Upload1.files[0]?.nameWithExtension}}`**.

d. Untuk ukuran file Max, masukkan **5** di kotak teks, dan pastikan yang **MB** dipilih di dropdown.

e. Di bagian Pemicu, tambahkan tindakan yang berjalan setelah unggahan berhasil atau tidak berhasil dengan melakukan langkah-langkah berikut:

   Untuk menambahkan tindakan yang berjalan setelah unggahan berhasil:

   1. Di On success, pilih + Add action dan pilih Navigate.
   2. Pilih tindakan yang ditambahkan untuk mengonfigurasinya.
   3. Untuk jenis Navigasi, pilih Halaman.
   4. Untuk Navigasi ke, pilih**FileList**.
   5. Pilih ikon panah di bagian atas panel untuk kembali ke panel Properties utama.

   Untuk menambahkan tindakan yang berjalan setelah unggahan gagal:

   1. Di Kegagalan, pilih + Tambahkan tindakan dan pilih Navigasi.
   2. Pilih tindakan yang ditambahkan untuk mengonfigurasinya.
   3. Untuk jenis Navigasi, pilih Halaman.
   4. Untuk Navigasi ke, pilih**FailUpload**.
   5. Pilih ikon panah di bagian atas panel untuk kembali ke panel Properties utama.

Tambahkan komponen ke FailUploadhalaman

FailUploadHalaman ini adalah halaman sederhana yang berisi kotak teks yang memberi tahu pengguna bahwa unggahan mereka gagal.

1. Di panel Pages sebelah kiri, pilih halaman. FailUpload
2. Di halaman Components sebelah kanan, temukan komponen Text dan seret ke tengah kanvas.
3. Pilih komponen teks yang baru saja Anda tambahkan ke halaman.
4. Di menu Properties sebelah kanan, di Value, masukkan**Failed to upload, try again**.

Memperbarui setelan keamanan aplikasi

Setiap aplikasi di App Studio memiliki pengaturan keamanan konten yang dapat Anda gunakan untuk membatasi media atau sumber daya eksternal, atau domain mana di Amazon S3 yang dapat Anda unggah objek. Pengaturan default adalah memblokir semua domain. Untuk mengunggah objek ke

Amazon S3 dari aplikasi Anda, Anda harus memperbarui pengaturan untuk mengizinkan domain yang ingin Anda unggah objek.

Untuk mengizinkan domain untuk mengunggah objek ke Amazon S3

1. Pilih tab Pengaturan aplikasi.
2. Pilih tab Pengaturan Keamanan Konten.
3. Untuk Connect source, pilih Izinkan semua koneksi.
4. Pilih Simpan.

Langkah selanjutnya: Pratinjau dan publikasikan aplikasi untuk pengujian

Aplikasi ini sekarang siap untuk pengujian. Untuk informasi selengkapnya tentang pratinjau dan penerbitan aplikasi, lihatMempratinjau, menerbitkan, dan berbagi aplikasi.

Memanggil fungsi Lambda di aplikasi App Studio

Tutorial ini menunjukkan cara menghubungkan App Studio ke Lambda dan menjalankan fungsi Lambda dari aplikasi Anda.

Prasyarat

Panduan ini mengasumsikan Anda telah menyelesaikan prasyarat berikut:

1. Membuat aplikasi App Studio. Jika Anda tidak memilikinya, Anda dapat membuat aplikasi kosong untuk digunakan dalam tutorial. Untuk informasi selengkapnya, lihat Membuat aplikasi.

> **Note**
>
> Meskipun Anda tidak memerlukan fungsi Lambda untuk mengikuti tutorial ini dan mempelajari cara mengonfigurasinya, mungkin bermanfaat jika memilikinya untuk memastikan Anda telah mengonfigurasi aplikasi dengan benar. Tutorial ini tidak berisi informasi tentang membuat fungsi Lambda. untuk informasi lebih lanjut, lihat Panduan Pengembang AWS Lambda .

Buat konektor Lambda

Untuk menggunakan fungsi Lambda di aplikasi App Studio, Anda harus menggunakan konektor untuk menghubungkan App Studio ke Lambda untuk menyediakan akses ke fungsi Anda. Anda harus menjadi Administrator untuk membuat konektor di App Studio. Untuk informasi selengkapnya tentang membuat konektor Lambda, termasuk langkah-langkah untuk membuatnya, lihat. Connect ke AWS Lambda

Buat dan konfigurasikan otomatisasi

Otomatisasi digunakan untuk menentukan logika aplikasi Anda dan terdiri dari tindakan. Untuk menjalankan fungsi Lambda di aplikasi, Anda terlebih dahulu menambahkan dan mengonfigurasi tindakan Invoke Lambda ke otomatisasi. Gunakan langkah-langkah berikut untuk membuat otomatisasi dan menambahkan tindakan Invoke Lambda ke dalamnya.

1. Saat mengedit aplikasi Anda, pilih tab Automations.
2. Pilih + Tambahkan otomatisasi.
3. Di menu Tindakan sebelah kanan, pilih Panggil Lambda untuk menambahkan langkah ke otomatisasi Anda.
4. Pilih langkah Lambda baru di kanvas untuk melihat dan mengkonfigurasi propertinya.
5. Di menu Properties sebelah kanan, konfigurasikan langkah dengan melakukan langkah-langkah berikut:

    a. Di Connector, pilih konektor yang dibuat untuk menghubungkan App Studio ke fungsi Lambda Anda.

    b. Di nama Fungsi, masukkan nama fungsi Lambda Anda.

    c. Dalam acara Fungsi, masukkan acara yang akan diteruskan ke fungsi Lambda. Beberapa contoh kasus penggunaan umum disediakan dalam daftar berikut:

    - Melewati nilai parameter otomatisasi, seperti nama file atau string lainnya: `varName: params.paramName`
    - Melewati hasil dari tindakan sebelumnya: `varName: results.actionName1.data[0].fieldName`
    - Jika Anda menambahkan tindakan Invoke Lambda di dalam tindakan Loop, Anda dapat mengirim kolom dari setiap item iterasi yang mirip dengan parameter: `varName: currentItem.fieldName`

d. Bidang keluaran Mocked dapat digunakan untuk menyediakan output tiruan untuk menguji aplikasi saat melihat pratinjau, di mana konektor tidak aktif.

Konfigurasikan elemen UI untuk menjalankan otomatisasi

Sekarang setelah Anda memiliki otomatisasi yang dikonfigurasi dengan tindakan untuk menjalankan fungsi Lambda Anda, Anda dapat mengonfigurasi elemen UI untuk menjalankan otomatisasi. Dalam tutorial ini, Anda akan membuat tombol yang menjalankan otomatisasi saat diklik.

> **Tip**
>
> Anda juga dapat menjalankan otomatisasi dari otomatisasi lain dengan tindakan otomatisasi Invoke.

Untuk menjalankan otomatisasi Anda dari sebuah tombol

1. Saat mengedit aplikasi Anda, pilih tab Pages.
2. Di menu sebelah kanan, pilih komponen Tombol untuk menambahkan tombol ke halaman.
3. Pilih tombol baru untuk mengkonfigurasinya.
4. Di menu Properties sebelah kanan, di Triggers, pilih + Add dan pilih Invoke automation.
5. Pilih pemicu pemanggilan otomatisasi baru untuk mengonfigurasinya.
6. Di otomatisasi Invoke, pilih otomatisasi yang memanggil fungsi Lambda Anda dan konfigurasikan parameter apa pun yang ingin Anda kirim ke otomatisasi.

Sekarang, setiap pengguna yang memilih tombol ini di aplikasi Anda akan menyebabkan otomatisasi yang dikonfigurasi berjalan.

Langkah selanjutnya: Pratinjau dan publikasikan aplikasi untuk pengujian

Aplikasi Anda sekarang siap untuk pengujian. Saat mempratinjau aplikasi di lingkungan Pengembangan, konektor tidak aktif, sehingga Anda tidak dapat menguji otomatisasi saat melihat pratinjau karena menggunakan konektor untuk disambungkan. AWS Lambda Untuk menguji fungsionalitas aplikasi yang bergantung pada konektor, Anda harus memublikasikan aplikasi ke lingkungan Pengujian. Untuk informasi selengkapnya tentang pratinjau dan penerbitan aplikasi, lihatMempratinjau, menerbitkan, dan berbagi aplikasi.

Membangun aplikasi App Studio Anda dengan AI generatif

AWS App Studio menyediakan kemampuan AI generatif terintegrasi untuk mempercepat pengembangan dan merampingkan tugas-tugas umum. Anda dapat memanfaatkan AI generatif untuk menghasilkan dan mengedit aplikasi, model data, data sampel, dan bahkan mendapatkan bantuan kontekstual saat membuat aplikasi.

Menghasilkan aplikasi Anda

Untuk memulai yang dipercepat, Anda dapat menghasilkan seluruh aplikasi menggunakan perintah bahasa alami yang didukung oleh AI. Kemampuan ini memungkinkan Anda untuk menggambarkan fungsionalitas aplikasi yang Anda inginkan, dan AI akan secara otomatis membangun model data, antarmuka pengguna, alur kerja, dan konektor. Untuk informasi selengkapnya tentang membuat aplikasi dengan AI, lihatMembuat aplikasi.

Membangun atau mengedit aplikasi Anda

Saat mengedit aplikasi, Anda dapat menggunakan obrolan untuk menjelaskan perubahan yang ingin Anda buat dan aplikasi Anda diperbarui secara otomatis. Anda dapat memilih dari petunjuk sampel yang ada atau memasukkan prompt Anda sendiri. Obrolan dapat digunakan untuk menambah, mengedit, dan menghapus komponen yang didukung, dan juga membuat dan mengonfigurasi otomatisasi dan tindakan. Gunakan prosedur berikut untuk menggunakan AI untuk mengedit atau membangun aplikasi Anda.

Untuk mengedit aplikasi Anda dengan AI

1. Jika perlu, edit aplikasi Anda untuk menavigasi ke studio aplikasi.
2. (Opsional) Pilih halaman atau komponen yang ingin Anda edit dengan AI.
3. Pilih Build with AI di pojok kiri bawah untuk membuka obrolan.
4. Masukkan perubahan yang ingin Anda buat, atau pilih dari contoh petunjuk.
5. Tinjau perubahan yang akan dilakukan. Jika Anda ingin perubahan dilakukan, pilih Konfirmasi. Jika tidak, masukkan prompt lain.
6. Tinjau ringkasan perubahan.

Menghasilkan model data Anda

Anda dapat secara otomatis membuat entitas dengan bidang, tipe data, dan tindakan data berdasarkan nama entitas yang disediakan. Untuk informasi selengkapnya tentang membuat entitas, termasuk membuat entitas menggunakan GenAi, lihatMembuat entitas di aplikasi App Studio.

Anda juga dapat memperbarui entitas yang ada dengan cara berikut:

- Tambahkan lebih banyak bidang ke entitas. Untuk informasi selengkapnya, lihat Menambahkan, mengedit, atau menghapus bidang entitas.
- Tambahkan tindakan data ke entitas. Untuk informasi selengkapnya, lihat Membuat tindakan data.

Menghasilkan data sampel

Anda dapat menghasilkan data sampel untuk entitas berdasarkan bidang entitas. Ini berguna untuk menguji aplikasi Anda sebelum menghubungkan sumber data eksternal, atau menguji aplikasi Anda di lingkungan Pengembangan, yang tidak berkomunikasi dengan sumber data eksternal. Untuk informasi selengkapnya, lihat Menambahkan atau menghapus data sampel.

Setelah memublikasikan aplikasi ke Pengujian atau Produksi, sumber dan konektor data langsung Anda akan digunakan di lingkungan tersebut.

Mengkonfigurasi tindakan untuk layanan AWS

Saat mengintegrasikan dengan AWS layanan seperti Amazon Simple Email Service, Anda dapat menggunakan AI untuk menghasilkan contoh konfigurasi dengan bidang yang telah diisi sebelumnya berdasarkan layanan yang dipilih. Untuk mencobanya, Di menu Properties dari tindakan otomatisasi AWS Invoke, perluas bidang Konfigurasi dengan memilih panah dua sisi. Kemudian, pilih Hasilkan konfigurasi sampel.

Tanggapan mengejek

Anda dapat menghasilkan respons yang diejek untuk tindakan AWS layanan. Ini berguna untuk menguji aplikasi Anda di lingkungan Pengembangan, yang tidak berkomunikasi dengan sumber data eksternal.

Meminta bantuan AI saat membangun

Di studio aplikasi, Anda akan menemukan tombol Minta AI untuk bantuan pada sumber daya atau properti yang didukung. Gunakan ini untuk mendapatkan saran kontekstual, dokumentasi, dan panduan yang terkait dengan tampilan saat ini atau komponen yang dipilih. Ajukan pertanyaan umum tentang App Studio, praktik terbaik pembuatan aplikasi, atau kasus penggunaan aplikasi khusus Anda untuk menerima informasi dan rekomendasi yang disesuaikan.

Membuat, mengedit, dan menghapus aplikasi

Daftar Isi

Melihat aplikasi

Gunakan prosedur berikut untuk melihat aplikasi di App Studio.

Untuk melihat aplikasi

1. Di panel navigasi, pilih Aplikasi saya di bagian Build. Anda akan dibawa ke halaman yang menampilkan daftar aplikasi yang dapat Anda akses.
2. Pada halaman Aplikasi saya, tabel menampilkan daftar aplikasi Anda dengan rincian berikut:

   - Nama aplikasi: Nama aplikasi.
   - Status: Status aplikasi. Nilai yang mungkin adalah:
     - Draf: Aplikasi belum dipublikasikan.
     - Diterbitkan: Aplikasi telah diterbitkan.
   - Terakhir diperbarui: Tanggal aplikasi terakhir diedit.
   - Peran: Peran Anda dalam kaitannya dengan aplikasi. Nilai yang mungkin adalah:
     - Pemilik: Pemilik aplikasi memiliki semua akses dan izin ke aplikasi.

- Pemilik bersama: Pemilik bersama aplikasi memiliki akses yang mirip dengan pemilik aplikasi.
- Hanya edit: Pengguna dengan akses Edit-only ke aplikasi dapat mengedit aplikasi, tetapi tidak dapat mengundang pembangun lain ke aplikasi, mempublikasikan aplikasi ke produksi, menghapus aplikasi, atau mengkloning aplikasi.

3. Anda dapat memilih panah di kolom Tindakan untuk membuka menu tindakan untuk aplikasi itu dengan opsi berikut:

- Sunting: Membuka aplikasi untuk diedit di studio pembuat. Pengeditan hanya tersedia untuk pemilik aplikasi dan editor.
- Bagikan: Membuka kotak dialog tempat tautan aplikasi dapat disalin. Berbagi hanya tersedia pada aplikasi yang dipublikasikan.
- Lihat: Membuka aplikasi yang sedang berjalan. Melihat hanya tersedia pada aplikasi yang dipublikasikan.
- Duplikat: Buat aplikasi lain dengan komponen, otomatisasi, dan entitas yang sama dengan aplikasi saat ini.
- Ganti nama: Berikan nama baru untuk aplikasi.
- Hapus: Menghapus aplikasi. Penghapusan hanya tersedia untuk pemilik aplikasi dan admin.

Membuat aplikasi

Gunakan prosedur berikut untuk membuat aplikasi di App Studio.

Untuk membuat aplikasi

1. Di panel navigasi, pilih Aplikasi saya di bagian Build untuk menavigasi ke daftar aplikasi Anda.
2. Pilih + Buat aplikasi.
3. Di kotak dialog Create app, beri nama aplikasi Anda dan pilih salah satu metode pembuatan aplikasi berikut:
    - Buat aplikasi dengan AI: Pilih opsi ini untuk mendeskripsikan aplikasi Anda dengan bahasa alami, dan minta AI menghasilkan aplikasi dan sumber dayanya untuk Anda.
    - Mulai dari awal: Pilih opsi ini untuk mulai membangun dari aplikasi kosong.
4. Pilih Berikutnya.
5. Jika Anda memilih Menghasilkan aplikasi dengan AI:

a. Di kotak dialog Connect to existing data, tambahkan sumber data yang ada ke aplikasi Anda dengan memilih Connector yang menyediakan akses App Studio ke sumber data, lalu pilih Tablse, dan pilih Berikutnya. Menambahkan sumber data di sini membantu AI menghasilkan aplikasi yang dioptimalkan untuk Anda. Anda dapat melewati langkah ini dan menambahkan sumber data nanti dengan memilih Lewati.

b. Setelah penundaan singkat (beberapa menit), Anda akan dibawa ke halaman Hasilkan aplikasi menggunakan AI, tempat Anda dapat mendeskripsikan aplikasi yang ingin Anda buat.

c. Anda dapat mulai mendeskripsikan aplikasi Anda dalam obrolan, atau Anda dapat memilih dan menyesuaikan contoh prompt yang disediakan.

d. Setelah prompt Anda dianalisis, tinjau persyaratan dan ikhtisar aplikasi. Gunakan obrolan untuk meminta perubahan apa pun, atau pilih Mulai dari awal untuk memulai dari prompt kosong.

e. Saat siap, pilih Hasilkan aplikasi.

f. Setelah dibuat, pratinjau aplikasi Anda di tab lain dengan memilih aplikasi Pratinjau. Saat Anda siap untuk mulai mengedit, Anda dapat memilih Edit aplikasi. Jelajahi halaman, otomatisasi, dan data aplikasi Anda untuk membiasakan diri dengannya. Tinjau kesalahan atau peringatan apa pun di panel debug bawah. Untuk mempelajari cara membuat aplikasi menggunakan AI, lihatTutorial: Menghasilkan aplikasi menggunakan AI. Untuk informasi umum tentang cara kerja pembuatan di App Studio, lihatCara kerja AWS App Studio.

6. Jika Anda memilih Mulai dari awal:

a. Di kotak dialog Connect to existing data, tambahkan sumber data yang ada ke aplikasi Anda dengan memilih Connector yang menyediakan akses App Studio ke sumber data, lalu pilih Tablse, dan pilih Berikutnya. Anda dapat melewati langkah ini dan menambahkan sumber data nanti dengan memilih Lewati.

b. Setelah aplikasi Anda dibuat, pilih Edit aplikasi untuk mulai mengedit aplikasi Anda. Untuk mempelajari tentang membangun dari aplikasi kosong, lihatTutorial: Mulai membangun dari aplikasi kosong. Untuk informasi umum tentang cara kerja pembuatan di App Studio, lihatCara kerja AWS App Studio.

Mengedit aplikasi

Gunakan prosedur berikut untuk mengedit aplikasi di App Studio.

Untuk mengedit aplikasi

1. Di panel navigasi, pilih Aplikasi saya di bagian Build. Anda akan dibawa ke halaman yang menampilkan daftar aplikasi yang dapat Anda akses.
2. Pilih dropdown di kolom Actions dari aplikasi yang ingin Anda edit.
3. Untuk mengganti nama aplikasi, pilih Ganti nama, beri nama baru pada aplikasi Anda, dan pilih Ganti nama.
4. Untuk mengedit aplikasi, pilih Edit. Ini akan membawa Anda ke studio aplikasi di mana Anda dapat menggunakan komponen, otomatisasi, dan data untuk mengonfigurasi tampilan dan fungsi aplikasi Anda. Untuk informasi tentang membangun aplikasi, lihatMemulai dengan AWS App Studio.

Pengaturan aplikasi

Di studio aplikasi, Anda dapat melihat dan memperbarui pengaturan aplikasi berikut.

Navigasi aplikasi

Secara default, App Studio menampilkan semua halaman dalam navigasi aplikasi aplikasi yang dipublikasikan atau saat melihat pratinjau aplikasi. Anda dapat menyusun ulang halaman atau menghapus halaman dari navigasi di bagian Navigasi aplikasi, yang berisi pengaturan berikut:

- Toggle Tampilkan navigasi untuk halaman ini menentukan apakah pengguna aplikasi dapat menavigasi ke halaman yang ditentukan di aplikasi Anda.
- Di Halaman Beranda, pilih halaman yang ingin dinavigasi oleh pengguna aplikasi saat pertama kali mengakses aplikasi Anda dari menu tarik-turun.
- Di Halaman lain, pilih apakah halaman dapat dinavigasi, dan urutan mana yang ditampilkan di menu navigasi aplikasi.

Menghapus aplikasi

Gunakan prosedur berikut untuk menghapus aplikasi di App Studio.

Untuk menghapus aplikasi

1. Di panel navigasi, pilih Aplikasi saya di bagian Build. Anda akan dibawa ke halaman yang menampilkan daftar aplikasi yang dapat Anda akses.

2. Pilih dropdown di kolom Actions dari aplikasi yang ingin Anda hapus.
3. Pilih Hapus.
4. Di kotak dialog Hapus aplikasi, tinjau informasi tentang menghapus aplikasi dengan cermat. Jika Anda ingin menghapus aplikasi, pilih Hapus.

Mempratinjau, menerbitkan, dan berbagi aplikasi

Topik

Pratinjau aplikasi

Anda dapat melihat pratinjau aplikasi di App Studio untuk melihat bagaimana mereka akan muncul kepada pengguna dan juga menguji fungsinya dengan menggunakannya dan memeriksa log di panel debug.

Lingkungan pratinjau aplikasi tidak mendukung menampilkan data langsung atau koneksi dengan sumber daya eksternal dengan konektor, seperti sumber data. Untuk menguji fungsionalitas di lingkungan pratinjau, Anda dapat menggunakan keluaran tiruan dalam otomatisasi dan data sampel dalam entitas. Untuk melihat aplikasi Anda dengan data real-time, Anda harus mempublikasikan aplikasi Anda. Untuk informasi selengkapnya, lihat Penerbitan aplikasi.

Lingkungan pratinjau atau pengembangan tidak memperbarui aplikasi yang diterbitkan di lingkungan lain. Jika aplikasi belum dipublikasikan, pengguna tidak akan dapat mengaksesnya sampai diterbitkan dan dibagikan. Jika aplikasi telah diterbitkan dan dibagikan, pengguna masih akan mengakses versi yang telah diterbitkan, dan bukan versi yang digunakan dalam lingkungan pratinjau.

Untuk melihat pratinjau aplikasi Anda

1. Jika perlu, navigasikan ke studio aplikasi aplikasi yang ingin Anda pratinjau:
    a. Di panel navigasi, pilih Aplikasi saya di bagian Build.
    b. Pilih Edit untuk aplikasi.

2. Pilih Pratinjau untuk membuka lingkungan pratinjau untuk aplikasi.
3. (Opsional) Perluas panel debug dengan memilih header di dekat bagian bawah layar. Anda dapat memfilter panel berdasarkan jenis pesan dengan memilih jenis pesan di bagian Filter log. Anda dapat menghapus log panel dengan memilih Hapus konsol.
4. Saat berada di lingkungan pratinjau, Anda dapat menguji aplikasi Anda dengan menavigasi sekitar halamannya, menggunakan komponennya, dan memilih tombolnya untuk memulai otomatisasi yang mentransfer data. Karena lingkungan pratinjau tidak mendukung data langsung atau koneksi ke sumber eksternal, Anda dapat melihat contoh data yang ditransfer di panel debug.

Penerbitan aplikasi

Setelah selesai membuat dan mengonfigurasi aplikasi, langkah selanjutnya adalah mempublikasikannya untuk menguji transfer data atau membagikannya dengan pengguna akhir. Untuk memahami aplikasi penerbitan di App Studio, penting untuk memahami lingkungan yang tersedia. App Studio menyediakan tiga lingkungan terpisah, yang dijelaskan dalam daftar berikut:

1. Pengembangan: Di mana Anda membangun dan melihat pratinjau aplikasi Anda. Anda tidak perlu mempublikasikan ke lingkungan Pengembangan, karena versi terbaru aplikasi Anda di-host di sana secara otomatis. Tidak ada data langsung atau layanan atau sumber daya pihak ketiga yang tersedia di lingkungan ini.
2. Pengujian: Di mana Anda dapat melakukan pengujian komprehensif aplikasi Anda. Di lingkungan Pengujian, Anda dapat terhubung ke, mengirim data ke, dan menerima data dari layanan lain.
3. Produksi: Lingkungan operasional langsung untuk konsumsi pengguna akhir.

Semua pembuatan aplikasi Anda berlangsung di lingkungan Pengembangan. Kemudian, publikasikan ke lingkungan Pengujian untuk menguji transfer data antara layanan lain, dan pengujian penerimaan pengguna (UAT) dengan menyediakan URL akses ke pengguna akhir. Setelah itu, publikasikan aplikasi Anda ke lingkungan Produksi untuk melakukan pengujian akhir sebelum membagikannya kepada pengguna. Untuk informasi selengkapnya tentang lingkungan aplikasi, lihatLingkungan aplikasi.

Ketika Anda mempublikasikan aplikasi, itu tidak tersedia untuk pengguna sampai dibagikan. Ini memberi Anda kesempatan untuk menggunakan dan menguji aplikasi di lingkungan Pengujian dan Produksi sebelum pengguna dapat mengaksesnya. Saat Anda memublikasikan aplikasi ke

Production yang sebelumnya telah diterbitkan dan dibagikan, versi yang tersedia untuk pengguna diperbarui.

Penerbitan aplikasi

Gunakan prosedur berikut untuk memublikasikan aplikasi App Studio ke lingkungan Pengujian atau Produksi.

Untuk mempublikasikan aplikasi ke lingkungan Pengujian atau Produksi

1. Di panel navigasi, pilih Aplikasi saya di bagian Build. Anda akan dibawa ke halaman yang menampilkan daftar aplikasi yang dapat Anda akses.
2. Pilih Edit untuk aplikasi yang ingin Anda publikasikan.
3. Pilih Publikasikan di pojok kanan atas.
4. Dalam kotak dialog Publikasikan pembaruan Anda:
    a. Tinjau informasi tentang menerbitkan aplikasi.
    b. (Opsional) Dalam deskripsi Versi, sertakan deskripsi versi aplikasi ini.
    c. Pilih kotak untuk mengetahui informasi tentang lingkungan.
    d. Pilih Mulai. Diperlukan waktu hingga 15 menit agar aplikasi diperbarui di lingkungan hidup.
5. Untuk informasi tentang melihat aplikasi di lingkungan Pengujian atau Produksi, lihatMelihat aplikasi yang dipublikasikan.

> **Note**
>
> Menggunakan aplikasi di lingkungan Pengujian atau Produksi akan menghasilkan transfer data langsung, seperti membuat catatan dalam tabel sumber data yang telah terhubung dengan konektor.

Aplikasi yang dipublikasikan yang belum pernah dibagikan tidak akan tersedia untuk pengguna atau pembangun lainnya. Untuk membuat aplikasi tersedia bagi pengguna, Anda harus membagikannya setelah penerbitan. Untuk informasi selengkapnya, lihat Berbagi aplikasi yang diterbitkan.

Melihat aplikasi yang dipublikasikan

Anda dapat melihat aplikasi yang dipublikasikan ke lingkungan Pengujian dan Produksi untuk menguji aplikasi sebelum membagikannya dengan pengguna akhir atau pembangun lainnya.

Untuk melihat aplikasi yang dipublikasikan di lingkungan Pengujian atau Produksi

1. Jika perlu, navigasikan ke studio aplikasi aplikasi yang ingin Anda pratinjau:
   a. Di panel navigasi, pilih Aplikasi saya di bagian Build.
   b. Pilih Edit untuk aplikasi.
2. Pilih panah tarik-turun di samping Publish di pojok kanan atas dan pilih Publish Center.
3. Dari pusat penerbitan, Anda dapat melihat lingkungan tempat aplikasi Anda dipublikasikan. Jika aplikasi Anda dipublikasikan ke lingkungan Pengujian atau Produksi, Anda dapat melihat aplikasi menggunakan tautan URL untuk setiap lingkungan.

> **Note**
>
> Menggunakan aplikasi di lingkungan Pengujian atau Produksi akan menghasilkan transfer data langsung, seperti membuat catatan dalam tabel sumber data yang telah terhubung dengan konektor.

Lingkungan aplikasi

AWS App Studio menyediakan kemampuan manajemen siklus hidup aplikasi (ALM) dengan tiga lingkungan terpisah - Pengembangan, Pengujian, dan Produksi. Ini membantu Anda lebih mudah melakukan praktik terbaik seperti memelihara lingkungan terpisah, kontrol versi, berbagi, dan memantau di seluruh siklus hidup aplikasi.

Lingkungan pengembangan

Lingkungan Pengembangan adalah kotak pasir terisolasi tempat Anda dapat membuat aplikasi tanpa terhubung ke sumber atau layanan data langsung apa pun menggunakan studio aplikasi dan data sampel. Di lingkungan Pengembangan, Anda dapat melihat pratinjau aplikasi untuk melihat dan menguji aplikasi tanpa mengorbankan data produksi.

Meskipun aplikasi Anda tidak terhubung ke layanan lain di lingkungan Pengembangan, Anda dapat mengonfigurasi berbagai sumber daya di aplikasi untuk meniru konektor dan otomatisasi data langsung.

Ada panel debug yang dapat dilipat yang menyertakan kesalahan dan peringatan di bagian bawah studio aplikasi di lingkungan Pengembangan untuk membantu Anda memeriksa dan men-debug

aplikasi saat Anda membangun. Untuk informasi selengkapnya tentang pemecahan masalah dan debugging aplikasi, lihat. Memecahkan masalah dan men-debug App Studio

Lingkungan pengujian

Setelah pengembangan aplikasi awal Anda selesai, langkah selanjutnya adalah memublikasikan ke lingkungan Pengujian. Saat berada di lingkungan Pengujian, aplikasi Anda dapat terhubung, mengirim data ke, dan menerima data dari layanan lain. Oleh karena itu, Anda dapat menggunakan lingkungan ini untuk melakukan pengujian komprehensif termasuk pengujian penerimaan pengguna (UAT) dengan menyediakan URL akses ke pengguna akhir.

> **Note**
>
> Publikasi awal Anda ke lingkungan Pengujian dapat memakan waktu hingga 15 menit.

Versi aplikasi yang dipublikasikan ke lingkungan Pengujian akan dihapus setelah 3 jam pengguna akhir tidak aktif. Namun, semua versi tetap ada dan dapat dipulihkan dari tab Riwayat Versi.

Fitur utama dari lingkungan Pengujian adalah sebagai berikut:

- Pengujian integrasi dengan sumber data langsung dan APIs
- Pengujian penerimaan pengguna (UAT) difasilitasi melalui akses terkontrol
- Lingkungan untuk mengumpulkan umpan balik dan mengatasi masalah
- Kemampuan untuk memeriksa dan men-debug aktivitas sisi klien dan sisi server menggunakan konsol browser dan alat pengembang.

Untuk informasi selengkapnya tentang pemecahan masalah dan debugging aplikasi, lihat. Memecahkan masalah dan men-debug App Studio

Lingkungan produksi

Setelah menguji dan memperbaiki masalah apa pun, Anda dapat mempromosikan versi aplikasi dari lingkungan Pengujian ke lingkungan Produksi untuk penggunaan operasional langsung. Meskipun lingkungan Produksi adalah lingkungan operasional langsung untuk konsumsi pengguna akhir, Anda dapat menguji versi yang dipublikasikan sebelum membagikannya kepada pengguna.

Versi Anda yang dipublikasikan di lingkungan Produksi akan dihapus setelah 14 hari pengguna akhir tidak aktif. Namun, semua versi tetap ada dan dapat dipulihkan dari tab Riwayat Versi.

Fitur utama dari lingkungan Produksi adalah sebagai berikut:

- Lingkungan operasional langsung untuk konsumsi pengguna akhir
- Kontrol akses berbasis peran granular
- Kontrol versi dan kemampuan rollback
- Kemampuan untuk memeriksa dan men-debug aktivitas sisi klien saja
- Menggunakan konektor langsung, data, otomatisasi, dan APIs

Manajemen versi dan rilis

App Studio menyediakan kontrol versi dan kemampuan manajemen rilis melalui sistem versioningnya di pusat Publish.

Kemampuan pembuatan versi kunci:

- Penerbitan ke lingkungan Pengujian menghasilkan nomor versi baru (1.0, 2.0, 3.0...).
- Nomor versi tidak berubah saat mempromosikan dari lingkungan Pengujian ke Produksi.
- Anda dapat memutar kembali ke versi sebelumnya dari Riwayat Versi.
- Aplikasi yang dipublikasikan ke lingkungan Pengujian dihentikan sementara setelah 3 jam tidak aktif. Versi dipertahankan dan dapat dipulihkan dari Riwayat Versi.
- Aplikasi yang dipublikasikan ke lingkungan Produksi dihapus setelah 14 hari tidak aktif. Versi dipertahankan dan dapat dipulihkan dari Riwayat Versi.

Model pembuatan versi ini memungkinkan iterasi cepat sambil mempertahankan keterlacakan, kemampuan rollback, dan kinerja optimal di seluruh siklus pengembangan dan pengujian aplikasi.

Pemeliharaan dan operasi

App Studio mungkin perlu menerbitkan ulang aplikasi Anda secara otomatis untuk menangani tugas pemeliharaan tertentu, aktivitas operasional, dan menggabungkan pustaka perangkat lunak baru. Tidak ada tindakan yang diperlukan dari Anda, pembangun, tetapi pengguna akhir mungkin perlu masuk kembali ke aplikasi. Dalam situasi tertentu, kami mungkin meminta Anda untuk menerbitkan ulang aplikasi Anda untuk menggabungkan fitur dan pustaka baru yang tidak dapat kami tambahkan sendiri secara otomatis. Anda harus menyelesaikan kesalahan dan meninjau peringatan sebelum menerbitkan ulang.

Berbagi aplikasi yang diterbitkan

Ketika Anda mempublikasikan aplikasi yang belum dipublikasikan, itu tidak tersedia untuk pengguna sampai dibagikan. Setelah aplikasi yang diterbitkan telah dibagikan, itu akan tersedia untuk pengguna dan tidak perlu dibagikan lagi jika versi lain diterbitkan.

> **Note**
>
> Bagian ini adalah tentang berbagi aplikasi yang diterbitkan dengan pengguna akhir atau penguji. Untuk informasi tentang mengundang pengguna lain untuk membuat aplikasi, lihatMembangun aplikasi dengan banyak pengguna.

Untuk berbagi aplikasi yang diterbitkan

1. Akses kotak dialog Bagikan dari daftar aplikasi, atau studio aplikasi aplikasi Anda dengan menggunakan petunjuk berikut:
   - Untuk mengakses kotak dialog Bagikan dari daftar aplikasi: Di panel navigasi, pilih Aplikasi saya di bagian Build. Pilih dropdown di kolom Tindakan aplikasi yang ingin Anda bagikan dan pilih Bagikan.
   - Untuk mengakses kotak dialog Bagikan dari studio aplikasi: Dari studio aplikasi aplikasi Anda, pilih Bagikan di header atas.
2. Di kotak dialog Bagikan, pilih tab untuk lingkungan yang ingin Anda bagikan. Jika Anda tidak melihat tab Pengujian atau Produksi, aplikasi Anda mungkin tidak dipublikasikan ke lingkungan terkait. Untuk informasi selengkapnya tentang penerbitan, lihatPenerbitan aplikasi.
3. Di tab yang sesuai, pilih grup dari menu tarik-turun untuk berbagi lingkungan dengan mereka.
4. (Opsional) Tetapkan peran tingkat aplikasi ke grup untuk menguji atau mengonfigurasi visibiitas halaman bersyarat. Untuk informasi selengkapnya, lihat Mengkonfigurasi visibilitas halaman berbasis peran.
5. Pilih Bagikan.
6. (Opsional) Salin dan bagikan tautan dengan pengguna. Hanya pengguna yang telah dibagikan aplikasi dan lingkungan yang dapat mengakses aplikasi di lingkungan yang sesuai.

Kembali ke versi yang diterbitkan sebelumnya

Gunakan prosedur berikut untuk memutar kembali (atau mengembalikan) ke versi aplikasi App Studio yang telah dipublikasikan sebelumnya. Ini berguna jika Anda secara tidak sengaja mempublikasikan perubahan yang tidak diinginkan, atau perubahan yang merusak aplikasi untuk pengguna Anda.

Untuk memutar kembali atau kembali ke versi aplikasi yang diterbitkan sebelumnya

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.
2. Pilih panah tarik-turun di sebelah tombol Publikasikan, lalu pilih Publish Center.
3. Pilih Riwayat versi untuk melihat daftar versi aplikasi yang diterbitkan sebelumnya.
4. Temukan versi yang ingin Anda putar kembali, dan pilih Edit.
5. Tinjau informasi pengembalian, dan pilih Kembalikan.
6. Versi yang Anda kembalikan sekarang adalah versi saat ini di studio aplikasi. Anda dapat membuat perubahan padanya, atau mempublikasikannya ke lingkungan Pengujian sebagaimana adanya dengan memilih Publikasikan. Setelah dipublikasikan ke Pengujian, Anda dapat mempublikasikan lagi ke lingkungan Produksi jika diinginkan.

Membangun antarmuka pengguna aplikasi Anda dengan halaman dan komponen

Topik

Membuat, mengedit, atau menghapus halaman

Gunakan prosedur berikut untuk membuat, mengedit, atau menghapus halaman dari AWS aplikasi App Studio Anda.

Halaman adalah wadah untuk komponen, yang membentuk UI aplikasi di App Studio. Setiap halaman mewakili layar antarmuka pengguna (UI) aplikasi Anda yang akan berinteraksi dengan pengguna Anda. Halaman dibuat dan diedit di tab Halaman di studio aplikasi.

Membuat halaman

Gunakan prosedur berikut untuk membuat halaman dalam aplikasi di App Studio.

Untuk membuat halaman

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.
2. Arahkan ke tab Pages di studio aplikasi.
3. Di menu Halaman sisi kiri, pilih + Tambah.

Melihat dan mengedit properti halaman

Gunakan prosedur berikut untuk mengedit halaman dalam aplikasi di App Studio. Anda dapat mengedit properti seperti nama halaman, parameternya, dan tata letaknya.

Untuk melihat atau mengedit properti halaman

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.
2. Arahkan ke tab Pages di studio aplikasi.
3. Di menu Pages sisi kiri, pilih menu elips di sebelah nama halaman yang ingin Anda edit dan pilih properti Halaman. Ini membuka menu Properties sisi kanan.
4. Untuk mengedit nama halaman:

   > **Note**
   >
   > Karakter nama halaman yang valid: A-Z, a-z, 0-9, _, $

   a. Pilih ikon pensil di sebelah nama di dekat bagian atas menu Properties.
   b. Masukkan nama baru untuk halaman Anda dan tekan Enter.
5. Untuk membuat, mengedit, atau menghapus parameter halaman:

   a. Untuk membuat parameter halaman, pilih + Tambahkan baru di bagian Parameter halaman.

b. Untuk mengedit nilai Kunci atau Deskripsi parameter halaman, pilih bidang input properti yang ingin Anda ubah dan masukkan nilai baru. Perubahan Anda disimpan saat Anda mengedit.

c. Untuk menghapus parameter halaman, pilih ikon sampah dari parameter halaman yang ingin Anda hapus.

6. Untuk menambah, mengedit, atau menghapus logo atau spanduk halaman:

a. Untuk menambahkan logo halaman atau spanduk, aktifkan opsi masing-masing di bagian Gaya. Konfigurasikan sumber gambar dan berikan teks alt secara opsional.

b. Untuk mengedit logo halaman atau spanduk, perbarui bidang di bagian Gaya.

c. Untuk menghapus logo halaman atau spanduk, nonaktifkan opsi masing-masing di bagian Gaya.

7. Untuk mengedit tata letak halaman:

- Perbarui bidang di bagian Tata Letak.

Menghapus halaman

Gunakan prosedur berikut untuk menghapus halaman dari aplikasi di App Studio.

Untuk menghapus halaman

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.
2. Arahkan ke tab Pages di studio aplikasi.
3. Di menu Halaman sisi kiri, pilih menu elips di sebelah nama halaman yang ingin Anda hapus dan pilih Hapus.

Menambahkan, mengedit, dan menghapus komponen

Gunakan prosedur berikut untuk menambah, mengedit, dan menghapus komponen di atau dari halaman di studio aplikasi App Studio untuk membuat antarmuka pengguna yang diinginkan untuk aplikasi Anda.

Menambahkan komponen ke halaman

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.

2. Arahkan ke tab Pages di studio aplikasi.
3. Panel komponen terletak di menu sisi kanan, yang berisi komponen yang tersedia.
4. Seret dan jatuhkan komponen yang diinginkan dari panel ke kanvas. Atau, Anda dapat mengklik dua kali pada komponen di panel untuk secara otomatis menambahkannya ke tengah halaman saat ini.
5. Sekarang setelah Anda menambahkan komponen, gunakan panel Properties sisi kanan untuk menyesuaikan pengaturannya, seperti sumber data, tata letak, dan perilaku. Untuk informasi rinci tentang mengonfigurasi setiap jenis komponen, lihatReferensi komponen.

Melihat dan mengedit properti komponen

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.
2. Arahkan ke tab Pages di studio aplikasi.
3. Di menu Pages sisi kiri, perluas halaman yang berisi komponen dan pilih komponen yang akan dilihat atau diedit. Atau, Anda dapat memilih halaman dan kemudian memilih komponen dari kanvas.
4. Panel Properties sisi kanan menampilkan pengaturan yang dapat dikonfigurasi untuk komponen yang dipilih.
5. Jelajahi berbagai properti dan opsi yang tersedia, dan perbarui seperlunya untuk mengonfigurasi tampilan dan perilaku komponen. Misalnya, Anda mungkin ingin mengubah sumber data, mengonfigurasi tata letak, atau mengaktifkan fungsionalitas tambahan.

   Untuk informasi rinci tentang mengonfigurasi setiap jenis komponen, lihatReferensi komponen.

Menghapus komponen

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.
2. Arahkan ke tab Pages di studio aplikasi.
3. Di menu Pages sisi kiri, pilih komponen yang akan dihapus untuk memilihnya.
4. Di menu Properties sisi kanan, pilih ikon sampah.
5. Di kotak dialog konfirmasi, pilih Hapus.

Mengkonfigurasi visibilitas halaman berbasis peran

Anda dapat membuat peran dalam aplikasi App Studio dan mengonfigurasi visbilitas halaman berdasarkan peran tersebut. Misalnya, Anda dapat membuat peran berdasarkan kebutuhan pengguna atau tingkat akses, seperti administrator, manajer, atau pengguna untuk aplikasi yang menyediakan fitur seperti persetujuan proyek atau pemrosesan klaim dan membuat halaman tertentu terlihat oleh peran tertentu. Dalam contoh ini, administrator mungkin memiliki akses penuh, manajer mungkin memiliki akses untuk melihat dasbor pelaporan, dan pengguna mungkin memiliki akses ke halaman tugas dengan formulir input.

Gunakan prosedur berikut untuk mengonfigurasi visbilitas halaman berbasis peran di aplikasi App Studio Anda.

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda. Dari menu navigasi sisi kiri, pilih Aplikasi saya, temukan aplikasi Anda dan pilih Edit.
2. Buat peran tingkat aplikasi di studio aplikasi.
    a. Pilih tab Pengaturan aplikasi di bagian atas studio aplikasi.
    b. Pilih + Tambahkan Peran
    c. Dalam nama Peran, berikan nama untuk mengidentifikasi peran Anda. Sebaiknya gunakan nama yang mendeskriptif tingkat akses atau tugas grup, karena Anda akan menggunakan nama tersebut untuk mengatur visibilitas halaman.
    d. Secara opsional, di Deskripsi, tambahkan deskripsi untuk peran tersebut.
    e. Ulangi langkah-langkah ini untuk membuat peran sebanyak yang diperlukan.
3. Konfigurasikan visiblitas halaman Anda
    a. Pilih tab Pages di bagian atas studio aplikasi.
    b. Dari menu Halaman sisi kiri, pilih halaman yang ingin Anda konfigurasikan visibilitas berbasis peran.
    c. Di menu sisi kanan, pilih tab Properties.
    d. Di Visibilitas, nonaktifkan Buka untuk semua pengguna akhir.
    e. Tetap pilih Peran untuk memilih dari daftar peran yang Anda buat di langkah sebelumnya. Pilih Kustom untuk menulis JavaScript ekspresi untuk konfigurasi visibilitas yang lebih kompleks.
        1. Dengan Peran yang dipilih, centang kotak peran aplikasi yang halamannya akan terlihat.

2. Dengan Custom selected, masukkan JavaScript ekspresi yang menyelesaikan ke true atau false. Gunakan contoh berikut untuk memeriksa apakah pengguna saat ini memiliki peran manajer:`{{currentUser.roles.includes('manager')}}`.

4. Setelah visibilitas Anda dikonfigurasi, Anda dapat menguji visiblitas halaman dengan melihat pratinjau aplikasi Anda.

   a. Pilih Pratinjau untuk membuka pratinjau aplikasi Anda.

   b. Di kanan atas pratinjau, pilih menu Pratinjau sebagai dan centang kotak peran yang ingin Anda uji. Halaman yang terlihat harus mencerminkan peran yang dipilih.

5. Sekarang, tetapkan grup ke peran aplikasi untuk aplikasi yang dipublikasikan. Penugasan grup dan peran harus dikonfigurasi secara terpisah untuk setiap lingkungan. Untuk informasi selengkapnya tentang lingkungan aplikasi, lihatLingkungan aplikasi.

   > **Note**
   >
   > Aplikasi Anda harus dipublikasikan ke lingkungan Pengujian atau Produksi untuk menetapkan grup App Studio ke peran yang telah Anda buat dan konfigurasikan. Jika perlu, publikasikan aplikasi Anda untuk menetapkan grup ke peran. Untuk informasi selengkapnya tentang penerbitan, lihatPenerbitan aplikasi.

   a. Di kanan atas studio aplikasi, pilih Bagikan.

   b. Pilih tab untuk lingkungan yang ingin Anda konfigurasikan visibilitas halaman.

   c. Pilih kotak input Grup pencarian dan pilih grup yang akan digunakan untuk berbagi versi aplikasi. Anda dapat memasukkan teks untuk mencari grup.

   d. Di menu tarik-turun, pilih peran yang akan ditetapkan ke grup. Anda dapat memilih Tidak ada peran untuk membagikan versi aplikasi dan tidak menetapkan peran ke grup. Hanya halaman yang terlihat oleh semua pengguna yang akan terlihat oleh grup tanpa peran.

   e. Pilih Bagikan. Ulangi langkah-langkah ini untuk menambahkan grup sebanyak yang diperlukan.

Memesan dan mengatur halaman dalam navigasi aplikasi

Topik ini mencakup informasi tentang penataan ulang dan pengorganisasian halaman dalam aplikasi App Studio. Ada dua area produk di mana halaman aplikasi terlihat: di menu Halaman sebelah kiri saat mengedit aplikasi di studio aplikasi, dan navigasi sebelah kiri pratinjau aplikasi yang diterbitkan.

Mengurutkan halaman di menu Pages sebelah kiri saat mengedit aplikasi

Saat mengedit aplikasi di studio aplikasi, halaman diurutkan berdasarkan waktu pembuatan di menu Halaman sebelah kiri. Anda tidak dapat menyusun ulang halaman di menu ini.

Mengurutkan, menampilkan, atau menyembunyikan halaman dalam navigasi pratinjau atau aplikasi yang dipublikasikan

Anda dapat mengedit pengaturan navigasi sebelah kiri pratinjau atau aplikasi yang dipublikasikan berikut:

- Visibilitas seluruh navigasi
- Visibilitas halaman tertentu dalam navigasi
- Urutan halaman dalam navigasi

Untuk mengedit navigasi sebelah kiri pratinjau atau aplikasi yang dipublikasikan

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda untuk mengeditnya.
2. Di menu Pages sebelah kiri, pilih Header & navigasi.
3. Di menu Header & navigasi sisi kanan, lihat atau edit yang berikut ini:
    a. Untuk menyembunyikan atau menampilkan navigasi di aplikasi, gunakan sakelar Navigasi aplikasi.
    b. Untuk menyembunyikan halaman dari navigasi aplikasi, seret halaman ke bagian Halaman yang tidak ditautkan.
    c. Untuk menyusun ulang halaman dalam navigasi aplikasi, seret ke urutan yang diinginkan di bagian Halaman Tertaut.

Ubah warna di aplikasi Anda dengan tema aplikasi

Gunakan prosedur berikut untuk memperbarui warna dalam aplikasi Anda dengan mengonfigurasi tema aplikasi.

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda untuk mengeditnya.
2. Di studio aplikasi, arahkan ke tab Halaman.
3. Di navigasi sebelah kiri, pilih Tema aplikasi untuk membuka setelan tema aplikasi sebelah kanan.
4. Pada tema Dasar, pilih mode Cahaya atau mode Gelap.
5. Untuk menambahkan warna kustom ke aplikasi Anda, aktifkan tombol Sesuaikan dan perbarui pengaturan berikut:

    a. Dalam warna Primer, pilih warna yang diterapkan ke komponen tertentu dan navigasi aplikasi Anda. Anda dapat memilih warna dengan color picker, RGB, HSL, atau kode HEX.

    > **Note**
    >
    > App Studio akan secara otomatis memastikan warna Anda dapat diakses. Misalnya, jika Anda memilih warna terang dalam mode cahaya, itu akan diperbarui agar lebih mudah diakses.

    b. Dalam warna Header, pilih warna yang diterapkan ke header aplikasi Anda. Anda dapat memilih warna dengan color picker, RGB, HSL, atau kode HEX.

    c. Pilih tema default untuk melihat dan memilih dari tema yang telah ditentukan sebelumnya, atau pilih Randomize untuk menghasilkan warna primer dan header acak.
6. Pilih Simpan Perubahan untuk memperbarui tema aplikasi Anda.

Referensi komponen

Topik ini merinci setiap komponen App Studio, propertinya, dan menyertakan contoh konfigurasi.

Properti komponen umum

Bagian ini menguraikan properti dan fitur umum yang dibagikan di beberapa komponen di studio aplikasi. Detail implementasi spesifik dan kasus penggunaan untuk setiap jenis properti dapat

bervariasi tergantung pada komponennya, tetapi konsep umum properti ini tetap konsisten di seluruh App Studio.

Nama

Nama default dihasilkan untuk setiap komponen; Namun, Anda dapat mengedit untuk mengubah nama unik untuk setiap komponen. Anda akan menggunakan nama ini untuk mereferensikan komponen dan datanya dari komponen atau ekspresi lain dalam halaman yang sama. Batasan: Jangan sertakan spasi dalam nama komponen; itu hanya dapat memiliki huruf, angka, garis bawah dan tanda dolar. Contoh:`userNameInput`,`ordersTable`,`metricCard1`.

Nilai primer, Nilai sekunder, dan Nilai

Banyak komponen di studio aplikasi menyediakan bidang untuk menentukan nilai atau ekspresi yang menentukan konten atau data yang ditampilkan dalam komponen. Bidang ini sering diberi label sebagai`Primary value`,`Secondary value`, atau sederhananya`Value`, tergantung pada jenis dan tujuan komponen.

`Primary value`Bidang ini biasanya digunakan untuk menentukan nilai utama, titik data, atau konten yang harus ditampilkan secara jelas dalam komponen.

`Secondary value`Bidang, bila tersedia, digunakan untuk menampilkan nilai atau informasi tambahan atau pendukung di samping nilai primer.

`Value`Bidang ini memungkinkan Anda untuk menentukan nilai atau ekspresi yang harus ditampilkan dalam komponen.

Bidang ini mendukung input teks statis dan ekspresi dinamis. Dengan menggunakan ekspresi, Anda dapat mereferensikan data dari komponen lain, sumber data, atau variabel dalam aplikasi Anda, memungkinkan tampilan konten dinamis dan berbasis data.

Sintaks untuk ekspresi

Sintaks untuk memasukkan ekspresi di bidang ini mengikuti pola yang konsisten:

```
{{expression}}
```

Di mana *`expression`* adalah ekspresi valid yang mengevaluasi nilai atau data yang diinginkan yang ingin Anda tampilkan.

Contoh: Teks statis

- Nilai primer: Anda dapat memasukkan nomor statis atau nilai secara langsung, seperti `"123"` atau`"$1,999.99"`.
- Nilai sekunder: Anda dapat memasukkan label teks statis, seperti `"Goal"` atau`"Projected Revenue"`.
- Nilai: Anda dapat memasukkan string statis, seperti `"since last month"` atau`"Total Quantity"`.

Contoh: Ekspresi

- `Hello, {{currentUser.firstName}}`: Menampilkan salam dengan nama depan pengguna yang saat ini masuk.
- `{{currentUser.role === 'Admin' ? 'Admin Dashboard' : 'User Dashboard'}}`: Secara kondisional menampilkan judul dasbor yang berbeda berdasarkan peran pengguna.
- `{{ui.componentName.data?.[0]?.fieldName}}`: Mengambil nilai `fieldName` bidang dari item pertama dalam data komponen dengan ID`componentName`.
- `{{ui.componentName.value * 100}}`: Melakukan perhitungan pada nilai komponen dengan ID`componentName`.
- `{{ui.componentName.value + ' items'}}`: Menggabungkan nilai komponen dengan ID `componentName` dan string. `' items'`
- `{{ui.ordersTable.data?.[0]?.orderNumber}}`: Mengambil nomor urut dari baris pertama data dalam `ordersTable` komponen.
- `{{ui.salesMetrics.data?.[0]?.totalRevenue * 1.15}}`: Menghitung pendapatan yang diproyeksikan dengan meningkatkan total pendapatan dari baris pertama data dalam `salesMetrics` komponen sebesar 15%.
- `{{ui.customerProfile.data?.[0]?.firstName + ' ' + ui.customerProfile.data?.lastName}}`: Menggabungkan nama depan dan belakang dari data dalam komponen. `customerProfile`
- `{{new Date(ui.orderDetails.data?.orderDate).toLocaleDateString()}}`: Memformat tanggal pesanan dari `orderDetails` komponen ke string tanggal yang lebih mudah dibaca.
- `{{ui.productList.data?.length}}`: Menampilkan jumlah total produk dalam data yang terhubung ke `productList` komponen.

- `{{ui.discountPercentage.value * ui.orderTotal.value}}`: Menghitung jumlah diskon berdasarkan persentase diskon dan total pesanan.
- `{{ui.cartItemCount.value + ' items in cart'}}`: Menampilkan jumlah item di keranjang belanja, bersama dengan label`items in cart`.

Dengan menggunakan bidang ekspresi ini, Anda dapat membuat konten dinamis dan berbasis data dalam aplikasi Anda, memungkinkan Anda menampilkan informasi yang disesuaikan dengan konteks pengguna atau status aplikasi Anda. Ini memungkinkan pengalaman pengguna yang lebih personal dan interaktif.

Label

Properti Label memungkinkan Anda untuk menentukan keterangan atau judul untuk komponen. Label ini biasanya ditampilkan di samping atau di atas komponen, membantu pengguna memahami tujuannya.

Anda dapat menggunakan teks statis dan ekspresi untuk menentukan label.

Contoh: Teks statis

Jika Anda memasukkan teks “Nama Depan” di bidang Label, komponen akan menampilkan “Nama Depan” sebagai labelnya.

Contoh: Ekspresi

Contoh: Toko ritel

Contoh berikut mempersonalisasi label untuk setiap pengguna, membuat antarmuka terasa lebih disesuaikan dengan individu:

```
{{currentUser.firstName}} {{currentUser.lastName}}'s Account
```

Contoh: Manajemen proyek SaaS

Contoh berikut menarik data dari proyek yang dipilih untuk memberikan label khusus konteks, membantu pengguna tetap berorientasi dalam aplikasi:

```
Project {{ui.projectsTable.selectedRow.id}} - {{ui.projectsTable.selectedRow.name}}
```

Contoh: Klinik kesehatan

Contoh berikut merujuk profil pengguna saat ini dan informasi dokter, memberikan pengalaman yang lebih personal bagi pasien.

```
Dr. {{ui.doctorProfileTable.data.firstName}}
       {{ui.doctorProfileTable.data.lastName}}
```

Placeholder

Properti Placeholder memungkinkan Anda menentukan petunjuk atau teks panduan yang ditampilkan dalam komponen saat kosong. Ini dapat membantu pengguna memahami format input yang diharapkan atau memberikan konteks tambahan.

Anda dapat menggunakan teks statis dan ekspresi untuk menentukan placeholder.

Contoh: Teks statis

Jika Anda memasukkan teks `Enter your name` di bidang Placeholder, komponen akan ditampilkan `Enter your name` sebagai teks placeholder.

Contoh: Ekspresi

Contoh: Jasa keuangan

`Enter the amount you'd like to deposit into your {{ui.accountsTable.selectedRow.balance}} account`Contoh-contoh ini menarik data dari akun yang dipilih untuk menampilkan petunjuk yang relevan, membuat antarmuka intuitif bagi pelanggan perbankan.

Contoh: E-commerce

`Enter the coupon code for {{ui.cartTable.data.currency}} total`Placeholder di sini diperbarui secara dinamis berdasarkan konten keranjang pengguna, memberikan pengalaman checkout yang mulus.

Contoh: Klinik kesehatan

`Enter your {{ui.patientProfile.data.age}}-year-old patient's symptoms`Dengan menggunakan ekspresi yang merujuk usia pasien, aplikasi dapat membuat placeholder yang lebih personal dan bermanfaat.

Sumber

Properti Sumber memungkinkan Anda untuk memilih sumber data untuk komponen. Setelah dipilih, Anda dapat memilih dari jenis sumber data berikut:`entity`,`expression`, atau`automation`.

Entitas

Memilih Entitas sebagai sumber data memungkinkan Anda menghubungkan komponen ke entitas atau model data yang ada dalam aplikasi Anda. Ini berguna ketika Anda memiliki struktur atau skema data yang terdefinisi dengan baik yang ingin Anda manfaatkan di seluruh aplikasi Anda.

Kapan menggunakan sumber data entitas:

- Bila Anda memiliki model data atau entitas yang berisi informasi yang ingin ditampilkan dalam komponen (misalnya, entitas “Produk” dengan bidang seperti “Nama”, “Deskripsi”, “Harga”).
- Saat Anda perlu mengambil data secara dinamis dari database, API, atau sumber data eksternal lainnya dan menyajikannya dalam komponen.
- Bila Anda ingin memanfaatkan hubungan dan asosiasi yang ditentukan dalam model data aplikasi Anda.

Memilih kueri pada entitas

Terkadang, Anda mungkin ingin menghubungkan komponen ke kueri tertentu yang mengambil data dari entitas, bukan seluruh entitas. Di sumber data Entitas, Anda memiliki opsi untuk memilih dari kueri yang ada atau membuat yang baru.

Dengan memilih kueri, Anda dapat:

- Filter data yang ditampilkan dalam komponen berdasarkan kriteria tertentu.
- Teruskan parameter ke kueri untuk memfilter atau mengurutkan data secara dinamis.
- Manfaatkan gabungan kompleks, agregasi, atau teknik manipulasi data lainnya yang didefinisikan dalam kueri.

Misalnya, jika Anda memiliki `Customers` entitas dalam aplikasi Anda dengan bidang seperti`Name`,`Email`, dan`PhoneNumber`. Anda dapat menghubungkan komponen tabel ke entitas ini dan memilih tindakan `ActiveCustomers` data yang telah ditentukan sebelumnya yang memfilter pelanggan berdasarkan statusnya. Ini memungkinkan Anda untuk menampilkan hanya pelanggan aktif dalam tabel, bukan seluruh basis data pelanggan.

Menambahkan parameter ke sumber data entitas

Saat menggunakan entitas sebagai sumber data, Anda juga dapat menambahkan parameter ke komponen. Parameter ini dapat digunakan untuk memfilter, mengurutkan, atau mengubah data yang ditampilkan dalam komponen.

Misalnya, jika Anda memiliki `Products` entitas dengan bidang sepertiName,Description,Price, danCategory. Anda dapat menambahkan parameter bernama `category` ke komponen tabel yang menampilkan daftar produk. Saat pengguna memilih kategori dari dropdown, tabel akan secara otomatis diperbarui untuk hanya menampilkan produk yang termasuk dalam kategori yang dipilih, menggunakan `{{params.category}}` ekspresi dalam tindakan data.

Ekspresi

Pilih Ekspresi sebagai sumber data untuk memasukkan ekspresi atau perhitungan khusus untuk menghasilkan data komponen secara dinamis. Ini berguna ketika Anda perlu melakukan transformasi, menggabungkan data dari berbagai sumber, atau menghasilkan data berdasarkan logika bisnis tertentu.

Kapan menggunakan sumber data Ekspresi:

- Ketika Anda perlu menghitung atau memperoleh data yang tidak tersedia secara langsung dalam model data Anda (misalnya, menghitung total nilai pesanan berdasarkan kuantitas dan harga).
- Bila Anda ingin menggabungkan data dari beberapa entitas atau sumber data untuk membuat tampilan komposit (misalnya, menampilkan riwayat pesanan pelanggan bersama dengan informasi kontak mereka).
- Saat Anda perlu membuat data berdasarkan aturan atau ketentuan tertentu (misalnya, menampilkan daftar “Produk yang Direkomendasikan” berdasarkan riwayat penelusuran pengguna).

Misalnya, jika Anda memiliki *`Metrics`* komponen yang perlu menampilkan total pendapatan untuk bulan berjalan, Anda dapat menggunakan ekspresi seperti berikut ini untuk menghitung dan menampilkan pendapatan bulanan:

```
{{ui.table1.orders.concat(ui.table1.orderDetails).filter(o => o.orderDate.getMonth()
 === new Date().getMonth()).reduce((a, b) => a + (b.quantity * b.unitPrice), 0)}}
```

Otomatisasi

Pilih Otomasi sebagai sumber data untuk menghubungkan komponen ke otomatisasi atau alur kerja yang ada di aplikasi Anda. Ini berguna ketika data atau fungsionalitas untuk komponen dihasilkan atau diperbarui sebagai bagian dari proses atau alur kerja tertentu.

Kapan menggunakan sumber data Otomasi:

- Ketika data yang ditampilkan dalam komponen adalah hasil dari otomatisasi atau alur kerja tertentu (misalnya, tabel “Persetujuan Tertunda” yang diperbarui sebagai bagian dari proses persetujuan).
- Saat Anda ingin memicu tindakan atau pembaruan komponen berdasarkan peristiwa atau kondisi dalam otomatisasi (misalnya, memperbarui Metrik dengan angka penjualan terbaru untuk SKU).
- Ketika Anda perlu mengintegrasikan komponen dengan layanan atau sistem lain dalam aplikasi Anda melalui otomatisasi (misalnya, mengambil data dari API pihak ketiga dan menampilkannya dalam tabel).

Misalnya, jika Anda memiliki komponen stepflow yang memandu pengguna melalui proses lamaran kerja. Komponen stepflow dapat dihubungkan ke otomatisasi yang menangani pengajuan lamaran kerja, pemeriksaan latar belakang, dan pembuatan penawaran. Saat otomatisasi berlangsung melalui langkah-langkah ini, komponen stepflow dapat diperbarui secara dinamis untuk mencerminkan status aplikasi saat ini.

Dengan hati-hati memilih sumber data yang sesuai untuk setiap komponen, Anda dapat memastikan bahwa antarmuka pengguna aplikasi Anda didukung oleh data dan logika yang tepat, memberikan pengalaman yang mulus dan menarik bagi pengguna Anda.

Terlihat jika

Gunakan properti Visible if untuk menampilkan atau menyembunyikan komponen atau elemen berdasarkan kondisi atau nilai data tertentu. Ini berguna ketika Anda ingin mengontrol visibilitas bagian-bagian tertentu dari antarmuka pengguna aplikasi Anda secara dinamis.

Properti Visible if menggunakan sintaks berikut:

```
{{expression ? true : false}}
```

atau

```
{{expression}}
```

Di mana *`expression`* adalah ekspresi boolean yang mengevaluasi salah satu atau`true`. `false`

Jika ekspresi dievaluasi`true`, komponen akan terlihat. Jika ekspresi dievaluasi`false`, komponen akan disembunyikan. Ekspresi dapat mereferensikan nilai dari komponen lain, sumber data, atau variabel dalam aplikasi Anda.

Contoh ekspresi jika terlihat

Contoh: Menampilkan atau menyembunyikan kolom input kata sandi berdasarkan input email

Bayangkan Anda memiliki formulir login dengan bidang input email dan bidang input kata sandi. Anda ingin menampilkan bidang input kata sandi hanya jika pengguna telah memasukkan alamat email. Anda dapat menggunakan ekspresi Visible if berikut:

```
{{ui.emailInput.value !== ""}}
```

Ekspresi ini memeriksa apakah nilai `emailInput` komponen bukan string kosong. Jika pengguna telah memasukkan alamat email, ekspresi akan dievaluasi`true`, dan bidang input kata sandi akan terlihat. Jika bidang email kosong, ekspresi dievaluasi`false`, dan bidang input kata sandi akan disembunyikan.

Contoh: Menampilkan kolom formulir tambahan berdasarkan pilihan dropdown

Katakanlah Anda memiliki formulir di mana pengguna dapat memilih kategori dari daftar dropdown. Bergantung pada kategori yang dipilih, Anda ingin menampilkan atau menyembunyikan kolom formulir tambahan untuk mengumpulkan informasi yang lebih spesifik.

Misalnya, jika pengguna memilih *`Products`* kategori, Anda dapat menggunakan ekspresi berikut untuk menampilkan *`Product Details`* bidang tambahan:

```
{{ui.categoryDropdown.value === "Products"}}
```

Jika pengguna memilih *`Services`* atau *`Consulting`* kategori, Anda dapat menggunakan ekspresi ini untuk menampilkan kumpulan bidang tambahan yang berbeda:

```
{{ui.categoryDropdown.value === "Services" || ui.categoryDropdown.value ===
 "Consulting"}}
```

Contoh: Lainnya

Untuk membuat komponen terlihat jika nilai `textInput1` komponen bukan string kosong:

```
{{ui.textInput1.value === "" ? false : true}}
```

Untuk membuat komponen selalu terlihat:

```
{{true}}
```

Untuk membuat komponen terlihat jika nilai `emailInput` komponen bukan string kosong:

```
{{ui.emailInput.value !== ""}}
```

Dinonaktifkan jika

Fitur Disabled if memungkinkan Anda mengaktifkan atau menonaktifkan komponen secara kondisional berdasarkan kondisi atau nilai data tertentu. Hal ini dicapai dengan menggunakan properti Disabled if, yang menerima ekspresi boolean yang menentukan apakah komponen harus diaktifkan atau dinonaktifkan.

Properti Disabled if menggunakan sintaks berikut:

```
{{expression ? true : false}}
```

atau

```
{{expression}}
```

Contoh ekspresi jika dinonaktifkan

Contoh: Menonaktifkan tombol kirim berdasarkan validasi formulir

Jika Anda memiliki formulir dengan beberapa bidang input, dan Anda ingin menonaktifkan tombol kirim sampai semua bidang yang diperlukan diisi dengan benar, Anda dapat menggunakan ekspresi Disabled If berikut:

```
{{ui.nameInput.value === "" || ui.emailInput.value === "" || ui.passwordInput.value ===
 ""}}
```

Ekspresi ini memeriksa apakah ada bidang input yang diperlukan (`nameInputemailInput`,,`passwordInput`) kosong. Jika salah satu bidang kosong, ekspresi

dievaluasi`true`, dan tombol kirim akan dinonaktifkan. Setelah semua bidang yang diperlukan diisi, ekspresi dievaluasi`false`, dan tombol kirim akan diaktifkan.

Dengan menggunakan jenis ekspresi bersyarat ini di properti Visible if dan Disabled ff, Anda dapat membuat antarmuka pengguna dinamis dan responsif yang beradaptasi dengan input pengguna, memberikan pengalaman yang lebih efisien dan relevan bagi pengguna aplikasi Anda.

Di mana *`expression`* adalah ekspresi boolean yang mengevaluasi baik benar atau salah.

Contoh:

```
{{ui.textInput1.value === "" ? true : false}}: The component will be Disabled if the
 textInput1 component's value is an empty string.
{{!ui.nameInput.isValid || !ui.emailInput.isValid || !ui.passwordInput.isValid}}: The
 component will be Disabled if any of the named input fields are invalid.
```

Tata letak kontainer

Properti tata letak menentukan bagaimana konten atau elemen dalam komponen diatur dan diposisikan. Beberapa opsi tata letak tersedia, masing-masing diwakili oleh ikon:

- Tata Letak Kolom: Tata letak ini mengatur konten atau elemen secara vertikal, dalam satu kolom.
- Tata letak dua kolom: Tata letak ini membagi komponen menjadi dua kolom dengan lebar yang sama, memungkinkan Anda untuk memposisikan konten atau elemen berdampingan.
- Tata letak baris: Tata letak ini mengatur konten atau elemen secara horizontal, dalam satu baris.

Orientasi

- Horizontal: Tata letak ini mengatur konten atau elemen secara horizontal, dalam satu baris.
- Vertikal: Tata letak ini mengatur konten atau elemen secara vertikal, dalam satu kolom.
- Inline wrapped: Tata letak ini mengatur konten atau elemen secara horizontal, tetapi membungkus ke baris berikutnya jika elemen melebihi lebar yang tersedia.

Penyelarasan

- Kiri: Sejajarkan konten atau elemen ke sisi kiri komponen.
- Pusat: Memusatkan konten atau elemen secara horizontal di dalam komponen.
- Kanan: Menyelaraskan konten atau elemen ke sisi kanan komponen.

Lebar

Properti Width menentukan ukuran horizontal komponen. Anda dapat memasukkan nilai persentase antara 0% dan 100%, mewakili lebar komponen relatif terhadap wadah induknya atau ruang yang tersedia.

Tinggi

Properti Height menentukan ukuran vertikal komponen. Nilai “auto” menyesuaikan tinggi komponen secara otomatis berdasarkan kontennya atau ruang yang tersedia.

Ruang antara

Spasi antar properti menentukan jarak atau kesenjangan antara konten atau elemen dalam komponen. Anda dapat memilih nilai dari 0px (tidak ada spasi) ke 64px, dengan penambahan 4px (misalnya, 4px, 8px, 12px, dll.).

Bantalan

Properti Padding mengontrol ruang antara konten atau elemen dan tepi komponen. Anda dapat memilih nilai dari 0px (tanpa padding) ke 64px, dengan penambahan 4px (misalnya, 4px, 8px, 12px, dll.).

Latar Belakang

Latar Belakang mengaktifkan atau menonaktifkan warna latar belakang atau gaya untuk komponen.

Properti tata letak ini memberikan fleksibilitas dalam mengatur dan memposisikan konten dalam suatu komponen, serta mengontrol ukuran, jarak, dan tampilan visual komponen itu sendiri.

Komponen data

Bagian ini mencakup berbagai komponen data yang tersedia di studio aplikasi, termasuk komponen Tabel, Detail, Metrik, Formulir, dan Repeater. Komponen ini digunakan untuk menampilkan, mengumpulkan, dan memanipulasi data dalam aplikasi Anda.

Tabel

Komponen Tabel menampilkan data dalam format tabel, dengan baris dan kolom. Ini digunakan untuk menyajikan data terstruktur, seperti daftar item atau catatan dari database, secara terorganisir dan easy-to-read cara.

Properti tabel

Komponen Tabel berbagi beberapa sifat umum dengan komponen lain, seperti`Name`,`Source`, dan`Actions`. Untuk informasi lebih lanjut tentang properti ini, lihatProperti komponen umum.

Selain properti umum, komponen Tabel memiliki properti dan opsi konfigurasi tertentu, termasuk`Columns`,`Search and export`, dan`Expressions`.

Kolom

Di bagian ini, Anda dapat menentukan kolom yang akan ditampilkan dalam tabel. Setiap kolom dapat dikonfigurasi dengan properti berikut:

- Format: Tipe data bidang, misalnya: teks, nomor, tanggal.
- Label kolom: Teks header untuk kolom.
- Nilai: Bidang dari sumber data yang harus ditampilkan di kolom ini.

  Bidang ini memungkinkan Anda untuk menentukan nilai atau ekspresi yang harus ditampilkan di sel kolom. Anda dapat menggunakan ekspresi untuk mereferensikan data dari sumber yang terhubung atau komponen lainnya.

  Contoh: `{{currentRow.title}}` - Ekspresi ini akan menampilkan nilai *`title`* bidang dari baris saat ini di sel kolom.
- Aktifkan pengurutan: Toggle ini memungkinkan Anda mengaktifkan atau menonaktifkan fungsionalitas penyortiran untuk kolom tertentu. Saat diaktifkan, pengguna dapat mengurutkan data tabel berdasarkan nilai di kolom ini.

Cari dan ekspor

Komponen Tabel menyediakan sakelar berikut untuk mengaktifkan atau menonaktifkan fungsi pencarian dan ekspor:

- Tampilkan pencarian Saat diaktifkan, sakelar ini menambahkan kolom input pencarian ke tabel, memungkinkan pengguna untuk mencari dan memfilter data yang ditampilkan.
- Tampilkan ekspor Saat diaktifkan, sakelar ini menambahkan opsi ekspor ke tabel, memungkinkan pengguna mengunduh data tabel dalam berbagai format, misalnya: CSV.

> **Note**
>
> Secara default, fungsi pencarian terbatas pada data yang telah dimuat ke dalam tabel. Untuk menggunakan pencarian secara mendalam, Anda harus memuat semua halaman data.

Baris per halaman

Anda dapat menentukan jumlah baris yang akan ditampilkan per halaman dalam tabel. Pengguna kemudian dapat menavigasi antar halaman untuk melihat kumpulan data lengkap.

Batas pra-pengambilan

Tentukan jumlah maksimum catatan yang akan diambil sebelumnya di setiap permintaan kueri. Maksimal 3000.

Tindakan

Di bagian Tindakan, konfigurasikan properti berikut:

- Lokasi tindakan: Saat Pin ke kanan diaktifkan, tindakan tambahan apa pun akan selalu ditampilkan di sebelah kanan tabel, terlepas dari pengguliran pengguna.
- Tindakan: Tambahkan tombol tindakan ke tabel. Anda dapat mengonfigurasi tombol-tombol ini untuk melakukan tindakan tertentu saat diklik oleh pengguna, seperti:
  - Jalankan tindakan komponen
  - Arahkan ke halaman yang berbeda
  - Memanggil tindakan data
  - Jalankan kustom JavaScript
  - Memanggil otomatisasi

Ekspresi

Komponen Tabel menyediakan beberapa area untuk menggunakan ekspresi dan kemampuan tindakan tingkat baris yang memungkinkan Anda menyesuaikan dan meningkatkan fungsionalitas dan interaktivitas tabel. Mereka memungkinkan Anda untuk secara dinamis referensi dan menampilkan data dalam tabel. Dengan memanfaatkan bidang ekspresi ini, Anda dapat membuat kolom dinamis, meneruskan data ke tindakan tingkat baris, dan data tabel referensi dari komponen atau ekspresi lain dalam aplikasi Anda.

Contoh: Merujuk nilai baris

`{{currentRow.columnName}}`atau Ekspresi `{{currentRow["Column Name"]}}` ini memungkinkan Anda untuk mereferensikan nilai kolom tertentu untuk baris saat ini yang sedang dirender. Ganti *`columnName`* atau *`Column Name`* dengan nama sebenarnya dari kolom yang ingin Anda referensikan.

Contoh:

- `{{currentRow.productName}}`Menampilkan nama produk untuk baris saat ini.
- `{{currentRow["Supplier Name"]}}`Menampilkan nama pemasok untuk baris saat ini, di mana header kolom berada*`Supplier Name`*.
- `{{currentRow.orderDate}}`Menampilkan tanggal pesanan untuk baris saat ini.

Contoh: Mereferensikan baris yang dipilih

`{{ui.table1.selectedRow["columnName"]}}`Ekspresi ini memungkinkan Anda untuk mereferensikan nilai kolom tertentu untuk baris yang saat ini dipilih dalam tabel dengan ID*`table1`*. Ganti *`table1`* dengan ID sebenarnya dari komponen tabel Anda, dan *`columnName`* dengan nama kolom yang ingin Anda referensikan.

Contoh:

- `{{ui.ordersTable.selectedRow["totalAmount"]}}`Menampilkan jumlah total untuk baris yang saat ini dipilih dalam tabel dengan ID*`ordersTable`*.
- `{{ui.customersTable.selectedRow["email"]}}`Menampilkan alamat email untuk baris yang dipilih saat ini dalam tabel dengan ID*`customersTable`*.
- `{{ui.employeesTable.selectedRow["department"]}}`Menampilkan departemen untuk baris yang saat ini dipilih dalam tabel dengan ID*`employeesTable`*.

Contoh: Membuat kolom kustom

Anda dapat menambahkan kolom kustom ke tabel berdasarkan data yang dikembalikan dari tindakan, otomatisasi, atau ekspresi data yang mendasarinya. Anda dapat menggunakan nilai kolom dan JavaScript ekspresi yang ada untuk membuat kolom baru.

Contoh:

- `{{currentRow.quantity * currentRow.unitPrice}}`Membuat kolom baru yang menampilkan harga total dengan mengalikan kolom kuantitas dan harga satuan.
- `{{new Date(currentRow.orderDate).toLocaleDateString()}}`Membuat kolom baru yang menampilkan tanggal pemesanan dalam format yang lebih mudah dibaca.
- `{{currentRow.firstName + ' ' + currentRow.lastName + ' (' + currentRow.email + ')' }}`Membuat kolom baru yang menampilkan nama lengkap dan alamat email untuk setiap baris.

Contoh: Menyesuaikan nilai tampilan kolom:

Anda dapat menyesuaikan nilai tampilan bidang dalam kolom tabel dengan mengatur `Value` bidang pemetaan kolom. Ini memungkinkan Anda untuk menerapkan pemformatan atau transformasi khusus ke data yang ditampilkan.

Contoh:

- `{{ currentRow.rating >= 4 ? '##'.repeat(currentRow.rating) : currentRow.rating }}`Menampilkan emoji bintang berdasarkan nilai peringkat untuk setiap baris.
- `{{ currentRow.category.toLowerCase().replace(/\b\w/g, c => c.toUpperCase()) }}`Menampilkan nilai kategori dengan setiap kata dikapitalisasi untuk setiap baris.
- `{{ currentRow.status === 'Active' ? '# Active' : '# Inactive' }}`: Menampilkan emoji lingkaran berwarna dan teks berdasarkan nilai status untuk setiap baris.

Tindakan tombol tingkat baris

`{{currentRow.columnName}}`atau `{{currentRow["Column Name"]}}` Anda dapat menggunakan ekspresi ini untuk meneruskan konteks baris yang direferensikan dalam tindakan tingkat baris, seperti menavigasi ke halaman lain dengan data baris yang dipilih atau memicu otomatisasi dengan data baris.

Contoh:

- Jika Anda memiliki tombol edit di kolom tindakan baris, Anda dapat meneruskan `{{currentRow.orderId}}` sebagai parameter untuk menavigasi ke halaman pengeditan pesanan dengan ID pesanan yang dipilih.

- Jika Anda memiliki tombol hapus di kolom tindakan baris, Anda dapat meneruskan `{{currentRow.customerName}}` ke otomatisasi yang mengirimkan email konfirmasi ke pelanggan sebelum menghapus pesanan mereka.
- Jika Anda memiliki tombol detail tampilan di kolom tindakan baris, Anda dapat meneruskan `{{currentRow.employeeId}}` ke otomatisasi yang mencatat karyawan yang melihat detail pesanan.

Dengan memanfaatkan bidang ekspresi dan kemampuan tindakan tingkat baris ini, Anda dapat membuat tabel yang sangat disesuaikan dan interaktif yang menampilkan dan memanipulasi data berdasarkan kebutuhan spesifik Anda. Selain itu, Anda dapat menghubungkan tindakan tingkat baris dengan komponen atau otomatisasi lain dalam aplikasi Anda, memungkinkan aliran dan fungsionalitas data yang mulus.

Detail

Komponen Detail dirancang untuk menampilkan informasi rinci tentang catatan atau item tertentu. Ini menyediakan ruang khusus untuk menyajikan data komprehensif yang terkait dengan satu entitas atau baris, sehingga ideal untuk menampilkan detail mendalam atau memfasilitasi entri data dan tugas pengeditan.

Properti detail

Komponen Detail berbagi beberapa properti umum dengan komponen lain, seperti`Name`,`Source`, dan`Actions`. Untuk informasi lebih lanjut tentang properti ini, lihatProperti komponen umum.

Komponen Detail juga memiliki properti dan opsi konfigurasi tertentu, termasuk`Fields`,`Layout`, dan`Expressions`.

Tata Letak

Bagian Layout memungkinkan Anda untuk menyesuaikan pengaturan dan presentasi bidang dalam komponen Detail. Anda dapat mengonfigurasi opsi seperti:

- Jumlah kolom: Tentukan jumlah kolom untuk menampilkan bidang di.
- Urutan bidang: Seret dan lepas bidang untuk menyusun ulang tampilannya.
- Spasi dan perataan: Sesuaikan jarak dan penyelarasan bidang di dalam komponen.

Ekspresi dan contoh

Komponen Detail menyediakan berbagai bidang ekspresi yang memungkinkan Anda mereferensikan dan menampilkan data dalam komponen secara dinamis. Ekspresi ini memungkinkan Anda membuat komponen Detail yang disesuaikan dan interaktif yang terhubung secara mulus dengan data dan logika aplikasi Anda.

Contoh: Mereferensikan data

`{{ui.details.data[0]?.["colName"]}}`: Ekspresi ini memungkinkan Anda untuk mereferensikan nilai kolom bernama “colName” untuk item pertama (indeks 0) dalam larik data yang terhubung ke komponen Detail dengan ID “detail”. Ganti “colName” dengan nama sebenarnya dari kolom yang ingin Anda referensikan. Misalnya, ekspresi berikut akan menampilkan nilai kolom “CustomerName” untuk item pertama dalam larik data yang terhubung ke komponen “detail”:

```
{{ui.details.data[0]?.["customerName"]}}
```

> **Note**
>
> Ekspresi ini berguna ketika komponen Detail berada di halaman yang sama dengan tabel yang direferensikan, dan Anda ingin menampilkan data dari baris pertama tabel di komponen Detail.

Contoh: Rendering bersyarat

`{{ui.`*`table1`*`.selectedRow["`*`colName`*`"]}}`: Ekspresi ini mengembalikan true jika baris yang dipilih dalam tabel dengan ID *`table1`* memiliki data untuk kolom bernama*`colName`*. Ini dapat digunakan untuk menampilkan atau menyembunyikan komponen Detail secara kondisional berdasarkan apakah baris tabel yang dipilih kosong atau tidak.

Contoh:

Anda dapat menggunakan ekspresi ini di `Visible if` properti komponen Detail untuk menampilkan atau menyembunyikannya secara kondisional berdasarkan baris yang dipilih dalam tabel.

```
{{ui.table1.selectedRow["customerName"]}}
```

Jika ekspresi ini dievaluasi ke true (baris yang dipilih dalam *`table1`* komponen memiliki nilai untuk *`customerName`* kolom), komponen Detail akan terlihat. Jika ekspresi mengevaluasi ke false (yaitu,

baris yang dipilih kosong atau tidak memiliki nilai untuk “CustomerName”), komponen Detail akan disembunyikan.

Contoh: Tampilan bersyarat

`{{(ui.Component.value === "green" ? "#" : ui.Component.value === "yellow" ? "#" : ui.detail1.data?.[0]?.CustomerStatus)}}`: Ekspresi ini secara kondisional menampilkan emoji berdasarkan nilai komponen atau bidang data.

Kerusakan:

- `ui.`*`Component`*`.value`: Referensi nilai komponen dengan ID*`Component`*.
- `=== "green"`: Memeriksa apakah nilai komponen sama dengan string “hijau”.
- `? "#"`: Jika kondisinya benar, menampilkan emoji lingkaran hijau.
- `: ui.Component.value === "yellow" ? "#"`: Jika kondisi pertama salah, periksa apakah nilai komponen sama dengan string “kuning”.
- `? "#"`: Jika kondisi kedua benar, menampilkan emoji kotak kuning.
- `: ui.detail1.data?.[0]?.CustomerStatus`: Jika kedua kondisi salah, ini mereferensikan nilai CustomerStatus "" dari item pertama dalam larik data yang terhubung ke komponen Detail dengan ID “detail1".

Ekspresi ini dapat digunakan untuk menampilkan emoji atau nilai tertentu berdasarkan nilai komponen atau bidang data dalam komponen Detail.

Metrik

Komponen Metrik adalah elemen visual yang menampilkan metrik kunci atau titik data dalam format seperti kartu. Ini dirancang untuk memberikan cara yang ringkas dan menarik secara visual untuk menyajikan informasi penting atau indikator kinerja.

Properti metrik

Komponen Metrik berbagi beberapa properti umum dengan komponen lain, seperti`Name`,`Source`, dan`Actions`. Untuk informasi lebih lanjut tentang properti ini, lihatProperti komponen umum.

Tren

Fitur tren Metrik memungkinkan Anda untuk menampilkan indikator visual kinerja atau perubahan dari waktu ke waktu untuk metrik yang ditampilkan.

Nilai tren

Bidang ini memungkinkan Anda untuk menentukan nilai atau ekspresi yang harus digunakan untuk menentukan arah dan besarnya tren. Biasanya, ini akan menjadi nilai yang mewakili perubahan atau kinerja selama periode waktu tertentu.

Contoh:

```
{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue}}
```

Ekspresi ini mengambil nilai month-over-month pendapatan dari item pertama dalam data yang terhubung ke Metrik “SalesMetrics”.

Tren positif

Bidang ini memungkinkan Anda untuk memasukkan ekspresi yang mendefinisikan kondisi untuk tren positif. Ekspresi harus mengevaluasi benar atau salah.

Contoh:

```
{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue > 0}}
```

Ekspresi ini memeriksa apakah nilai month-over-month pendapatan lebih besar dari 0, menunjukkan tren positif.

Tren negatif

Bidang ini memungkinkan Anda untuk memasukkan ekspresi yang mendefinisikan kondisi untuk tren negatif. Ekspresi harus mengevaluasi benar atau salah.

Contoh:

```
{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue < 0}}
```

Ekspresi ini memeriksa apakah nilai month-over-month pendapatan kurang dari 0, menunjukkan tren negatif.

Bilah warna

Toggle ini memungkinkan Anda mengaktifkan atau menonaktifkan tampilan bilah berwarna untuk menunjukkan status tren secara visual.

Contoh Color Bar:

Contoh: Tren metrik penjualan

- Nilai tren: `{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue}}`
- Tren positif: `{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue > 0}}`
- Tren negatif: `{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue < 0}}`
- Bilah warna: Diaktifkan

Contoh: tren metrik inventaris

- Nilai tren: `{{ui.inventoryMetrics.data?.[0]?.currentInventory - ui.inventoryMetrics.data?.[1]?.currentInventory}}`
- Tren positif: `{{ui.inventoryMetrics.data?.[0]?.currentInventory > ui.inventoryMetrics.data?.[1]?.currentInventory}}`
- Tren negatif: `{{ui.inventoryMetrics.data?.[0]?.currentInventory < ui.inventoryMetrics.data?.[1]?.currentInventory}}`
- Warna Bbar: Diaktifkan

Contoh: Tren kepuasan pelanggan

- Nilai tren: `{{ui.customerSatisfactionMetrics.data?.[0]?.npsScore}}`
- Tren positif: `{{ui.customerSatisfactionMetrics.data?.[0]?.npsScore >= 8}}`
- Tren negatif: `{{ui.customerSatisfactionMetrics.data?.[0]?.npsScore < 7}}`
- Bilah warna: Diaktifkan

Dengan mengonfigurasi properti terkait tren ini, Anda dapat membuat komponen Metrik yang memberikan representasi visual dari kinerja atau perubahan dari waktu ke waktu untuk metrik yang ditampilkan.

Dengan memanfaatkan ekspresi ini, Anda dapat membuat komponen metrik yang sangat disesuaikan dan interaktif yang mereferensikan dan menampilkan data secara dinamis, memungkinkan Anda menampilkan metrik utama, indikator kinerja, dan visualisasi berbasis data dalam aplikasi Anda.

Contoh ekspresi metrik

Di panel properti, Anda dapat memasukkan ekspresi untuk menampilkan judul, nilai primer, nilai sekunder, dan keterangan nilai untuk menampilkan nilai secara dinamis.

Contoh: Merujuk nilai primer

`{{ui.metric1.primaryValue}}`: Ekspresi ini memungkinkan Anda untuk mereferensikan nilai utama komponen Metrik dengan ID *metric1* dari komponen atau ekspresi lain dalam halaman yang sama.

Contoh: `{{ui.salesMetrics.primaryValue}}` akan menampilkan nilai utama komponen *salesMetrics* Metrik.

Contoh: Merujuk nilai sekunder

`{{ui.metric1.secondaryValue}}`: Ekspresi ini memungkinkan Anda untuk mereferensikan nilai sekunder komponen Metrik dengan ID *metric1* dari komponen atau ekspresi lain dalam halaman yang sama.

Contoh: `{{ui.revenueMetrics.secondaryValue}}` akan menampilkan nilai sekunder dari komponen *revenueMetrics* Metrik.

Contoh: Mereferensikan data

`{{ui.metric1.data}}`: Ekspresi ini memungkinkan Anda untuk mereferensikan data komponen Metrik dengan ID *metric1* dari komponen atau ekspresi lain dalam halaman yang sama.

Contoh: `{{ui.kpiMetrics.data}}` akan mereferensikan data yang terhubung ke komponen *kpiMetrics* Metrik.

Contoh: Menampilkan nilai data tertentu:

`{{ui.metric1.data?.[0]?.id}}`: Ekspresi ini adalah contoh bagaimana menampilkan informasi tertentu dari data yang terhubung ke komponen Metrik dengan ID*metric1*. Ini berguna ketika Anda ingin menampilkan properti tertentu dari item pertama dalam data.

Kerusakan:

- `ui.metric1`: Referensi komponen Metrik dengan ID*metric1*.
- `data`: Mengacu pada informasi atau kumpulan data yang terhubung ke komponen itu.
- `?.[0]`: Berarti item atau entri pertama dalam kumpulan data itu.

- `?.id`: Menampilkan *`id`* nilai atau pengenal item atau entri pertama itu.

Contoh: `{{ui.`*`orderMetrics`*`.data?.[0]?.`*`orderId`*`}}` akan menampilkan *`orderId`* nilai item pertama dalam data yang terhubung ke komponen *`orderMetrics`* Metrik.

Contoh: Menampilkan panjang data

`{{ui.`*`metric1`*`.data?.length}}`: Ekspresi ini menunjukkan cara menampilkan panjang (jumlah item) dalam data yang terhubung ke komponen Metrik dengan ID. *`metric1`* Ini berguna ketika Anda ingin menampilkan jumlah item dalam data.

Kerusakan:

- `ui.`*`metric1`*`.data`: Referensi kumpulan data yang terhubung ke komponen.
- `?.length`: Mengakses jumlah total atau jumlah item atau entri dalam kumpulan data tersebut.

Contoh: `{{ui.`*`productMetrics`*`.data?.length}}` akan menampilkan jumlah item dalam data yang terhubung ke komponen *`productMetrics`* Metrik.

Repeater

Komponen Repeater adalah komponen dinamis yang memungkinkan Anda menghasilkan dan menampilkan kumpulan elemen berdasarkan sumber data yang disediakan. Ini dirancang untuk memfasilitasi pembuatan daftar, kisi, atau pola berulang dalam antarmuka pengguna aplikasi Anda. Beberapa contoh kasus penggunaan meliputi:

- Menampilkan kartu untuk setiap pengguna di akun
- Menampilkan daftar produk yang menyertakan gambar dan tombol untuk menambahkannya ke keranjang
- Menampilkan daftar file yang dapat diakses pengguna

Komponen Repeater membedakan dirinya dari komponen Table dengan konten yang kaya. Komponen Tabel memiliki format baris dan kolom yang ketat. Repeater dapat menampilkan data Anda dengan lebih fleksibel.

Properti repeater

Komponen Repeater berbagi beberapa properti umum dengan komponen lain, seperti`Name`,`Source`, dan`Actions`. Untuk informasi lebih lanjut tentang properti ini, lihatProperti komponen umum.

Selain properti umum, komponen Repeater memiliki properti tambahan dan opsi konfigurasi berikut.

Template barang

Template Item adalah wadah tempat Anda dapat menentukan struktur dan komponen yang akan diulang untuk setiap item di sumber data. Anda dapat menarik dan melepas komponen lain ke dalam wadah ini, seperti Teks, Gambar, Tombol, atau komponen lain yang Anda butuhkan untuk mewakili setiap item.

Di dalam Template item, Anda dapat mereferensikan properti atau nilai dari item saat ini menggunakan ekspresi dalam format`{{currentItem.propertyName}}`.

Misalnya, jika sumber data Anda berisi `itemName` properti, Anda dapat menggunakan `{{currentItem.itemName}}` untuk menampilkan nama item dari item saat ini.

Tata Letak

Bagian Layout memungkinkan Anda untuk mengkonfigurasi pengaturan elemen berulang dalam Komponen Repeater.

Orientasi

- Daftar: Mengatur elemen berulang secara vertikal dalam satu kolom.
- Grid: Mengatur elemen berulang dalam tata letak grid dengan beberapa kolom.

Baris per halaman

Tentukan jumlah baris yang akan ditampilkan per halaman dalam tata letak daftar. Pagination disediakan untuk item yang meluap jumlah baris yang ditentukan.

Kolom dan Baris per Halaman (Grid)

- Kolom: Tentukan jumlah kolom dalam tata letak kisi.
- Baris per Halaman: Tentukan jumlah baris yang akan ditampilkan per halaman dalam tata letak kisi. Pagination disediakan untuk item yang meluap dimensi grid yang ditentukan.

Ekspresi dan contoh

Komponen Repeater menyediakan berbagai bidang ekspresi yang memungkinkan Anda mereferensikan dan menampilkan data dalam komponen secara dinamis. Ekspresi ini memungkinkan

Anda membuat komponen Repeater yang disesuaikan dan interaktif yang terhubung secara mulus dengan data dan logika aplikasi Anda.

Contoh: Merujuk item

- `{{currentItem.propertyName}}`: Properti referensi atau nilai dari item saat ini dalam Template Item.
- `{{ui.repeaterID[index]}}`: Referensi item tertentu di Komponen Repeater dengan indeksnya.

Contoh: Merender daftar produk

- Sumber: Pilih *`Products`* entitas sebagai sumber data.
- Template Item: Tambahkan komponen Container dengan komponen Teks di dalamnya untuk menampilkan nama produk (`{{currentItem.productName}}`) dan komponen Image untuk menampilkan image produk (`{{currentItem.productImageUrl}}`).
- Layout: Atur `Orientation` ke `List` dan `Rows per Page` sesuaikan sesuai keinginan.

Contoh: Menghasilkan kisi avatar pengguna

- Sumber: Gunakan ekspresi untuk menghasilkan array data pengguna (misalnya,`[{name: 'John', avatarUrl: '...'}, {...}, {...}]`).
- Template Item: Tambahkan komponen Image dan atur `Source` propertinya ke`{{currentItem.avatarUrl}}`.
- Layout: Atur `Orientation` ke`Grid`, tentukan jumlah `Columns` dan`Rows per Page`, dan sesuaikan `Space Between` dan `Padding` sesuai kebutuhan.

Dengan menggunakan `Repeater` komponen, Anda dapat membuat antarmuka pengguna dinamis dan berbasis data, merampingkan proses rendering koleksi elemen dan mengurangi kebutuhan untuk pengulangan manual atau hard-coding.

Formulir

Komponen Formulir dirancang untuk menangkap input pengguna dan memfasilitasi tugas entri data dalam aplikasi Anda. Ini menyediakan tata letak terstruktur untuk menampilkan bidang input, dropdown, kotak centang, dan kontrol formulir lainnya, memungkinkan pengguna untuk memasukkan

atau memodifikasi data dengan mulus. Anda dapat menyarangkan komponen lain di dalam komponen formulir, seperti tabel.

Properti bentuk

Komponen Form berbagi beberapa properti umum dengan komponen lain, seperti`Name`,`Source`, dan`Actions`. Untuk informasi lebih lanjut tentang properti ini, lihatProperti komponen umum.

Menghasilkan Formulir

Fitur Hasilkan Formulir memudahkan untuk membuat bidang formulir dengan cepat dengan mengisinya secara otomatis berdasarkan sumber data yang dipilih. Hal ini dapat menghemat waktu dan tenaga ketika membangun formulir yang perlu menampilkan sejumlah besar bidang.

Untuk menggunakan fitur Generate Form:

1. Dalam properti komponen Form, cari bagian Generate Form.
2. Pilih sumber data yang ingin Anda gunakan untuk menghasilkan bidang formulir. Ini bisa berupa entitas, alur kerja, atau sumber data lain yang tersedia di aplikasi Anda.
3. Bidang formulir akan dibuat secara otomatis berdasarkan sumber data yang dipilih, termasuk label bidang, jenis, dan pemetaan data.
4. Tinjau bidang yang dihasilkan dan buat penyesuaian yang diperlukan, seperti menambahkan aturan validasi atau mengubah urutan bidang.
5. Setelah Anda puas dengan konfigurasi formulir, pilih Kirim untuk menerapkan bidang yang dihasilkan ke komponen Formulir Anda.

Fitur Generate Form sangat berguna ketika Anda memiliki model data yang terdefinisi dengan baik atau kumpulan entitas dalam aplikasi Anda yang Anda perlukan untuk menangkap masukan pengguna. Dengan membuat kolom formulir secara otomatis, Anda dapat menghemat waktu dan memastikan konsistensi di seluruh formulir aplikasi Anda.

Setelah menggunakan fitur Hasilkan Formulir, Anda dapat menyesuaikan tata letak, tindakan, dan ekspresi untuk komponen Formulir agar sesuai dengan kebutuhan spesifik Anda.

Ekspresi dan contoh

Seperti komponen lainnya, Anda dapat menggunakan ekspresi untuk referensi dan menampilkan data dalam komponen Formulir. Sebagai contoh:

- `{{ui.userForm.data.email}}`: Referensi nilai `email` bidang dari sumber data yang terhubung ke komponen Formulir dengan ID`userForm`.

> **Note**
>
> Lihat Properti komponen umum untuk contoh ekspresi lebih lanjut dari properti umum.

Dengan mengonfigurasi properti ini dan memanfaatkan ekspresi, Anda dapat membuat komponen Formulir yang disesuaikan dan interaktif yang terintegrasi secara mulus dengan sumber data dan logika aplikasi Anda. Komponen-komponen ini dapat digunakan untuk menangkap input pengguna, menampilkan data yang telah diisi sebelumnya, dan memicu tindakan berdasarkan pengiriman formulir atau interaksi pengguna.

Stepflow

Komponen Stepflow dirancang untuk memandu pengguna melalui proses multi-langkah atau alur kerja dalam aplikasi Anda. Ini menyediakan antarmuka terstruktur dan intuitif untuk menyajikan urutan langkah, masing-masing dengan set input, validasi, dan tindakannya sendiri.

Komponen Stepflow berbagi beberapa properti umum dengan komponen lain, seperti`Name`,`Source`, dan`Actions`. Untuk informasi lebih lanjut tentang properti ini, lihatProperti komponen umum.

Komponen Stepflow memiliki properti tambahan dan opsi konfigurasi, seperti`Step Navigation`,`Validation`, dan`Expressions`.

Komponen AI

Generasi AI

Komponen Gen AI adalah wadah pengelompokan yang digunakan untuk mengelompokkan komponen dan logika yang menyertainya untuk dengan mudah mengeditnya dengan AI menggunakan obrolan di dalam studio aplikasi. Saat Anda menggunakan obrolan untuk membuat komponen, mereka akan dikelompokkan ke dalam wadah Gen AI. Untuk informasi tentang mengedit atau menggunakan komponen ini, lihatMembangun atau mengedit aplikasi Anda.

Komponen teks & angka

Masukan teks

Komponen input Teks memungkinkan pengguna untuk memasukkan dan mengirimkan data teks dalam aplikasi Anda. Ini menyediakan cara sederhana dan intuitif untuk menangkap input pengguna, seperti nama, alamat, atau informasi tekstual lainnya.

- `{{ui.inputTextID.value}}`: Mengembalikan nilai yang disediakan di bidang input.
- `{{ui.inputTextID.isValid}}`: Mengembalikan validitas nilai yang disediakan di bidang input.

Teks

Komponen Teks digunakan untuk menampilkan informasi tekstual dalam aplikasi Anda. Ini dapat digunakan untuk menampilkan teks statis, nilai dinamis, atau konten yang dihasilkan dari ekspresi.

Area teks

Komponen area Teks dirancang untuk menangkap input teks multi-baris dari pengguna. Ini menyediakan area bidang input yang lebih besar bagi pengguna untuk memasukkan entri teks yang lebih panjang, seperti deskripsi, catatan, atau komentar.

- `{{ui.textAreaID.value}}`: Mengembalikan nilai yang disediakan di area teks.
- `{{ui.textAreaID.isValid}}`: Mengembalikan validitas nilai yang disediakan di area teks.

Email

Komponen Email adalah bidang input khusus yang dirancang untuk menangkap alamat email dari pengguna. Ini dapat menegakkan aturan validasi khusus untuk memastikan nilai yang dimasukkan mematuhi format email yang benar.

- `{{ui.emailID.value}}`: Mengembalikan nilai yang disediakan di bidang input email.
- `{{ui.emailID.isValid}}`: Mengembalikan validitas nilai yang disediakan di bidang input email.

Kata sandi

Komponen Kata Sandi adalah bidang input yang dirancang khusus bagi pengguna untuk memasukkan informasi sensitif, seperti kata sandi atau kode PIN. Ini menutupi karakter yang dimasukkan untuk menjaga privasi dan keamanan.

- `{{ui.passwordID.value}}`: Mengembalikan nilai yang disediakan di bidang input password.
- `{{ui.passwordID.isValid}}`: Mengembalikan validitas nilai yang disediakan di bidang input password.

Pencarian

Komponen Pencarian memberi pengguna bidang input khusus untuk melakukan kueri penelusuran atau memasukkan istilah penelusuran dalam data yang terisi dalam aplikasi.

- `{{ui.searchID.value}}`: Mengembalikan nilai yang disediakan di bidang pencarian.

Telepon

Komponen Telepon adalah bidang input yang disesuaikan untuk menangkap nomor telepon atau informasi kontak lainnya dari pengguna. Ini dapat mencakup aturan validasi khusus dan opsi pemformatan untuk memastikan nilai yang dimasukkan mematuhi format nomor telepon yang benar.

- `{{ui.phoneID.value}}`: Mengembalikan nilai yang disediakan di bidang input telepon.
- `{{ui.phoneID.isValid}}`: Mengembalikan validitas nilai yang disediakan di bidang input telepon.

Jumlah

Komponen Number adalah bidang input yang dirancang khusus bagi pengguna untuk memasukkan nilai numerik. Ini dapat menegakkan aturan validasi untuk memastikan nilai yang dimasukkan adalah nomor yang valid dalam rentang atau format tertentu.

- `{{ui.numberID.value}}`: Mengembalikan nilai yang disediakan di bidang input angka.
- `{{ui.numberID.isValid}}`: Mengembalikan validitas nilai yang disediakan di bidang input angka.

Mata Uang

Komponen mata uang adalah bidang input khusus untuk menangkap nilai atau jumlah moneter. Ini dapat mencakup opsi pemformatan untuk menampilkan simbol mata uang, pemisah desimal, dan menegakkan aturan validasi khusus untuk input mata uang.

- `{{ui.currencyID.value}}`: Mengembalikan nilai yang diberikan dalam bidang masukan mata uang.
- `{{ui.currencyID.isValid}}`: Mengembalikan validitas nilai yang diberikan di bidang masukan mata uang.

Beralih

Komponen Switch adalah kontrol antarmuka pengguna yang memungkinkan pengguna untuk beralih di antara dua status atau opsi, seperti. on/off, true/false, or enabled/disabled Ini memberikan representasi visual dari keadaan saat ini dan memungkinkan pengguna untuk mengubahnya dengan satu klik atau ketuk.

Detail pasangan

Komponen pasangan Detail digunakan untuk menampilkan pasangan kunci-nilai atau pasangan informasi terkait dalam format terstruktur dan dapat dibaca. Ini biasanya digunakan untuk menyajikan detail atau metadata yang terkait dengan item atau entitas tertentu.

Komponen seleksi

Ganti grup

Komponen grup Switch adalah kumpulan kontrol sakelar individual yang memungkinkan pengguna memilih satu atau beberapa opsi dari set yang telah ditentukan sebelumnya. Ini memberikan representasi visual dari opsi yang dipilih dan tidak dipilih, sehingga memudahkan pengguna untuk memahami dan berinteraksi dengan pilihan yang tersedia.

Beralih bidang ekspresi grup

- `{{ui.switchGroupID.value}}`: Mengembalikan array string yang berisi nilai setiap sakelar yang diaktifkan oleh pengguna aplikasi.

Grup kotak centang

Komponen grup Checkbox menyajikan pengguna dengan sekelompok kotak centang, memungkinkan mereka untuk memilih beberapa opsi secara bersamaan. Ini berguna ketika Anda ingin memberi pengguna kemampuan untuk memilih satu atau lebih item dari daftar opsi.

Bidang ekspresi grup kotak centang

- `{{ui.checkboxGroupID.value}}`: Mengembalikan array string yang berisi nilai setiap kotak centang yang dipilih oleh pengguna aplikasi.

Grup radio

Komponen grup Radio adalah satu set tombol radio yang memungkinkan pengguna untuk memilih satu opsi dari beberapa pilihan yang saling eksklusif. Ini memastikan bahwa hanya satu opsi yang dapat dipilih pada satu waktu, memberikan cara yang jelas dan tidak ambigu bagi pengguna untuk membuat pilihan.

Bidang ekspresi grup radio

Bidang berikut dapat digunakan dalam ekspresi.

- `{{ui.radioGroupID.value}}`: Mengembalikan nilai tombol radio yang dipilih oleh pengguna aplikasi.

Pilih tunggal

Komponen Pilih Tunggal menyajikan pengguna dengan daftar opsi, dari mana mereka dapat memilih satu item. Ini biasanya digunakan dalam skenario di mana pengguna perlu membuat pilihan dari serangkaian opsi yang telah ditentukan, seperti memilih kategori, lokasi, atau preferensi.

Bidang ekspresi pilih tunggal

- `{{ui.singleSelectID.value}}`: Mengembalikan nilai item daftar yang dipilih oleh pengguna aplikasi.

Multi pilih

Komponen Multi select mirip dengan komponen Pilih Tunggal tetapi memungkinkan pengguna untuk memilih beberapa opsi secara bersamaan dari daftar pilihan. Ini berguna ketika pengguna perlu membuat beberapa pilihan dari serangkaian opsi yang telah ditentukan, seperti memilih beberapa tag, minat, atau preferensi.

Bidang ekspresi multi pilih

- `{{ui.multiSelectID.value}}`: Mengembalikan array string yang berisi nilai setiap item daftar yang dipilih oleh pengguna aplikasi.

Tombol & komponen navigasi

Studio aplikasi menyediakan berbagai tombol dan komponen navigasi untuk memungkinkan pengguna memicu tindakan dan menavigasi dalam aplikasi Anda.

Komponen tombol

Komponen tombol yang tersedia adalah:

- Tombol
- Tombol yang diuraikan
- Tombol ikon
- Tombol teks

Komponen tombol ini berbagi properti umum berikut:

Daftar isi

- Label tombol: Teks yang akan ditampilkan pada tombol.

Tipe

- Tombol: Tombol standar.
- Diuraikan: Tombol dengan gaya yang diuraikan.
- Ikon: Tombol dengan ikon.
- Teks: Tombol teks saja.

Size

Ukuran tombol. Nilai yang mungkin adalah `Small`, `Medium`, dan `Large`.

Ikon

Anda dapat memilih dari berbagai ikon yang akan ditampilkan pada tombol, termasuk:

- Amplop Tertutup
- lonceng
- Orang
- Menu Hamburger
- Pencarian
- Info dilingkari
- Perlengkapan
- Chevron Kiri
- Chevron Kanan
- Titik Horisontal
- Sampah
- Sunting
- Memeriksa
- Tutup
- Beranda
- Ditambah

Pemicu

Saat tombol diklik, Anda dapat mengonfigurasi satu atau beberapa tindakan yang akan dipicu. Jenis tindakan yang tersedia adalah:

- Basic
  - Jalankan tindakan komponen: Mengeksekusi tindakan tertentu dalam komponen.
  - Navigasi: Menavigasi ke halaman atau tampilan lain.
  - Memanggil Tindakan Data: Memicu tindakan terkait data, seperti membuat, memperbarui, atau menghapus catatan.
- Advanced
  - JavaScript: Menjalankan JavaScript kode kustom.

- Memanggil Otomasi: Memulai otomatisasi atau alur kerja yang ada.

JavaScript properti tombol aksi

Pilih jenis `JavaScript` tindakan untuk menjalankan JavaScript kode kustom saat tombol diklik.

Kode sumber

Di `Source code` bidang, Anda dapat memasukkan JavaScript ekspresi atau fungsi Anda. Sebagai contoh:

```
return "Hello World";
```

Ini hanya akan mengembalikan string `Hello World` ketika tombol diklik.

Kondisi: Jalankan jika

Anda juga dapat memberikan ekspresi boolean yang menentukan apakah JavaScript tindakan harus dijalankan atau tidak. Ini menggunakan sintaks berikut:

```
{{ui.textinput1.value !== ""}}
```

Dalam contoh ini, JavaScript tindakan hanya akan berjalan jika nilai `textinput1` komponen bukan string kosong.

Dengan menggunakan opsi pemicu lanjutan ini, Anda dapat membuat perilaku tombol yang sangat disesuaikan yang terintegrasi langsung dengan logika dan data aplikasi Anda. Ini memungkinkan Anda untuk memperluas fungsionalitas tombol bawaan dan menyesuaikan pengalaman pengguna dengan kebutuhan spesifik Anda.

> **Note**
>
> Selalu uji JavaScript tindakan Anda secara menyeluruh untuk memastikan mereka berfungsi seperti yang diharapkan.

Hyperlink

Komponen Hyperlink menyediakan tautan yang dapat diklik untuk menavigasi ke rute aplikasi eksternal atau internal. URLs

Properti hyperlink

Daftar isi

- Label hyperlink: Teks yang akan ditampilkan sebagai label hyperlink.

URL

URL tujuan untuk hyperlink, yang dapat berupa situs web eksternal atau rute aplikasi internal.

Pemicu

Saat hyperlink diklik, Anda dapat mengonfigurasi satu atau beberapa tindakan yang akan dipicu. Jenis tindakan yang tersedia sama dengan yang ada pada komponen tombol.

Komponen tanggal & waktu

Tanggal

Komponen Tanggal memungkinkan pengguna untuk memilih dan memasukkan tanggal.

Komponen Tanggal berbagi beberapa properti umum dengan komponen lain, seperti`Name`,`Source`, dan`Validation`. Untuk informasi lebih lanjut tentang properti ini, lihatProperti komponen umum.

Selain properti umum, komponen Tanggal memiliki properti spesifik berikut:

Properti tanggal

format

- YYYY/MM/DD, DD/MM/YYYY, YYYY/MM/DD, YYYY/DD/MM, MM/DD, DD/MM: Format di mana tanggal harus ditampilkan.

Nilai

- YYYY-MM-DD: Format di mana nilai tanggal disimpan secara internal.

Tanggal Min

- YYYY-MM-DD: Tanggal minimum yang dapat dipilih.

> **Note**
>
> Nilai ini harus sesuai dengan format`YYYY-MM-DD`.

Tanggal maks

- YYYY-MM-DD: Tanggal maksimum yang dapat dipilih.

> **Note**
>
> Nilai ini harus sesuai dengan format`YYYY-MM-DD`.

Jenis kalender

- 1 Bulan, 2 Bulan: Jenis UI kalender yang akan ditampilkan.

Tanggal dinonaktifkan

- Sumber: Sumber data untuk tanggal yang harus dinonaktifkan. Misalnya: Tidak ada, Ekspresi.
- Tanggal dinonaktifkan: Ekspresi yang menentukan tanggal mana yang harus dinonaktifkan, seperti:
  - `{{currentRow.column}}`: Menonaktifkan tanggal yang cocok dengan apa yang dievaluasi oleh ekspresi ini.
  - `{{new Date(currentRow.dateColumn) < new Date("2023-01-01")}}`: Menonaktifkan tanggal sebelum 1 Januari 2023
  - `{{new Date(currentRow.dateColumn).getDay() === 0 || new Date(currentRow.dateColumn).getDay() === 6}}`: Menonaktifkan akhir pekan.

Perilaku

- Terlihat jika: Ekspresi yang menentukan visibilitas komponen Tanggal.
- Nonaktifkan if: Ekspresi yang menentukan apakah komponen Tanggal harus dinonaktifkan.

Validasi

Bagian Validasi memungkinkan Anda untuk menentukan aturan dan batasan tambahan untuk masukan tanggal. Dengan mengonfigurasi aturan validasi ini, Anda dapat memastikan bahwa nilai tanggal yang dimasukkan oleh pengguna memenuhi persyaratan spesifik aplikasi Anda. Anda dapat menambahkan jenis validasi berikut:

- Wajib: Toggle ini memastikan bahwa pengguna harus memasukkan nilai tanggal sebelum mengirimkan formulir.
- Kustom: Anda dapat membuat aturan validasi kustom menggunakan JavaScript ekspresi. Sebagai contoh:

  ```
  {{new Date(ui.dateInput.value) < new Date("2023-01-01")}}
  ```

  Ekspresi ini memeriksa apakah tanggal yang dimasukkan sebelum 1 Januari 2023. Jika kondisinya benar, validasi akan gagal.

  Anda juga dapat memberikan pesan validasi kustom untuk ditampilkan ketika validasi tidak terpenuhi:

  ```
  "Validation not met. The date must be on or after January 1, 2023."
  ```

Dengan mengonfigurasi aturan validasi ini, Anda dapat memastikan bahwa nilai tanggal yang dimasukkan oleh pengguna memenuhi persyaratan spesifik aplikasi Anda.

Ekspresi dan contoh

Komponen Tanggal menyediakan bidang ekspresi berikut:

- `{{ui.dateID.value}}`: Mengembalikan nilai tanggal yang dimasukkan oleh pengguna dalam format`YYYY-MM-DD`.

Waktu

Komponen Time memungkinkan pengguna untuk memilih dan memasukkan nilai waktu. Dengan mengonfigurasi berbagai properti komponen Time, Anda dapat membuat kolom input waktu yang memenuhi persyaratan spesifik aplikasi Anda, seperti membatasi rentang waktu yang dapat dipilih, menonaktifkan waktu tertentu, dan mengontrol visibilitas dan interaktivitas komponen.

Properti waktu

Komponen Time berbagi beberapa properti umum dengan komponen lain, seperti`Name`,`Source`, dan`Validation`. Untuk informasi lebih lanjut tentang properti ini, lihatProperti komponen umum.

Selain properti umum, komponen Time memiliki properti spesifik berikut:

Interval waktu

- 5 menit, 10 menit, 15 menit, 20 menit, 25 menit, 30 menit, 60 menit: Interval yang tersedia untuk memilih waktu.

Nilai

- HH: MM AA: Format di mana nilai waktu disimpan secara internal.

> **Note**
>
> Nilai ini harus sesuai dengan format`HH:MM AA`.

Placeholder

- Pengaturan kalender: Teks placeholder ditampilkan saat bidang waktu kosong.

Waktu min

- HH: MM AA: Waktu minimum yang dapat dipilih.

> **Note**
>
> Nilai ini harus sesuai dengan format`HH:MM AA`.

Waktu maks

- HH: MM AA: Waktu maksimum yang dapat dipilih.

> **Note**
>
> Nilai ini harus sesuai dengan format`HH:MM AA`.

Waktu dinonaktifkan

- Sumber: Sumber data untuk waktu yang harus dinonaktifkan (misalnya, Tidak Ada, Ekspresi).
- Waktu dinonaktifkan: Ekspresi yang menentukan waktu mana yang harus dinonaktifkan, seperti`{{currentRow.column}}`.

Konfigurasi waktu dinonaktifkan

Anda dapat menggunakan bagian Waktu Dinonaktifkan untuk menentukan nilai waktu mana yang seharusnya tidak tersedia untuk dipilih.

Sumber

- Tidak ada: Tidak ada waktu dinonaktifkan.
- Ekspresi: Anda dapat menggunakan JavaScript ekspresi untuk menentukan waktu mana yang harus dinonaktifkan, seperti`{{currentRow.column}}`.

Contoh ekspresi:

```
{{currentRow.column === "Lunch Break"}}
```

Ekspresi ini akan menonaktifkan setiap saat di mana kolom “Lunch Break” benar untuk baris saat ini.

Dengan mengonfigurasi aturan validasi ini dan ekspresi waktu yang dinonaktifkan, Anda dapat memastikan bahwa nilai waktu yang dimasukkan oleh pengguna memenuhi persyaratan spesifik aplikasi Anda.

Perilaku

- Terlihat jika: Ekspresi yang menentukan visibilitas komponen Waktu.
- Nonaktifkan if: Ekspresi yang menentukan apakah komponen Time harus dinonaktifkan.

Validasi

- Wajib: Toggle yang memastikan pengguna harus memasukkan nilai waktu sebelum mengirimkan formulir.
- Kustom: Memungkinkan Anda membuat aturan validasi kustom menggunakan JavaScript ekspresi.

  Pesan Validasi Kustom: Pesan yang akan ditampilkan saat validasi kustom tidak terpenuhi.

Sebagai contoh:

```
{{ui.timeInput.value === "09:00 AM" || ui.timeInput.value === "09:30 AM"}}
```

Ekspresi ini memeriksa apakah waktu yang dimasukkan adalah 9:00 AM atau 9:30 AM. Jika kondisinya benar, validasi akan gagal.

Anda juga dapat memberikan pesan validasi kustom untuk ditampilkan ketika validasi tidak terpenuhi:

```
Validation not met. The time must be 9:00 AM or 9:30 AM.
```

Ekspresi dan contoh

Komponen Time menyediakan bidang ekspresi berikut:

- `{{ui.timeID.value}}`: Mengembalikan nilai waktu yang dimasukkan oleh pengguna dalam format HH: MM AA.

Contoh: Nilai waktu

- `{{ui.timeID.value}}`: Mengembalikan nilai waktu yang dimasukkan oleh pengguna dalam format`HH:MM AA`.

Contoh: Perbandingan waktu

- `{{ui.timeInput.value > "10:00 AM"}}`: Memeriksa apakah nilai waktu lebih besar dari 10:00 AM.
- `{{ui.timeInput.value < "05:00 pM"}}`: Memeriksa apakah nilai waktu kurang dari 05:00 PM.

Rentang tanggal

Komponen rentang Tanggal memungkinkan pengguna untuk memilih dan memasukkan rentang tanggal. Dengan mengonfigurasi berbagai properti komponen Rentang Tanggal, Anda dapat membuat kolom input rentang tanggal yang memenuhi persyaratan spesifik aplikasi Anda, seperti membatasi rentang tanggal yang dapat dipilih, menonaktifkan tanggal tertentu, dan mengontrol visibilitas dan interaktivitas komponen.

Properti rentang tanggal

Komponen Rentang Tanggal berbagi beberapa properti umum dengan komponen lain, seperti`Name`,`Source`, dan`Validation`. Untuk informasi lebih lanjut tentang properti ini, lihatProperti komponen umum.

Selain properti umum, komponen Rentang Tanggal memiliki properti spesifik berikut:

format

- MM/DD/YYYY: Format di mana rentang tanggal harus ditampilkan.

Tanggal mulai

- YYYY-MM-DD: Tanggal minimum yang dapat dipilih sebagai awal rentang.

> **Note**
>
> Nilai ini harus sesuai dengan format`YYYY-MM-DD`.

Tanggal akhir

- YYYY-MM-DD: Tanggal maksimum yang dapat dipilih sebagai akhir rentang.

> **Note**
>
> Nilai ini harus sesuai dengan format`YYYY-MM-DD`.

Placeholder

- Pengaturan kalender: Teks placeholder ditampilkan saat bidang rentang tanggal kosong.

Tanggal Min

- YYYY-MM-DD: Tanggal minimum yang dapat dipilih.

> **Note**
>
> Nilai ini harus sesuai dengan format`YYYY-MM-DD`.

Tanggal maks

- YYYY-MM-DD: Tanggal maksimum yang dapat dipilih.

> **Note**
>
> Nilai ini harus sesuai dengan format`YYYY-MM-DD`.

Jenis kalender

- 1 Bulan: Jenis UI kalender yang akan ditampilkan. Misalnya, satu bulan.
- 2 Bulan: Jenis UI kalender yang akan ditampilkan. Misalnya, dua bulan.

Hari wajib dipilih

- 0: Jumlah hari wajib yang harus dipilih dalam rentang tanggal.

Tanggal dinonaktifkan

- Sumber: Sumber data untuk tanggal yang harus dinonaktifkan (misalnya, Tidak Ada, Ekspresi, Entitas, atau Otomasi).
- Tanggal dinonaktifkan: Ekspresi yang menentukan tanggal mana yang harus dinonaktifkan, seperti`{{currentRow.column}}`.

Validasi

Bagian Validasi memungkinkan Anda untuk menentukan aturan dan batasan tambahan untuk input rentang tanggal.

Ekspresi dan contoh

Komponen Rentang Tanggal menyediakan bidang ekspresi berikut:

- `{{ui.dateRangeID.startDate}}`: Mengembalikan tanggal mulai dari rentang yang dipilih dalam format`YYYY-MM-DD`.
- `{{ui.dateRangeID.endDate}}`: Mengembalikan tanggal akhir rentang yang dipilih dalam format`YYYY-MM-DD`.

Contoh: Menghitung perbedaan tanggal

- `{{(new Date(ui.dateRangeID.endDate) - new Date(ui.dateRangeID.startDate)) / (1000 * 60 * 60 * 24)}}`Menghitung jumlah hari antara tanggal mulai dan akhir.

Contoh: Visibilitas bersyarat berdasarkan rentang tanggal

- `{{new Date(ui.dateRangeID.startDate) < new Date("2023-01-01") || new Date(ui.dateRangeID.endDate) > new Date("2023-12-31")}}`Memeriksa apakah rentang tanggal yang dipilih berada di luar tahun 2023.

Contoh: Tanggal dinonaktifkan berdasarkan data baris saat ini

- `{{currentRow.isHoliday}}`Menonaktifkan tanggal di mana kolom “isHoliday” di baris saat ini benar.
- `{{new Date(currentRow.dateColumn) < new Date("2023-01-01")}}`Menonaktifkan tanggal sebelum 1 Januari 2023 berdasarkan “DateColumn” di baris saat ini.
- `{{new Date(currentRow.dateColumn).getDay() === 0 || new Date(currentRow.dateColumn).getDay() === 6}}`Menonaktifkan akhir pekan berdasarkan “DateColumn” di baris saat ini.

Validasi kustom

- `{{new Date(ui.dateRangeID.startDate) > new Date(ui.dateRangeID.endDate)}}`Memeriksa apakah tanggal mulai lebih lambat dari tanggal akhir, yang akan gagal validasi kustom.

Komponen media

Studio aplikasi menyediakan beberapa komponen untuk menyematkan dan menampilkan berbagai jenis media dalam aplikasi Anda.

Sematan iFrame

Komponen embed iFrame memungkinkan Anda untuk menyematkan konten web eksternal atau aplikasi dalam aplikasi Anda menggunakan iFrame.

Properti embed iFrame

URL

> **Note**
>
> Sumber media yang ditampilkan dalam komponen ini harus diizinkan dalam pengaturan keamanan konten aplikasi Anda. Untuk informasi selengkapnya, lihat Melihat atau memperbarui setelan keamanan konten aplikasi Anda.

URL konten eksternal atau aplikasi yang ingin Anda sematkan.

Tata Letak

- Lebar: Lebar iFrame, ditentukan sebagai persentase (%) atau nilai piksel tetap (misalnya, 300px).
- Tinggi: Tinggi iFrame, ditentukan sebagai persentase (%) atau nilai piksel tetap.

Unggah S3

Komponen unggahan S3 memungkinkan pengguna untuk mengunggah file ke bucket Amazon S3. Dengan mengonfigurasi komponen Unggah S3, Anda dapat memungkinkan pengguna untuk dengan mudah mengunggah file ke penyimpanan Amazon S3 aplikasi Anda, dan kemudian memanfaatkan informasi file yang diunggah dalam logika dan antarmuka pengguna aplikasi Anda.

> **Note**
>
> Ingatlah untuk memastikan bahwa izin yang diperlukan dan konfigurasi bucket Amazon S3 tersedia untuk mendukung unggahan file dan persyaratan penyimpanan aplikasi Anda.

Properti unggah S3

Konfigurasi S3

- Konektor: Pilih konektor Amazon S3 yang telah dikonfigurasi sebelumnya untuk digunakan untuk unggahan file.
- Bucket: Bucket Amazon S3 tempat file akan diunggah.
- Folder: Folder di dalam ember Amazon S3 tempat file akan disimpan.
- Nama file: Konvensi penamaan untuk file yang diunggah.

Konfigurasi unggahan file

- Label: Label atau instruksi yang ditampilkan di atas area unggahan file.
- Deskripsi: Petunjuk atau informasi tambahan tentang unggahan file.
- Jenis file: Jenis file yang diizinkan untuk diunggah. Misalnya: gambar, dokumen, atau video.
- Ukuran: Ukuran maksimum file individual yang dapat diunggah.
- Label tombol: Teks yang ditampilkan pada tombol pemilihan file.
- Gaya tombol: Gaya tombol pemilihan file. Misalnya, diuraikan atau diisi.
- Ukuran tombol: Ukuran tombol pemilihan file.

Validasi

- Jumlah maksimum file: Jumlah maksimum file yang dapat diunggah sekaligus.
- Ukuran file maks: Ukuran maksimum yang diizinkan untuk setiap file individual.

Pemicu

- Saat berhasil: Tindakan yang akan dipicu saat unggahan file berhasil.
- Saat gagal: Tindakan yang akan dipicu saat unggahan file gagal.

Bidang ekspresi unggah S3

Komponen upload S3 menyediakan bidang ekspresi berikut:

- `{{ui.s3uploadID.files}}`: Mengembalikan array file yang telah diunggah.
- `{{ui.s3uploadID.files[0]?.size}}`: Mengembalikan ukuran file pada indeks yang ditunjuk.
- `{{ui.s3uploadID.files[0]?.type}}`: Mengembalikan jenis file pada indeks yang ditunjuk.
- `{{ui.s3uploadID.files[0]?.nameOnly}}`: Mengembalikan nama file, tanpa akhiran ekstensi, pada indeks yang ditunjuk.
- `{{ui.s3uploadID.files[0]?.nameWithExtension}}`: Mengembalikan nama file dengan akhiran ekstensi pada indeks yang ditunjuk.

Ekspresi dan contoh

Contoh: Mengakses file yang diunggah

- `{{ui.s3uploadID.files.length}}`: Mengembalikan jumlah file yang telah diunggah.
- `{{ui.s3uploadID.files.map(f => f.name).join(', ')}}`: Mengembalikan daftar dipisahkan koma dari nama file yang telah diunggah.
- `{{ui.s3uploadID.files.filter(f => f.type.startsWith('image/'))}}`: Mengembalikan array hanya file gambar yang telah diunggah.

Contoh: Memvalidasi unggahan file

- `{{ui.s3uploadID.files.some(f => f.size > 5 * 1024 * 1024)}}`: Memeriksa apakah ada file yang diunggah melebihi ukuran 5 MB.
- `{{ui.s3uploadID.files.every(f => f.type === 'image/png')}}`: Memeriksa apakah semua file yang diunggah adalah gambar PNG.
- `{{ui.s3uploadID.files.length > 3}}`: Memeriksa apakah lebih dari 3 file telah diunggah.

Contoh: Memicu tindakan

- `{{ui.s3uploadID.files.length > 0 ? 'Upload Successful' : 'No files uploaded'}}`: Menampilkan pesan sukses jika setidaknya satu file telah diunggah.

- `{{ui.s3uploadID.files.some(f => f.type.startsWith('video/')) ? triggerVideoProcessing() : null}}`: Memicu otomatisasi pemrosesan video jika ada file video yang telah diunggah.
- `{{ui.s3uploadID.files.map(f => f.url)}}`: Mengambil file URLs yang diunggah, yang dapat digunakan untuk menampilkan atau memproses file lebih lanjut.

Ekspresi ini memungkinkan Anda mengakses file yang diunggah, memvalidasi unggahan file, dan memicu tindakan berdasarkan hasil unggahan file. Dengan memanfaatkan ekspresi ini, Anda dapat membuat perilaku yang lebih dinamis dan cerdas dalam fungsionalitas unggah file aplikasi Anda.

> **Note**
>
> Ganti *s3uploadID* dengan ID komponen upload S3 Anda.

Komponen penampil PDF

Komponen penampil PDF memungkinkan pengguna untuk melihat dan berinteraksi dengan dokumen PDF dalam aplikasi Anda. App Studio mendukung jenis input yang berbeda ini untuk Sumber PDF, komponen penampil PDF memberikan fleksibilitas dalam cara Anda dapat mengintegrasikan dokumen PDF ke dalam aplikasi Anda, apakah itu dari URL statis, URI data inline, atau konten yang dihasilkan secara dinamis.

Properti penampil PDF

Sumber

> **Note**
>
> Sumber media yang ditampilkan dalam komponen ini harus diizinkan dalam pengaturan keamanan konten aplikasi Anda. Untuk informasi selengkapnya, lihat Melihat atau memperbarui setelan keamanan konten aplikasi Anda.

Sumber dokumen PDF, yang dapat berupa ekspresi, entitas, URL, atau otomatisasi.

Ekspresi

Gunakan ekspresi untuk menghasilkan sumber PDF secara dinamis.

Entitas

Hubungkan komponen penampil PDF ke entitas data yang berisi dokumen PDF.

URL

Tentukan URL dokumen PDF.

URL

Anda dapat memasukkan URL yang mengarah ke dokumen PDF yang ingin Anda tampilkan. Ini bisa berupa URL web publik atau URL dalam aplikasi Anda sendiri.

Contoh: `https://example.com/document.pdf`

URI Data

URI Data adalah cara ringkas untuk menyertakan file data kecil (seperti gambar atau PDFs) sebaris dalam aplikasi Anda. Dokumen PDF dikodekan sebagai string base64 dan disertakan langsung dalam konfigurasi komponen.

Gumpalan atau ArrayBuffer

Anda juga dapat memberikan dokumen PDF sebagai Blob atau ArrayBuffer objek, yang memungkinkan Anda untuk secara dinamis menghasilkan atau mengambil data PDF dari berbagai sumber dalam aplikasi Anda.

Otomatisasi

Hubungkan komponen penampil PDF ke otomatisasi yang menyediakan dokumen PDF.

Tindakan

- Unduh: Menambahkan tombol atau tautan yang memungkinkan pengguna mengunduh dokumen PDF.

Tata Letak

- Lebar: Lebar PDF Viewer, ditentukan sebagai persentase (%) atau nilai piksel tetap (misalnya, 600px).

- Tinggi: Ketinggian PDF Viewer, ditentukan sebagai nilai piksel tetap.

Penampil gambar

Komponen penampil gambar memungkinkan pengguna untuk melihat dan berinteraksi dengan file gambar dalam aplikasi Anda.

Properti penampil gambar

Sumber

> **Note**
>
> Sumber media yang ditampilkan dalam komponen ini harus diizinkan dalam pengaturan keamanan konten aplikasi Anda. Untuk informasi selengkapnya, lihat Melihat atau memperbarui setelan keamanan konten aplikasi Anda.

- Entity: Connect komponen Image viewer ke entitas data yang berisi file gambar.
- URL: Tentukan URL file gambar.
- Ekspresi: Gunakan ekspresi untuk menghasilkan sumber gambar secara dinamis.
- Otomasi: Hubungkan komponen penampil gambar ke otomatisasi yang menyediakan file gambar.

Teks Alt

Deskripsi teks alternatif dari gambar, yang digunakan untuk tujuan aksesibilitas.

Tata Letak

- Image fit: Menentukan bagaimana gambar harus diubah ukurannya dan ditampilkan di dalam komponen. Misalnya: `Contain`, `Cover`, atau `Fill`.
- Lebar: Lebar komponen penampil gambar, ditentukan sebagai persentase (%) atau nilai piksel tetap (misalnya, 300px).
- Tinggi: Ketinggian komponen penampil gambar, ditentukan sebagai nilai piksel tetap.
- Latar Belakang: Memungkinkan Anda untuk mengatur warna latar belakang atau gambar untuk komponen penampil Gambar.

Mendefinisikan dan menerapkan logika bisnis aplikasi Anda dengan otomatisasi

Otomatisasi adalah bagaimana Anda mendefinisikan logika bisnis aplikasi Anda. Komponen utama otomatisasi adalah: pemicu yang memulai otomatisasi, urutan satu atau lebih tindakan, parameter input yang digunakan untuk meneruskan data ke otomatisasi, dan output.

Topik

Konsep otomatisasi

Berikut adalah beberapa konsep dan istilah yang perlu diketahui saat mendefinisikan dan mengonfigurasi logika bisnis aplikasi Anda menggunakan otomatisasi di App Studio.

Otomatisasi

Otomatisasi adalah bagaimana Anda mendefinisikan logika bisnis aplikasi Anda. Komponen utama otomatisasi adalah: pemicu yang memulai otomatisasi, urutan satu atau lebih tindakan, parameter input yang digunakan untuk meneruskan data ke otomatisasi, dan output.

Tindakan

Tindakan otomatisasi, yang biasa disebut sebagai tindakan, adalah langkah logika individual yang membentuk otomatisasi. Setiap tindakan melakukan tugas tertentu, baik itu mengirim email, membuat catatan data, menjalankan fungsi Lambda, atau memanggil. APIs Tindakan ditambahkan ke otomatisasi dari pustaka tindakan, dan dapat dikelompokkan ke dalam pernyataan atau loop bersyarat.

Parameter masukan otomatisasi

Parameter input otomatisasi adalah nilai input dinamis yang dapat Anda teruskan dari komponen ke otomatisasi untuk membuatnya fleksibel dan dapat digunakan kembali. Pikirkan parameter sebagai variabel untuk otomatisasi Anda, alih-alih nilai hard-coding ke dalam otomatisasi, Anda dapat menentukan parameter dan memberikan nilai yang berbeda bila diperlukan. Parameter

memungkinkan Anda untuk menggunakan otomatisasi yang sama dengan input yang berbeda setiap kali dijalankan.

Output yang diejek

Beberapa tindakan berinteraksi dengan sumber daya atau layanan eksternal menggunakan konektor. Saat menggunakan lingkungan pratinjau, aplikasi tidak berinteraksi dengan layanan eksternal. Untuk menguji tindakan yang menggunakan konektor di lingkungan pratinjau, Anda dapat menggunakan keluaran tiruan untuk mensimulasikan perilaku dan keluaran konektor. Output tiruan dikonfigurasi menggunakan JavaScript, dan hasilnya disimpan dalam hasil tindakan, sama seperti respons konektor disimpan dalam aplikasi yang diterbitkan.

Dengan menggunakan mocking, Anda dapat menggunakan lingkungan pratinjau untuk menguji berbagai skenario dan dampaknya terhadap tindakan lain dengan otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus tepi, atau jalur tidak bahagia tanpa memanggil layanan eksternal melalui konektor.

Output otomatisasi

Output otomatisasi digunakan untuk meneruskan nilai dari satu otomatisasi ke sumber daya aplikasi lainnya, seperti komponen atau otomatisasi lainnya. Output otomatisasi dikonfigurasi sebagai ekspresi, dan ekspresi dapat mengembalikan nilai statis atau nilai dinamis yang dihitung dari parameter dan tindakan otomatisasi. Secara default, otomatisasi tidak mengembalikan data apa pun, termasuk hasil tindakan dalam otomatisasi.

Beberapa contoh bagaimana output otomatisasi dapat digunakan:

- Anda dapat mengonfigurasi output otomatisasi untuk mengembalikan array, dan meneruskan array itu untuk mengisi komponen data.
- Anda dapat menggunakan otomatisasi untuk menghitung nilai, dan meneruskan nilai itu ke beberapa otomatisasi lain sebagai cara untuk memusatkan dan menggunakan kembali logika bisnis.

Pemicu

Pemicu menentukan kapan, dan pada kondisi apa, otomatisasi akan berjalan. Beberapa contoh pemicu adalah `On click` untuk tombol dan `On select` untuk input teks. Jenis komponen menentukan daftar pemicu yang tersedia untuk komponen tersebut. Pemicu ditambahkan ke komponen dan dikonfigurasi di studio aplikasi.

Membuat, mengedit, dan menghapus otomatisasi

Daftar Isi

Membuat otomatisasi

Gunakan prosedur berikut untuk membuat otomatisasi dalam aplikasi App Studio. Setelah dibuat, otomatisasi harus dikonfigurasi dengan mengedit propertinya dan menambahkan tindakan ke dalamnya.

Untuk membuat otomatisasi

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.
2. Pilih tab Otomasi.
3. Jika Anda tidak memiliki otomatisasi, pilih + Tambahkan otomatisasi di kanvas. Jika tidak, di menu Otomatisasi sisi kiri, pilih + Tambah.
4. Otomatisasi baru akan dibuat, dan Anda dapat mulai mengedit propertinya atau menambahkan dan mengonfigurasi tindakan untuk menentukan logika bisnis aplikasi Anda.

Melihat atau mengedit properti otomatisasi

Gunakan prosedur berikut untuk melihat atau mengedit properti otomatisasi.

Untuk melihat atau mengedit properti otomatisasi

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.
2. Pilih tab Otomasi.
3. Di menu Automations sebelah kiri, pilih otomatisasi yang ingin Anda lihat atau edit properti untuk membuka menu Properties sisi kanan.
4. Di menu Properties, Anda dapat melihat properti berikut:

    - Pengidentifikasi otomatisasi: Nama unik otomatisasi. Untuk mengeditnya, masukkan pengenal baru di bidang teks.

- Parameter otomatisasi: Parameter otomatisasi digunakan untuk meneruskan nilai dinamis dari UI aplikasi Anda ke otomatisasi dan tindakan data. Untuk menambahkan parameter, pilih + Tambah. Pilih ikon pensil untuk mengubah nama, deskripsi, atau jenis parameter. Untuk menghapus parameter, pilih ikon sampah.

  > **Tip**
  >
  > Anda juga dapat menambahkan parameter otomatisasi langsung dari kanvas.

- Output otomatisasi: Output otomatisasi digunakan untuk mengonfigurasi data mana dari otomatisasi yang dapat direferensikan dalam otomatisasi atau komponen lain. Secara default, otomatisasi tidak membuat output. Untuk menambahkan output otomatisasi pilih + Tambah. Untuk menghapus output, pilih ikon sampah.

5. Anda menentukan apa yang dilakukan otomatisasi dengan menambahkan dan mengonfigurasi tindakan. Untuk informasi selengkapnya tentang tindakan, lihat Menambahkan, mengedit, dan menghapus tindakan otomatisasi danReferensi tindakan otomatis.

Menghapus otomatisasi

Gunakan prosedur berikut untuk menghapus otomatisasi dalam aplikasi App Studio.

Untuk menghapus otomatisasi

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.
2. Pilih tab Otomasi.
3. Di menu Otomatisasi sisi kiri, pilih menu elips dari otomatisasi yang ingin Anda hapus, dan pilih Hapus. Atau, Anda dapat memilih ikon sampah dari menu Properti sisi kanan otomatisasi.
4. Di kotak dialog konfirmasi, pilih Hapus.

Menambahkan, mengedit, dan menghapus tindakan otomatisasi

Tindakan otomatisasi, yang biasa disebut sebagai tindakan, adalah langkah logika individual yang membentuk otomatisasi. Setiap tindakan melakukan tugas tertentu, baik itu mengirim email, membuat catatan data, menjalankan fungsi Lambda, atau memanggil. APIs Tindakan ditambahkan ke otomatisasi dari pustaka tindakan, dan dapat dikelompokkan ke dalam pernyataan atau loop bersyarat.

Daftar Isi

Menambahkan tindakan otomatisasi

Gunakan prosedur berikut untuk menambahkan tindakan ke otomatisasi dalam aplikasi App Studio.

Untuk menambahkan tindakan otomatisasi

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.
2. Pilih tab Otomasi.
3. Di menu Otomatisasi sisi kiri, pilih otomatisasi yang ingin Anda tambahkan tindakan.
4. Di menu Tindakan sebelah kanan, pilih tindakan yang ingin Anda tambahkan, atau seret dan jatuhkan tindakan ke kanvas. Setelah tindakan dibuat, Anda dapat memilih tindakan untuk mengonfigurasi properti tindakan untuk menentukan fungsionalitas tindakan. Untuk informasi selengkapnya tentang properti tindakan dan mengonfigurasinya, lihatReferensi tindakan otomatis.

Melihat dan mengedit properti tindakan otomatisasi

Gunakan prosedur berikut untuk melihat atau mengedit properti tindakan otomatisasi di aplikasi App Studio.

Untuk melihat atau mengedit properti tindakan otomatisasi

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.
2. Pilih tab Otomasi.
3. Di menu Otomatisasi sisi kiri, pilih tindakan yang ingin Anda lihat atau edit properti. Atau, Anda dapat memilih tindakan di kanvas saat melihat otomatisasi yang mengandungnya.
4. Anda dapat melihat atau mengedit properti tindakan di menu Properti sisi kanan. Properti untuk tindakan berbeda untuk setiap jenis tindakan. Untuk informasi selengkapnya tentang properti tindakan dan mengonfigurasinya, lihatReferensi tindakan otomatis.

Menghapus tindakan otomatisasi

Gunakan prosedur berikut untuk menghapus tindakan dari otomatisasi di aplikasi App Studio.

Untuk menghapus tindakan otomatisasi

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.
2. Pilih tab Otomasi.
3. Di menu Otomatisasi sisi kiri, pilih otomatisasi yang berisi tindakan yang ingin Anda hapus.
4. Di kanvas, pilih ikon sampah dalam tindakan yang ingin Anda hapus dan pilih Hapus.

Referensi tindakan otomatis

Berikut ini adalah dokumentasi referensi untuk tindakan otomatisasi yang digunakan di App Studio.

Tindakan otomatisasi, yang biasa disebut sebagai tindakan, adalah langkah logika individual yang membentuk otomatisasi. Setiap tindakan melakukan tugas tertentu, baik itu mengirim email, membuat catatan data, menjalankan fungsi Lambda, atau memanggil. APIs Tindakan ditambahkan ke otomatisasi dari pustaka tindakan, dan dapat dikelompokkan ke dalam pernyataan atau loop bersyarat.

Untuk informasi tentang membuat dan mengonfigurasi otomatisasi dan tindakannya, lihat topik di. Mendefinisikan dan menerapkan logika bisnis aplikasi Anda dengan otomatisasi

Memanggil API

Memanggil permintaan HTTP REST API. Builder dapat menggunakan tindakan ini untuk mengirim permintaan dari App Studio ke sistem atau layanan lain APIs. Misalnya, Anda dapat menggunakannya untuk terhubung ke sistem pihak ketiga atau aplikasi lokal untuk mengakses data penting bisnis, atau memanggil titik akhir API yang tidak dapat dipanggil oleh tindakan App Studio khusus.

Untuk informasi selengkapnya tentang REST APIs, lihat Apa itu RESTful API? .

Properti

Konektor

Konektor yang akan digunakan untuk permintaan API yang dibuat oleh tindakan ini. Dropdown konektor hanya berisi konektor dari jenis berikut: `API Connector` dan. `OpenAPI Connector`

Bergantung pada bagaimana konektor dikonfigurasi, konektor dapat berisi informasi penting seperti kredensional dan header default atau parameter kueri.

Untuk informasi selengkapnya tentang konektor API, termasuk perbandingan antara penggunaan `API Connector` dan`OpenAPI Connector`, lihatConnect ke layanan pihak ketiga.

Properti konfigurasi permintaan API

Pilih Konfigurasi permintaan API dari panel properti untuk membuka kotak dialog konfigurasi permintaan. Jika konektor API dipilih, kotak dialog akan menyertakan informasi konektor.

Metode: Metode untuk panggilan API. Kemungkinan nilainya adalah sebagai berikut:

- `DELETE`: Menghapus sumber daya tertentu.
- `GET`: Mengambil informasi atau data.
- `HEAD`: Mengambil hanya header respons tanpa tubuh.
- `POST`: Mengirimkan data untuk diproses.
- `PUSH`: Mengirimkan data untuk diproses.
- `PATCH`: Sebagian memperbarui sumber daya tertentu.

Path: Jalur relatif ke sumber daya.

Header: Setiap header dalam bentuk pasangan kunci-nilai yang akan dikirim dengan permintaan API. Jika konektor dipilih, header yang dikonfigurasi akan ditambahkan secara otomatis dan tidak dapat dihapus. Header yang dikonfigurasi tidak dapat diedit, tetapi Anda dapat menggantinya dengan menambahkan header lain dengan nama yang sama.

Parameter kueri: Parameter kueri apa pun dalam bentuk pasangan nilai kunci yang akan dikirim bersama permintaan API. Jika konektor dipilih, parameter kueri yang dikonfigurasi akan ditambahkan secara otomatis dan tidak dapat diedit atau dihapus.

Body: Informasi yang akan dikirim dengan permintaan API dalam format JSON. Tidak ada badan untuk `GET` permintaan.

Output yang diejek

Tindakan tidak berinteraksi dengan layanan atau sumber daya eksternal di lingkungan pratinjau. Bidang keluaran Mocked digunakan untuk menyediakan ekspresi JSON yang mensimulasikan

perilaku konektor di lingkungan pratinjau untuk tujuan pengujian. Cuplikan ini disimpan di `results` peta tindakan, sama seperti respons konektor untuk aplikasi yang dipublikasikan di lingkungan langsung.

Dengan bidang ini, Anda dapat menguji berbagai skenario dan dampaknya terhadap tindakan lain dalam otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus tepi, atau jalur yang tidak menyenangkan tanpa berkomunikasi dengan layanan eksternal melalui konektor.

Memohon AWS

Memanggil operasi dari AWS layanan. Ini adalah tindakan umum untuk memanggil AWS layanan atau operasi, dan harus digunakan jika tidak ada tindakan khusus untuk AWS layanan atau operasi yang diinginkan.

Properti

Layanan

AWS Layanan yang berisi operasi yang akan dijalankan.

Operasi

Operasi yang harus dijalankan.

Konektor

Konektor yang akan digunakan untuk operasi yang dijalankan oleh tindakan ini. Konektor yang dikonfigurasi harus diatur dengan kredensional yang tepat untuk menjalankan operasi, dan informasi konfigurasi lainnya, seperti AWS wilayah yang berisi sumber daya apa pun yang direferensikan dalam operasi.

Konfigurasi

Input JSON menjadi ketika menjalankan operasi yang ditentukan. Untuk informasi selengkapnya tentang mengonfigurasi input untuk AWS operasi, lihat. AWS SDK for JavaScript

Memanggil Lambda

Memanggil fungsi Lambda yang ada.

Properti

Konektor

Konektor yang akan digunakan untuk fungsi Lambda dijalankan oleh tindakan ini. Konektor yang dikonfigurasi harus diatur dengan kredensional yang tepat untuk mengakses fungsi Lambda, dan informasi konfigurasi lainnya, seperti AWS wilayah yang berisi fungsi Lambda. Untuk informasi selengkapnya tentang mengonfigurasi konektor untuk Lambda, lihat. Langkah 3: Buat konektor Lambda

Nama fungsi

Nama fungsi Lambda yang akan dijalankan. Perhatikan bahwa ini adalah nama fungsi, dan bukan fungsi ARN (Amazon Resource Name).

Acara fungsi

Pasangan nilai kunci yang akan diteruskan ke fungsi Lambda Anda sebagai muatan acara.

Output yang diejek

Tindakan tidak berinteraksi dengan layanan atau sumber daya eksternal di lingkungan pratinjau. Bidang keluaran Mocked digunakan untuk menyediakan ekspresi JSON yang mensimulasikan perilaku konektor di lingkungan pratinjau untuk tujuan pengujian. Cuplikan ini disimpan di `results` peta tindakan, sama seperti respons konektor untuk aplikasi yang dipublikasikan di lingkungan langsung.

Dengan bidang ini, Anda dapat menguji berbagai skenario dan dampaknya terhadap tindakan lain dalam otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus tepi, atau jalur yang tidak menyenangkan tanpa berkomunikasi dengan layanan eksternal melalui konektor.

Loop

Menjalankan tindakan bersarang berulang kali untuk mengulang melalui daftar item, satu item pada satu waktu. Misalnya, tambahkan Buat catatan tindakan ke tindakan loop untuk membuat beberapa catatan.

Tindakan loop dapat disarangkan dalam loop lain atau tindakan kondisi. Tindakan loop dijalankan secara berurutan, dan tidak secara paralel. Hasil dari setiap tindakan dalam loop hanya dapat

diakses ke tindakan berikutnya dalam iterasi loop yang sama. Mereka tidak dapat diakses di luar loop atau dalam iterasi loop yang berbeda.

Properti

Sumber

Daftar item untuk diulang, satu item pada satu waktu. Sumber dapat berupa hasil dari tindakan sebelumnya atau daftar statis string, angka, atau objek yang dapat Anda berikan menggunakan JavaScript ekspresi.

Contoh

Daftar berikut berisi contoh input sumber.

- Hasil dari tindakan sebelumnya: `{{results.actionName.data}}`
- Daftar angka: `{{[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]}}`
- Daftar string: `{{["apple", "banana", "orange", "grape", "kiwi"]}}`
- Nilai yang dihitung: `{{params.actionName.split("\n")}}`

Nama item saat ini

Nama variabel yang dapat digunakan untuk mereferensikan item saat ini yang sedang diulang. Nama item saat ini dapat dikonfigurasi sehingga Anda dapat membuat sarang dua atau lebih loop dan mengakses variabel dari setiap loop. Misalnya, jika Anda melakukan looping melalui negara dan kota dengan dua loop, Anda dapat mengonfigurasi dan mereferensikan `currentCountry` dan. `currentCity`

Ketentuan

Menjalankan tindakan berdasarkan hasil dari satu atau lebih kondisi logis tertentu yang dievaluasi ketika otomatisasi dijalankan. Tindakan kondisi terdiri dari komponen-komponen berikut:

- Bidang kondisi, yang digunakan untuk memberikan JavaScript ekspresi yang mengevaluasi `true` atau`false`.
- Cabang yang benar, yang berisi tindakan yang dijalankan jika kondisi berubah menjadi. `true`
- Cabang palsu, yang berisi tindakan yang dijalankan jika kondisi berubah menjadi. `false`

Tambahkan tindakan ke cabang benar dan salah dengan menyeretnya ke dalam tindakan kondisi.

Properti

Ketentuan

JavaScript Ekspresi yang akan dievaluasi ketika tindakan dijalankan.

Buat catatan

Membuat satu rekaman di entitas App Studio yang ada.

Properti

Entitas

Entitas di mana catatan akan dibuat. Setelah entitas dipilih, nilai harus ditambahkan ke bidang entitas untuk catatan yang akan dibuat. Jenis bidang, dan jika bidang diperlukan atau opsional didefinisikan dalam entitas.

Perbarui catatan

Memperbarui rekaman yang ada di entitas App Studio.

Properti

Entitas

Entitas yang berisi catatan yang akan diperbarui.

Ketentuan

Kriteria yang menentukan catatan mana yang diperbarui oleh tindakan. Anda dapat mengelompokkan kondisi untuk membuat satu pernyataan logis. Anda dapat menggabungkan grup atau kondisi dengan `AND` atau `OR` pernyataan.

Bidang

Bidang yang akan diperbarui dalam catatan yang ditentukan oleh kondisi.

Nilai

Nilai yang akan diperbarui di bidang yang ditentukan.

Hapus catatan

Menghapus rekaman dari entitas App Studio.

Properti

Entitas

Entitas yang berisi catatan yang akan dihapus.

Ketentuan

Kriteria yang menentukan catatan mana yang dihapus oleh tindakan. Anda dapat mengelompokkan kondisi untuk membuat satu pernyataan logika. Anda dapat menggabungkan grup atau kondisi dengan `AND` atau `OR` pernyataan.

Memanggil tindakan data

Menjalankan tindakan data dengan parameter opsional.

Properti

Tindakan data

Tindakan data yang akan dijalankan oleh tindakan.

Parameter

Parameter tindakan data yang akan digunakan oleh tindakan data. Parameter tindakan data digunakan untuk mengirim nilai yang digunakan sebagai input untuk tindakan data. Parameter tindakan data dapat ditambahkan saat mengonfigurasi tindakan otomatisasi, tetapi harus diedit di tab Data.

Pengaturan lanjutan

`Invoke data action`Tindakan ini berisi pengaturan lanjutan berikut:

- Ukuran halaman: Jumlah maksimum catatan untuk diambil dalam setiap kueri. Nilai default adalah 500, dan nilai maksimumnya adalah 3000.
- Token pagination: Token yang digunakan untuk mengambil catatan tambahan dari kueri. Misalnya, jika `Page size` diatur ke 500, tetapi ada lebih dari 500 catatan, meneruskan token pagination ke

kueri berikutnya akan mengambil 500 berikutnya. Token akan tidak terdefinisi jika tidak ada lagi catatan atau halaman yang ada.

Amazon S3: Letakkan objek

Menggunakan `Amazon S3 PutObject` operasi untuk menambahkan objek yang diidentifikasi oleh kunci (jalur file) ke bucket Amazon S3 yang ditentukan.

Properti

Konektor

Konektor yang akan digunakan untuk operasi yang dijalankan oleh tindakan ini. Konektor yang dikonfigurasi harus diatur dengan kredensional yang sesuai untuk menjalankan operasi, dan informasi konfigurasi lainnya, seperti AWS wilayah yang berisi sumber daya apa pun yang direferensikan dalam operasi.

Konfigurasi

Opsi yang diperlukan untuk digunakan dalam `PutObject` perintah. Opsinya adalah sebagai berikut:

> **Note**
>
> Untuk informasi selengkapnya tentang `Amazon S3 PutObject` operasi, lihat PutObjectdi Referensi API Amazon Simple Storage Service.

- Bucket: Nama ember Amazon S3 untuk meletakkan objek.
- Kunci: Nama unik objek yang akan dimasukkan ke dalam ember Amazon S3.
- Tubuh: Isi objek yang akan dimasukkan ke dalam ember Amazon S3.

Output yang diejek

Tindakan tidak berinteraksi dengan layanan atau sumber daya eksternal di lingkungan pratinjau. Bidang keluaran Mocked digunakan untuk menyediakan ekspresi JSON yang mensimulasikan perilaku konektor di lingkungan pratinjau untuk tujuan pengujian. Cuplikan ini disimpan di `results` peta tindakan, sama seperti respons konektor untuk aplikasi yang dipublikasikan di lingkungan langsung.

Dengan bidang ini, Anda dapat menguji berbagai skenario dan dampaknya terhadap tindakan lain dalam otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus tepi, atau jalur yang tidak menyenangkan tanpa berkomunikasi dengan layanan eksternal melalui konektor.

Amazon S3: Hapus objek

Menggunakan `Amazon S3 DeleteObject` operasi untuk menghapus objek yang diidentifikasi oleh kunci (jalur file) dari bucket Amazon S3 yang ditentukan.

Properti

Konektor

Konektor yang akan digunakan untuk operasi yang dijalankan oleh tindakan ini. Konektor yang dikonfigurasi harus diatur dengan kredensional yang tepat untuk menjalankan operasi, dan informasi konfigurasi lainnya, seperti AWS wilayah yang berisi sumber daya apa pun yang direferensikan dalam operasi.

Konfigurasi

Opsi yang diperlukan untuk digunakan dalam `DeleteObject` perintah. Opsinya adalah sebagai berikut:

> **Note**
>
> Untuk informasi selengkapnya tentang `Amazon S3 DeleteObject` operasi, lihat DeleteObjectdi Referensi API Amazon Simple Storage Service.

- Bucket: Nama bucket Amazon S3 untuk menghapus objek.
- Kunci: Nama unik objek yang akan dihapus dari bucket Amazon S3.

Output yang diejek

Tindakan tidak berinteraksi dengan layanan atau sumber daya eksternal di lingkungan pratinjau. Bidang keluaran Mocked digunakan untuk menyediakan ekspresi JSON yang mensimulasikan perilaku konektor di lingkungan pratinjau untuk tujuan pengujian. Cuplikan ini disimpan di `results`

peta tindakan, sama seperti respons konektor untuk aplikasi yang dipublikasikan di lingkungan langsung.

Dengan bidang ini, Anda dapat menguji berbagai skenario dan dampaknya terhadap tindakan lain dalam otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus tepi, atau jalur yang tidak menyenangkan tanpa berkomunikasi dengan layanan eksternal melalui konektor.

Amazon S3: Dapatkan objek

Menggunakan `Amazon S3 GetObject` operasi untuk mengambil objek yang diidentifikasi oleh kunci (jalur file) dari bucket Amazon S3 yang ditentukan.

Properti

Konektor

Konektor yang akan digunakan untuk operasi yang dijalankan oleh tindakan ini. Konektor yang dikonfigurasi harus diatur dengan kredensional yang tepat untuk menjalankan operasi, dan informasi konfigurasi lainnya, seperti AWS wilayah yang berisi sumber daya apa pun yang direferensikan dalam operasi.

Konfigurasi

Opsi yang diperlukan untuk digunakan dalam `GetObject` perintah. Opsinya adalah sebagai berikut:

> **Note**
>
> Untuk informasi selengkapnya tentang `Amazon S3 GetObject` operasi, lihat GetObjectdi Referensi API Amazon Simple Storage Service.

- Bucket: Nama bucket Amazon S3 untuk mengambil objek.
- Kunci: Nama unik objek yang akan diambil dari bucket Amazon S3.

Output yang diejek

Tindakan tidak berinteraksi dengan layanan atau sumber daya eksternal di lingkungan pratinjau. Bidang keluaran Mocked digunakan untuk menyediakan ekspresi JSON yang mensimulasikan perilaku konektor di lingkungan pratinjau untuk tujuan pengujian. Cuplikan ini disimpan di `results`

peta tindakan, sama seperti respons konektor untuk aplikasi yang dipublikasikan di lingkungan langsung.

Dengan bidang ini, Anda dapat menguji berbagai skenario dan dampaknya terhadap tindakan lain dalam otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus tepi, atau jalur yang tidak menyenangkan tanpa berkomunikasi dengan layanan eksternal melalui konektor.

Amazon S3: Daftar objek

Menggunakan `Amazon S3 ListObjects` operasi untuk mencantumkan objek dalam bucket Amazon S3 yang ditentukan.

Properti

Konektor

Konektor yang akan digunakan untuk operasi yang dijalankan oleh tindakan ini. Konektor yang dikonfigurasi harus diatur dengan kredensional yang tepat untuk menjalankan operasi, dan informasi konfigurasi lainnya, seperti AWS wilayah yang berisi sumber daya apa pun yang direferensikan dalam operasi.

Konfigurasi

Opsi yang diperlukan untuk digunakan dalam `ListObjects` perintah. Opsinya adalah sebagai berikut:

> **Note**
>
> Untuk informasi selengkapnya tentang `Amazon S3 ListObjects` operasi, lihat ListObjectsdi Referensi API Amazon Simple Storage Service.

- Bucket: Nama bucket Amazon S3 untuk mencantumkan objek.

Output yang diejek

Tindakan tidak berinteraksi dengan layanan atau sumber daya eksternal di lingkungan pratinjau. Bidang keluaran Mocked digunakan untuk menyediakan ekspresi JSON yang mensimulasikan perilaku konektor di lingkungan pratinjau untuk tujuan pengujian. Cuplikan ini disimpan di `results`

peta tindakan, sama seperti respons konektor untuk aplikasi yang dipublikasikan di lingkungan langsung.

Dengan bidang ini, Anda dapat menguji berbagai skenario dan dampaknya terhadap tindakan lain dalam otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus tepi, atau jalur yang tidak menyenangkan tanpa berkomunikasi dengan layanan eksternal melalui konektor.

Amazon Texttract: Analisis dokumen

Menggunakan `Amazon Textract AnalyzeDocument` operasi untuk menganalisis dokumen masukan untuk hubungan antara item yang terdeteksi.

Properti

Konektor

Konektor yang akan digunakan untuk operasi yang dijalankan oleh tindakan ini. Konektor yang dikonfigurasi harus diatur dengan kredensional yang tepat untuk menjalankan operasi, dan informasi konfigurasi lainnya, seperti AWS wilayah yang berisi sumber daya apa pun yang direferensikan dalam operasi.

Konfigurasi

Isi permintaan yang akan digunakan dalam `AnalyzeDocument` perintah. Opsinya adalah sebagai berikut:

> **Note**
>
> Untuk informasi selengkapnya tentang `Amazon Textract AnalyzeDocument` operasi, lihat AnalyzeDocumentdi Panduan Pengembang Amazon Texttract.

- Document/S3Object/Bucket: Nama bucket Amazon S3. Parameter ini dapat dibiarkan kosong jika file diteruskan ke tindakan dengan komponen unggah S3.
- Document/S3Object/Name: Nama file dari dokumen input. Parameter ini dapat dibiarkan kosong jika file diteruskan ke tindakan dengan komponen unggah S3.
- Document/S3Object/Version: Jika bucket Amazon S3 mengaktifkan versi, Anda dapat menentukan versi objek. Parameter ini dapat dibiarkan kosong jika file diteruskan ke tindakan dengan komponen unggah S3.

- FeatureTypes: Daftar jenis analisis yang harus dilakukan. Nilai yang valid adalah: `TABLES`, `FORMS`, `QUERIES`, `SIGNATURES`, dan `LAYOUT`.

Output yang diejek

Tindakan tidak berinteraksi dengan layanan atau sumber daya eksternal di lingkungan pratinjau. Bidang keluaran Mocked digunakan untuk menyediakan ekspresi JSON yang mensimulasikan perilaku konektor di lingkungan pratinjau untuk tujuan pengujian. Cuplikan ini disimpan di `results` peta tindakan, sama seperti respons konektor untuk aplikasi yang dipublikasikan di lingkungan langsung.

Dengan bidang ini, Anda dapat menguji berbagai skenario dan dampaknya terhadap tindakan lain dalam otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus tepi, atau jalur yang tidak menyenangkan tanpa berkomunikasi dengan layanan eksternal melalui konektor.

Amazon Texttract: Analisis biaya

Menggunakan `Amazon Textract AnalyzeExpense` operasi untuk menganalisis dokumen masukan untuk hubungan terkait keuangan antar teks.

Properti

Konektor

Konektor yang akan digunakan untuk operasi yang dijalankan oleh tindakan ini. Konektor yang dikonfigurasi harus diatur dengan kredensional yang tepat untuk menjalankan operasi, dan informasi konfigurasi lainnya, seperti AWS wilayah yang berisi sumber daya apa pun yang direferensikan dalam operasi.

Konfigurasi

Isi permintaan yang akan digunakan dalam `AnalyzeExpense` perintah. Opsinya adalah sebagai berikut:

> **Note**
>
> Untuk informasi selengkapnya tentang `Amazon Textract AnalyzeExpense` operasi, lihat AnalyzeExpensedi Panduan Pengembang Amazon Texttract.

- Document/S3Object/Bucket: Nama bucket Amazon S3. Parameter ini dapat dibiarkan kosong jika file diteruskan ke tindakan dengan komponen unggah S3.
- Document/S3Object/Name: Nama file dari dokumen input. Parameter ini dapat dibiarkan kosong jika file diteruskan ke tindakan dengan komponen unggah S3.
- Document/S3Object/Version: Jika bucket Amazon S3 mengaktifkan versi, Anda dapat menentukan versi objek. Parameter ini dapat dibiarkan kosong jika file diteruskan ke tindakan dengan komponen unggah S3.

Output yang diejek

Tindakan tidak berinteraksi dengan layanan atau sumber daya eksternal di lingkungan pratinjau. Bidang keluaran Mocked digunakan untuk menyediakan ekspresi JSON yang mensimulasikan perilaku konektor di lingkungan pratinjau untuk tujuan pengujian. Cuplikan ini disimpan di `results` peta tindakan, sama seperti respons konektor untuk aplikasi yang dipublikasikan di lingkungan langsung.

Dengan bidang ini, Anda dapat menguji berbagai skenario dan dampaknya terhadap tindakan lain dalam otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus tepi, atau jalur yang tidak menyenangkan tanpa berkomunikasi dengan layanan eksternal melalui konektor.

Amazon Ttract: Analisis ID

Menggunakan `Amazon Textract AnalyzeID` operasi untuk menganalisis dokumen identitas untuk informasi yang relevan.

Properti

Konektor

Konektor yang akan digunakan untuk operasi yang dijalankan oleh tindakan ini. Konektor yang dikonfigurasi harus diatur dengan kredensional yang tepat untuk menjalankan operasi, dan informasi konfigurasi lainnya, seperti AWS wilayah yang berisi sumber daya apa pun yang direferensikan dalam operasi.

Konfigurasi

Isi permintaan yang akan digunakan dalam `AnalyzeID` perintah. Opsinya adalah sebagai berikut:

> **Note**
>
> Untuk informasi selengkapnya tentang `Amazon Textract AnalyzeID` operasi, lihat AnalyzeID di Panduan Pengembang Amazon Texttract.

- Document/S3Object/Bucket: Nama bucket Amazon S3. Parameter ini dapat dibiarkan kosong jika file diteruskan ke tindakan dengan komponen unggah S3.
- Document/S3Object/Name: Nama file dari dokumen input. Parameter ini dapat dibiarkan kosong jika file diteruskan ke tindakan dengan komponen unggah S3.
- Document/S3Object/Version: Jika bucket Amazon S3 mengaktifkan versi, Anda dapat menentukan versi objek. Parameter ini dapat dibiarkan kosong jika file diteruskan ke tindakan dengan komponen unggah S3.

Output yang diejek

Tindakan tidak berinteraksi dengan layanan atau sumber daya eksternal di lingkungan pratinjau. Bidang keluaran Mocked digunakan untuk menyediakan ekspresi JSON yang mensimulasikan perilaku konektor di lingkungan pratinjau untuk tujuan pengujian. Cuplikan ini disimpan di `results` peta tindakan, sama seperti respons konektor untuk aplikasi yang dipublikasikan di lingkungan langsung.

Dengan bidang ini, Anda dapat menguji berbagai skenario dan dampaknya terhadap tindakan lain dalam otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus tepi, atau jalur yang tidak menyenangkan tanpa berkomunikasi dengan layanan eksternal melalui konektor.

Amazon Textract: Mendeteksi teks dokumen

Menggunakan `Amazon Textract DetectDocumentText` operasi untuk mendeteksi baris teks dan kata-kata yang membentuk baris teks dalam dokumen input.

Properti

Konektor

Konektor yang akan digunakan untuk operasi yang dijalankan oleh tindakan ini. Konektor yang dikonfigurasi harus diatur dengan kredensional yang tepat untuk menjalankan operasi, dan informasi

konfigurasi lainnya, seperti AWS wilayah yang berisi sumber daya apa pun yang direferensikan dalam operasi.

Konfigurasi

Isi permintaan yang akan digunakan dalam `DetectDocumentText` perintah. Opsinya adalah sebagai berikut:

> **Note**
>
> Untuk informasi selengkapnya tentang `Amazon Textract DetectDocumentText` operasi, lihat DetectDocumentTextdi Panduan Pengembang Amazon Texttract.

- Document/S3Object/Bucket: Nama bucket Amazon S3. Parameter ini dapat dibiarkan kosong jika file diteruskan ke tindakan dengan komponen unggah S3.
- Document/S3Object/Name: Nama file dari dokumen input. Parameter ini dapat dibiarkan kosong jika file diteruskan ke tindakan dengan komponen unggah S3.
- Document/S3Object/Version: Jika bucket Amazon S3 mengaktifkan versi, Anda dapat menentukan versi objek. Parameter ini dapat dibiarkan kosong jika file diteruskan ke tindakan dengan komponen unggah S3.

Output yang diejek

Tindakan tidak berinteraksi dengan layanan atau sumber daya eksternal di lingkungan pratinjau. Bidang keluaran Mocked digunakan untuk menyediakan ekspresi JSON yang mensimulasikan perilaku konektor di lingkungan pratinjau untuk tujuan pengujian. Cuplikan ini disimpan di `results` peta tindakan, sama seperti respons konektor untuk aplikasi yang dipublikasikan di lingkungan langsung.

Dengan bidang ini, Anda dapat menguji berbagai skenario dan dampaknya terhadap tindakan lain dalam otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus tepi, atau jalur yang tidak menyenangkan tanpa berkomunikasi dengan layanan eksternal melalui konektor.

Batuan Dasar Amazon: GenAI Prompt

Menggunakan InvokeModel operasi Amazon Bedrock untuk menjalankan inferensi menggunakan parameter prompt dan inferensi yang disediakan dalam properti tindakan. Tindakan ini dapat menghasilkan teks, gambar, dan penyematan.

Properti

Konektor

Konektor yang akan digunakan untuk operasi yang dijalankan oleh tindakan ini. Agar berhasil menggunakan tindakan ini, konektor harus dikonfigurasi dengan Amazon Bedrock Runtime sebagai layanan. Konektor yang dikonfigurasi harus diatur dengan kredensional yang tepat untuk menjalankan operasi, dan informasi konfigurasi lainnya, seperti AWS wilayah yang berisi sumber daya apa pun yang direferensikan dalam operasi.

Model

Model dasar yang akan digunakan oleh Amazon Bedrock untuk memproses permintaan. Untuk informasi selengkapnya tentang model di Amazon Bedrock, lihat informasi model fondasi Amazon Bedrock di Panduan Pengguna Amazon Bedrock.

Jenis masukan

Jenis input dari input yang dikirim ke model Amazon Bedrock. Nilai yang mungkin adalah Teks, Dokumen, dan Gambar. Jika jenis input tidak tersedia untuk dipilih, kemungkinan besar tidak didukung oleh model yang dikonfigurasi.

Prompt pengguna

Prompt yang akan dikirim ke model Amazon Bedrock untuk diproses untuk menghasilkan respons. Anda dapat memasukkan teks statis, atau meneruskan input dari bagian lain aplikasi Anda, seperti dari komponen menggunakan parameter, tindakan sebelumnya dalam otomatisasi, atau otomatisasi lain. Contoh berikut menunjukkan cara meneruskan nilai dari komponen atau tindakan sebelumnya:

- Untuk meneruskan nilai dari komponen menggunakan parameter: `{{params.paramName}}`
- Untuk meneruskan nilai dari tindakan sebelumnya: `{{results.actionName}}`

Prompt sistem (model Claude)

Prompt sistem yang akan digunakan oleh model Amazon Bedrock saat memproses permintaan. Prompt sistem digunakan untuk memberikan konteks, instruksi, atau pedoman untuk model Claude.

Permintaan pengaturan

Konfigurasikan berbagai pengaturan permintaan dan parameter inferensi model. Anda dapat mengonfigurasi pengaturan berikut:

- Suhu: Suhu yang akan digunakan oleh model Amazon Bedrock saat memproses permintaan. Suhu menentukan keacakan atau kreativitas keluaran model Bedrock. Semakin tinggi suhunya, semakin kreatif dan kurang analitis responsnya. Nilai yang mungkin adalah`[0-10]`.
- Token Maks: Batasi panjang output model Amazon Bedrock.
- TopP: Dalam pengambilan sampel nukleus, model menghitung distribusi kumulatif atas semua opsi untuk setiap token berikutnya dalam urutan probabilitas yang menurun dan memotongnya setelah mencapai probabilitas tertentu yang ditentukan oleh TopP. Anda harus mengubah suhu atau TopP, tetapi tidak keduanya
- Stop Sequences: Urutan yang menyebabkan model berhenti memproses permintaan dan menghasilkan output.

Untuk informasi selengkapnya, lihat Parameter permintaan inferensi dan bidang respons untuk model foundation di Panduan Pengguna Amazon Bedrock.

Hentikan Urutan

Masukkan ID dan Versi Amazon Bedrock Guardrail. Pagar pembatas digunakan untuk menerapkan perlindungan berdasarkan kasus penggunaan Anda dan kebijakan AI yang bertanggung jawab. Untuk informasi selengkapnya, lihat Menghentikan konten berbahaya dalam model yang menggunakan Amazon Bedrock Guide di Panduan Pengguna Amazon Bedrock.

Output yang diejek

Tindakan tidak berinteraksi dengan layanan atau sumber daya eksternal di lingkungan pratinjau. Bidang keluaran Mocked digunakan untuk menyediakan ekspresi JSON yang mensimulasikan perilaku konektor di lingkungan pratinjau untuk tujuan pengujian. Cuplikan ini disimpan di `results` peta tindakan, sama seperti respons konektor untuk aplikasi yang dipublikasikan di lingkungan langsung.

Dengan bidang ini, Anda dapat menguji berbagai skenario dan dampaknya terhadap tindakan lain dalam otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus tepi, atau jalur yang tidak menyenangkan tanpa berkomunikasi dengan layanan eksternal melalui konektor.

Amazon Bedrock: Memanggil model

Menggunakan InvokeModel operasi Amazon Bedrock untuk menjalankan inferensi menggunakan parameter prompt dan inferensi yang disediakan di badan permintaan. Anda menggunakan inferensi model untuk menghasilkan teks, gambar, dan penyematan.

Properti

Konektor

Konektor yang akan digunakan untuk operasi yang dijalankan oleh tindakan ini. Agar berhasil menggunakan tindakan ini, konektor harus dikonfigurasi dengan Amazon Bedrock Runtime sebagai layanan. Konektor yang dikonfigurasi harus diatur dengan kredensional yang tepat untuk menjalankan operasi, dan informasi konfigurasi lainnya, seperti AWS wilayah yang berisi sumber daya apa pun yang direferensikan dalam operasi.

Konfigurasi

Isi permintaan yang akan digunakan dalam `InvokeModel` perintah.

> **Note**
>
> Untuk informasi selengkapnya tentang `Amazon Bedrock InvokeModel` operasi, termasuk perintah contoh, lihat InvokeModeldi Referensi Amazon Bedrock API.

Output yang diejek

Tindakan tidak berinteraksi dengan layanan atau sumber daya eksternal di lingkungan pratinjau. Bidang keluaran Mocked digunakan untuk menyediakan ekspresi JSON yang mensimulasikan perilaku konektor di lingkungan pratinjau untuk tujuan pengujian. Cuplikan ini disimpan di `results` peta tindakan, sama seperti respons konektor untuk aplikasi yang dipublikasikan di lingkungan langsung.

Dengan bidang ini, Anda dapat menguji berbagai skenario dan dampaknya terhadap tindakan lain dalam otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus

tepi, atau jalur yang tidak menyenangkan tanpa berkomunikasi dengan layanan eksternal melalui konektor.

JavaScript

Menjalankan JavaScript fungsi kustom untuk mengembalikan nilai tertentu.

> **Important**
>
> App Studio tidak mendukung penggunaan JavaScript pustaka pihak ketiga atau kustom.

Properti

Kode sumber

Cuplikan JavaScript kode yang akan dijalankan oleh tindakan.

> **Tip**
>
> Anda dapat menggunakan AI untuk membantu menghasilkan JavaScript untuk Anda dengan melakukan langkah-langkah berikut:
>
> 1. Pilih ikon perluas untuk membuka JavaScript editor yang diperluas.
> 2. (Opsional): Aktifkan sakelar Ubah kode untuk memodifikasi yang ada. JavaScript Jika tidak, AI akan menggantikan yang ada JavaScript.
> 3. Di Generate JavaScript, jelaskan apa yang ingin Anda lakukan JavaScript, misalnya:**`Add two numbers`**.
> 4. Pilih ikon kirim untuk menghasilkan ikon Anda JavaScript.

Memanggil otomatisasi

Menjalankan otomatisasi tertentu.

Properti

Memohon Otomasi

Otomatisasi yang akan dijalankan oleh tindakan.

Kirim email

Menggunakan `Amazon SES SendEmail` operasi untuk mengirim email.

Properti

Konektor

Konektor yang akan digunakan untuk operasi yang dijalankan oleh tindakan ini. Konektor yang dikonfigurasi harus diatur dengan kredensional yang tepat untuk menjalankan operasi, dan informasi konfigurasi lainnya, seperti AWS wilayah yang berisi sumber daya apa pun yang direferensikan dalam operasi.

Konfigurasi

Isi permintaan yang akan digunakan dalam `SendEmail` perintah. Opsinya adalah sebagai berikut:

> **Note**
>
> Untuk informasi selengkapnya tentang `Amazon SES SendEmail` operasi, lihat SendEmaildi Referensi API Amazon Simple Email Service.

Output yang diejek

Tindakan tidak berinteraksi dengan layanan atau sumber daya eksternal di lingkungan pratinjau. Bidang keluaran Mocked digunakan untuk menyediakan ekspresi JSON yang mensimulasikan perilaku konektor di lingkungan pratinjau untuk tujuan pengujian. Cuplikan ini disimpan di `results` peta tindakan, sama seperti respons konektor untuk aplikasi yang dipublikasikan di lingkungan langsung.

Dengan bidang ini, Anda dapat menguji berbagai skenario dan dampaknya terhadap tindakan lain dalam otomatisasi seperti mensimulasikan nilai hasil yang berbeda, skenario kesalahan, kasus tepi, atau jalur yang tidak menyenangkan tanpa berkomunikasi dengan layanan eksternal melalui konektor.

Mengonfigurasi model data aplikasi Anda dengan entitas

Entitas adalah tabel data di App Studio. Entitas berinteraksi langsung dengan tabel di sumber data. Entitas mencakup bidang untuk mendeskripsikan data di dalamnya, kueri untuk mencari dan mengembalikan data, dan pemetaan untuk menghubungkan bidang entitas ke kolom sumber data.

Topik

Praktik terbaik saat merancang model data

Gunakan praktik terbaik berikut untuk membuat model data relasional yang kuat, dapat diskalakan, dan aman AWS untuk digunakan dalam aplikasi App Studio Anda yang memenuhi persyaratan aplikasi Anda dan memastikan keandalan dan kinerja jangka panjang infrastruktur data Anda.

- Pilih layanan AWS data yang tepat: Tergantung pada kebutuhan Anda, pilih layanan AWS data yang sesuai. Misalnya, untuk aplikasi Online Transaction Processing (OLTP), Anda dapat mempertimbangkan database (DB) seperti Amazon Aurora yang merupakan layanan database cloud-native, relasional, dan dikelola sepenuhnya yang mendukung berbagai mesin database seperti MySQL dan PostgreSQL. Untuk daftar lengkap versi Aurora yang didukung oleh App Studio, lihat. Connect ke Amazon Aurora Di sisi lain, untuk kasus penggunaan Online Analytical Processing (OLAP), pertimbangkan untuk menggunakan Amazon Redshift, yang merupakan gudang data cloud yang memungkinkan Anda menjalankan kueri kompleks terhadap kumpulan data yang sangat besar. Kueri ini seringkali membutuhkan waktu (beberapa detik) untuk diselesaikan, membuat Amazon Redshift kurang cocok untuk aplikasi OLTP yang memerlukan akses data latensi rendah.
- Desain untuk skalabilitas: Rencanakan model data Anda dengan mempertimbangkan pertumbuhan dan skalabilitas di masa depan. Pertimbangkan faktor-faktor seperti volume data yang diharapkan, pola akses, dan persyaratan kinerja saat memilih layanan data yang sesuai dan jenis dan konfigurasi instans database (seperti kapasitas yang disediakan).
  - Untuk informasi selengkapnya tentang penskalaan dengan Aurora tanpa server, lihat Kinerja dan penskalaan untuk Aurora Tanpa Server V2.
- Menormalkan data Anda: Ikuti prinsip normalisasi database untuk meminimalkan redundansi data dan meningkatkan integritas data. Ini termasuk membuat tabel yang sesuai, mendefinisikan kunci primer dan asing, dan membangun hubungan antar entitas. Di App Studio, saat menanyakan data dari satu entitas, Anda dapat mengambil data terkait dari entitas lain dengan menentukan `join` klausa pada kueri.

- Menerapkan pengindeksan yang tepat: Identifikasi kueri dan pola akses yang paling penting, dan buat indeks yang sesuai untuk mengoptimalkan kinerja.
- Manfaatkan fitur layanan AWS data: Manfaatkan fitur yang ditawarkan oleh layanan AWS data yang Anda pilih, seperti pencadangan otomatis, penerapan multi-AZ, dan pembaruan perangkat lunak otomatis.
- Amankan data Anda: Menerapkan langkah-langkah keamanan yang kuat, seperti kebijakan IAM (AWS Identity and Access Management), pembuatan pengguna database dengan izin terbatas ke tabel dan skema, dan menerapkan enkripsi saat istirahat dan dalam perjalanan.
- Memantau dan mengoptimalkan kinerja: Terus memantau kinerja database Anda dan membuat penyesuaian sesuai kebutuhan, seperti penskalaan sumber daya, mengoptimalkan kueri, atau menyetel konfigurasi database.
- Mengotomatiskan manajemen database: Memanfaatkan AWS layanan seperti Aurora Autoscaling, Performance Insights for Aurora, dan AWS Database Migration Service untuk mengotomatiskan tugas manajemen database dan mengurangi overhead operasional.
- Menerapkan strategi pemulihan dan pencadangan bencana: Pastikan Anda memiliki rencana pencadangan dan pemulihan yang terdefinisi dengan baik, memanfaatkan fitur seperti Pencadangan Otomatis Aurora, point-in-time pemulihan, dan konfigurasi replika lintas wilayah.
- Ikuti praktik dan dokumentasi AWS terbaik: Ikuti up-to-date praktik, panduan, dan dokumentasi AWS terbaik terbaru untuk layanan data pilihan Anda guna memastikan bahwa model dan implementasi data Anda selaras dengan AWS rekomendasi.

Untuk panduan lebih rinci dari setiap layanan AWS data, lihat topik berikut:

- Praktik terbaik dengan Amazon Aurora
- Praktik terbaik dengan Amazon Aurora MySQL
- Penyetelan kinerja kueri Amazon Redshift
- Praktik terbaik untuk kueri dan pemindaian data di Amazon DynamoDB

Membuat entitas di aplikasi App Studio

Ada empat metode untuk membuat entitas di aplikasi App Studio. Daftar berikut berisi setiap metode, manfaatnya, dan tautan ke instruksi untuk menggunakan metode tersebut untuk membuat dan kemudian mengkonfigurasi entitas.

- Membuat entitas dari sumber data yang ada: Secara otomatis membuat entitas dan bidangnya dari tabel sumber data yang ada dan memetakan bidang ke kolom tabel sumber data. Opsi ini lebih disukai jika Anda memiliki sumber data yang sudah ada yang ingin Anda gunakan di aplikasi App Studio.
- Membuat entitas dengan sumber data terkelola App Studio: Buat entitas dan tabel DynamoDB yang dikelola App Studio untuk Anda. Tabel DynamoDB diperbarui secara otomatis saat Anda memperbarui entitas Anda. Dengan opsi ini, Anda tidak perlu membuat, mengelola, atau menghubungkan sumber data pihak ketiga secara manual, atau menetapkan pemetaan dari bidang entitas ke kolom tabel. Semua pemodelan dan konfigurasi data aplikasi Anda dilakukan di App Studio. Opsi ini lebih disukai jika Anda tidak ingin mengelola sumber data Anda sendiri dan tabel DynamoDB dan fungsinya cukup untuk aplikasi Anda.
- Membuat entitas kosong: Buat entitas kosong seluruhnya dari awal. Opsi ini lebih disukai jika Anda tidak memiliki sumber data atau konektor yang dibuat oleh admin, dan Anda ingin mendesain model data aplikasi secara fleksibel tanpa dibatasi oleh sumber data eksternal. Anda dapat menghubungkan entitas ke sumber data setelah pembuatan.
- Membuat entitas dengan AI: Menghasilkan entitas, bidang, tindakan data, dan data sampel berdasarkan nama entitas yang ditentukan. Opsi ini lebih disukai jika Anda memiliki gagasan tentang model data untuk aplikasi Anda, tetapi Anda ingin bantuan menerjemahkannya ke dalam entitas.

Membuat entitas dari sumber data yang ada

Gunakan tabel dari sumber data untuk secara otomatis membuat entitas dan bidangnya, dan petakan bidang entitas ke kolom tabel. Opsi ini lebih disukai jika Anda memiliki sumber data yang sudah ada yang ingin Anda gunakan di aplikasi App Studio.

1. Jika perlu, navigasikan ke aplikasi Anda.
2. Pilih tab Data di bagian atas kanvas.
3. Jika tidak ada entitas di aplikasi Anda, pilih + Buat entitas. Jika tidak, di menu Entitas sisi kiri, pilih + Tambah.
4. Pilih Gunakan tabel dari sumber data yang ada.
5. Di Connector, pilih konektor yang berisi tabel yang ingin Anda gunakan untuk membuat entitas Anda.
6. Dalam Tabel, pilih tabel yang ingin Anda gunakan untuk membuat entitas Anda.
7. Pilih kotak centang Buat tindakan data untuk membuat tindakan data.

8. Pilih Buat entitas. Entitas Anda sekarang dibuat, dan Anda dapat melihatnya di panel Entitas sebelah kiri.
9. Konfigurasikan entitas baru Anda dengan mengikuti prosedur diMengonfigurasi atau mengedit entitas di aplikasi App Studio. Perhatikan bahwa karena entitas Anda dibuat dengan sumber data yang ada, beberapa properti atau sumber daya telah dibuat, seperti bidang, sumber data yang terhubung, dan pemetaan bidang. Selain itu, entitas Anda akan berisi tindakan data jika Anda memilih kotak centang Buat tindakan data selama pembuatan.

Membuat entitas dengan sumber data terkelola App Studio

Buat entitas terkelola dan tabel DynamoDB terkait yang dikelola oleh App Studio. Meskipun tabel DynamoDB ada di akun AWS terkait, saat perubahan dilakukan pada entitas di aplikasi App Studio, tabel DynamoDB akan diperbarui secara otomatis. Dengan opsi ini, Anda tidak perlu membuat, mengelola, atau menghubungkan sumber data pihak ketiga secara manual, atau menetapkan pemetaan dari bidang entitas ke kolom tabel. Opsi ini lebih disukai jika Anda tidak ingin mengelola sumber data Anda sendiri dan tabel DynamoDB dan fungsinya cukup untuk aplikasi Anda. Untuk informasi selengkapnya tentang entitas terkelola, lihatEntitas data terkelola di AWS App Studio.

Anda dapat menggunakan entitas terkelola yang sama di beberapa aplikasi. Untuk petunjuk, silakan lihat Membuat entitas dari sumber data yang ada.

1. Jika perlu, navigasikan ke aplikasi Anda.
2. Pilih tab Data di bagian atas kanvas.
3. Jika tidak ada entitas di aplikasi Anda, pilih + Buat entitas. Jika tidak, di menu Entitas sisi kiri, pilih + Tambah.
4. Pilih Buat entitas terkelola App Studio.
5. Dalam nama Entitas, berikan nama untuk entitas Anda.
6. Dalam kunci Primer, berikan nama untuk kunci utama entitas Anda. Kunci utama adalah pengidentifikasi unik entitas dan tidak dapat diubah setelah entitas dibuat.
7. Di tipe data kunci primer, pilih tipe data kunci primer entitas Anda. Tipe data tidak dapat diubah setelah entitas dibuat.
8. Pilih Buat entitas. Entitas Anda sekarang dibuat, dan Anda dapat melihatnya di panel Entitas sebelah kiri.
9. Konfigurasikan entitas baru Anda dengan mengikuti prosedur diMengonfigurasi atau mengedit entitas di aplikasi App Studio. Perhatikan bahwa karena entitas Anda dibuat dengan data

terkelola, beberapa properti atau sumber daya telah dibuat, seperti bidang kunci utama, dan sumber data yang terhubung.

Membuat entitas kosong

Buat entitas kosong seluruhnya dari awal. Opsi ini lebih disukai jika Anda tidak memiliki sumber data atau konektor yang dibuat oleh admin. Membuat entitas kosong menawarkan fleksibilitas, karena Anda dapat mendesain entitas dalam aplikasi App Studio tanpa dibatasi oleh sumber data eksternal. Setelah mendesain model data aplikasi, dan mengonfigurasi entitas sesuai dengan itu, Anda masih dapat menghubungkannya ke sumber data eksternal nanti.

1. Jika perlu, navigasikan ke aplikasi Anda.
2. Pilih tab Data di bagian atas kanvas.
3. Jika tidak ada entitas di aplikasi Anda, pilih + Buat entitas. Jika tidak, di menu Entitas sisi kiri, pilih + Tambah.
4. Pilih Buat entitas.
5. Pilih Buat entitas. Entitas Anda sekarang dibuat, dan Anda dapat melihatnya di panel Entitas sebelah kiri.
6. Konfigurasikan entitas baru Anda dengan mengikuti prosedur diMengonfigurasi atau mengedit entitas di aplikasi App Studio.

Membuat entitas dengan AI

Hasilkan entitas, bidang, tindakan data, dan data sampel berdasarkan nama entitas yang ditentukan. Opsi ini lebih disukai jika Anda memiliki gagasan tentang model data untuk aplikasi Anda, tetapi Anda ingin bantuan menerjemahkannya ke dalam entitas.

1. Jika perlu, navigasikan ke aplikasi Anda.
2. Pilih tab Data di bagian atas kanvas.
3. Jika tidak ada entitas di aplikasi Anda, pilih + Buat entitas. Jika tidak, di menu Entitas sisi kiri, pilih + Tambah.
4. Pilih Buat entitas dengan AI.
5. Dalam nama Entitas, berikan nama untuk entitas Anda. Nama ini digunakan untuk menghasilkan bidang, tindakan data, dan data sampel entitas Anda.
6. Pilih kotak centang Buat tindakan data untuk membuat tindakan data.

7. Pilih Menghasilkan entitas. Entitas Anda sekarang dibuat, dan Anda dapat melihatnya di panel Entitas sebelah kiri.
8. Konfigurasikan entitas baru Anda dengan mengikuti prosedur diMengonfigurasi atau mengedit entitas di aplikasi App Studio. Perhatikan bahwa karena entitas Anda dibuat dengan AI, entitas Anda sudah berisi bidang yang dihasilkan. Selain itu, entitas Anda akan berisi tindakan data jika Anda memilih kotak centang Buat tindakan data selama pembuatan.

Mengonfigurasi atau mengedit entitas di aplikasi App Studio

Gunakan topik berikut untuk mengonfigurasi entitas dalam aplikasi App Studio.

Topik

Mengedit nama entitas

1. Jika perlu, navigasikan ke entitas yang ingin Anda edit.
2. Di tab Konfigurasi, di nama Entitas, perbarui nama entitas dan pilih di luar kotak teks untuk menyimpan perubahan Anda.

Menambahkan, mengedit, atau menghapus bidang entitas

> **Tip**
>
> Anda dapat menekan CTRL+Z untuk membatalkan perubahan terbaru pada entitas Anda.

1. Jika perlu, navigasikan ke entitas yang ingin Anda edit.
2. Di tab Konfigurasi, di Bidang, Anda melihat tabel bidang entitas Anda. Bidang entitas memiliki kolom berikut:

- Nama tampilan: Nama tampilan mirip dengan header tabel atau bidang formulir dan dapat dilihat oleh pengguna aplikasi. Ini dapat berisi spasi dan karakter khusus tetapi harus unik dalam suatu entitas.
- Nama sistem: Nama sistem adalah pengidentifikasi unik yang digunakan dalam kode untuk referensi bidang. Saat memetakan ke kolom dalam tabel Amazon Redshift, itu harus cocok dengan nama kolom tabel Amazon Redshift.
- Tipe data: Jenis data yang akan disimpan dalam bidang ini, seperti`Integer`,`Boolean`, atau`String`.

3. Untuk menambahkan bidang:

   a. Untuk menggunakan AI untuk menghasilkan bidang berdasarkan nama entitas dan sumber data yang terhubung, pilih Hasilkan lebih banyak bidang.

   b. Untuk menambahkan satu bidang, pilih + Tambah bidang.

4. Untuk mengedit bidang:

   a. Untuk mengedit nama tampilan, masukkan nilai yang diinginkan di kotak teks Nama tampilan. Jika nama sistem bidang belum diedit, itu akan diperbarui ke nilai baru dari nama tampilan.

   b. Untuk mengedit nama sistem, masukkan nilai yang diinginkan di kotak teks Nama sistem.

   c. Untuk mengedit tipe data, pilih menu tarik-turun tipe data dan pilih jenis yang diinginkan dari daftar.

   d. Untuk mengedit properti bidang, pilih ikon roda gigi bidang. Daftar berikut merinci properti bidang:

      - Wajib: Aktifkan opsi ini jika bidang diperlukan oleh sumber data Anda.
      - Kunci utama: Aktifkan opsi ini jika bidang dipetakan ke kunci utama di sumber data Anda.
      - Unik: Aktifkan opsi ini jika nilai bidang ini harus unik.
      - Gunakan sumber data default: Aktifkan opsi ini jika nilai bidang disediakan oleh sumber data, seperti menggunakan kenaikan otomatis, atau stempel waktu acara.
      - Opsi tipe data: Bidang tipe data tertentu dapat dikonfigurasi dengan opsi tipe data seperti nilai minimum atau maksimum.

5. Untuk menghapus bidang, pilih ikon sampah dari bidang yang ingin Anda hapus.

Membuat, mengedit, atau menghapus tindakan data

Tindakan data digunakan dalam aplikasi untuk menjalankan tindakan pada data entitas, seperti mengambil semua catatan, atau mengambil catatan berdasarkan ID. Tindakan data dapat digunakan untuk menemukan dan mengembalikan data yang cocok dengan kondisi tertentu untuk dilihat dalam komponen seperti tabel atau tampilan detail.

Daftar Isi

Membuat tindakan data

> **Tip**
>
> Anda dapat menekan CTRL+Z untuk membatalkan perubahan terbaru pada entitas Anda.

1. Jika perlu, navigasikan ke entitas yang ingin Anda buat tindakan data.
2. Pilih tab Tindakan data.
3. Ada dua metode untuk membuat tindakan data:
   - (Disarankan) Untuk menggunakan AI untuk menghasilkan tindakan data untuk Anda, berdasarkan nama entitas, bidang, dan sumber data yang terhubung, pilih Hasilkan tindakan data. Tindakan berikut akan dihasilkan:
     1. `getAll`: Mengambil semua catatan dari entitas. Tindakan ini berguna ketika Anda perlu menampilkan daftar catatan atau melakukan operasi pada beberapa catatan sekaligus.
     2. `getByID`: Mengambil satu catatan dari entitas berdasarkan pengenal uniknya (ID atau kunci utama). Tindakan ini berguna ketika Anda perlu menampilkan atau melakukan operasi pada catatan tertentu.
   - Untuk menambahkan tindakan data tunggal, pilih + Tambahkan tindakan data.
4. Untuk melihat atau mengonfigurasi tindakan data baru, lihat bagian berikut,Mengedit atau mengonfigurasi tindakan data.

Mengedit atau mengonfigurasi tindakan data

1. Jika perlu, navigasikan ke entitas yang ingin Anda buat tindakan data.
2. Pilih tab Tindakan data.
3. Di Bidang konfigurasikan bidang yang akan dikembalikan oleh kueri. Secara default, semua bidang yang dikonfigurasi dalam entitas dipilih.

   Anda juga dapat menambahkan Gabungan ke tindakan data dengan melakukan langkah-langkah berikut:

   1. Pilih + Tambahkan Gabung untuk membuka kotak dialog.
   2. Di Entitas terkait, pilih entitas yang ingin Anda gabungkan dengan entitas saat ini.
   3. Di Alias, secara opsional masukkan nama alias sementara untuk entitas terkait.
   4. Di Jenis Gabung, pilih jenis gabungan yang diinginkan.
   5. Tentukan klausa gabungan dengan memilih bidang dari setiap entitas.
   6. Pilih Tambah untuk membuat bergabung.

   Setelah dibuat, gabungan akan ditampilkan di bagian Gabung, membuat bidang tambahan tersedia di dropdown Fields to Return. Anda dapat menambahkan beberapa gabungan, termasuk gabungan berantai di seluruh entitas. Anda juga dapat memfilter dan mengurutkan berdasarkan bidang dari entitas yang bergabung.

   Untuk menghapus gabungan, pilih ikon sampah di sebelahnya. Ini akan menghapus bidang apa pun dari gabungan itu dan mematahkan gabungan atau batasan dependen apa pun menggunakan bidang tersebut.
4. Dalam Kondisi, tambahkan, edit, atau hapus aturan yang memfilter output kueri. Anda dapat mengatur aturan ke dalam grup, dan Anda dapat menggabungkan beberapa aturan dengan `AND` atau `OR` pernyataan.
5. Di Sorting, konfigurasikan bagaimana hasil kueri diurutkan dengan memilih atribut dan memilih urutan naik atau turun. Anda dapat menghapus konfigurasi penyortiran dengan memilih ikon sampah di sebelah aturan penyortiran.
6. Di hasil Transform, Anda dapat memasukkan kustom JavaScript untuk memodifikasi atau memformat hasil sebelum ditampilkan atau dikirim ke otomatisasi.
7. Di Pratinjau keluaran, lihat tabel pratinjau output kueri berdasarkan bidang yang dikonfigurasi, filter, pengurutan, dan JavaScript.

Menghapus tindakan data

Gunakan prosedur berikut untuk menghapus tindakan data dari entitas App Studio.

1. Jika perlu, navigasikan ke entitas yang ingin Anda hapus tindakan datanya.
2. Pilih tab Tindakan data.
3. Untuk setiap tindakan data yang ingin Anda hapus, pilih menu tarik-turun di sebelah Edit dan pilih Hapus.
4. Pilih Konfirmasi di kotak dialog.

Menambahkan atau menghapus data sampel

Anda dapat menambahkan data sampel ke entitas dalam aplikasi App Studio. Karena aplikasi tidak berkomunikasi dengan layanan eksternal hingga dipublikasikan, data sampel dapat digunakan untuk menguji aplikasi dan entitas Anda di lingkungan pratinjau.

1. Jika perlu, navigasikan ke entitas yang ingin Anda edit.
2. Pilih tab Data sampel.
3. Untuk menghasilkan data sampel, pilih Hasilkan lebih banyak data sampel.
4. Untuk menghapus data sampel, pilih kotak centang data yang ingin Anda hapus, lalu tekan tombol Hapus atau Backspace. Pilih Simpan untuk menyimpan perubahan.

Menambahkan atau mengedit sumber data yang terhubung dan bidang peta

> **Tip**
>
> Anda dapat menekan CTRL+Z untuk membatalkan perubahan terbaru pada entitas Anda.

1. Jika perlu, navigasikan ke entitas yang ingin Anda edit.
2. Pilih tab Connection untuk melihat atau mengelola koneksi antara entitas dan tabel sumber data tempat data disimpan saat aplikasi Anda dipublikasikan. Setelah tabel sumber data terhubung, Anda dapat memetakan bidang entitas ke kolom tabel.
3. Di Connector, pilih konektor yang berisi koneksi ke tabel sumber data yang diinginkan. Untuk informasi lebih lanjut tentang konektor, lihatConnect App Studio ke layanan lain dengan konektor.

4. Dalam Tabel, pilih tabel yang ingin Anda gunakan sebagai sumber data untuk entitas.
5. Tabel menunjukkan bidang entitas, dan kolom sumber data yang dipetakan. Pilih Peta otomatis untuk secara otomatis memetakan bidang entitas Anda dengan kolom sumber data Anda. Anda juga dapat memetakan bidang secara manual dalam tabel dengan memilih kolom sumber data di dropdown untuk setiap bidang entitas.

Menghapus entitas

Gunakan prosedur berikut untuk menghapus entitas dari aplikasi App Studio.

> **Note**
>
> Menghapus entitas dari aplikasi App Studio tidak akan menghapus tabel sumber data yang terhubung, termasuk tabel DynamoDB terkait entitas terkelola. Tabel sumber data akan tetap berada di AWS akun terkait dan perlu dihapus dari layanan yang sesuai jika diinginkan.

Untuk menghapus entitas

1. Jika perlu, navigasikan ke aplikasi Anda.
2. Pilih tab Data.
3. Di menu Entitas sebelah kiri, pilih menu elips di sebelah entitas yang ingin Anda hapus dan pilih Hapus.
4. Tinjau informasi di kotak dialog, masukkan **`confirm`** dan pilih Hapus untuk menghapus entitas.

Entitas data terkelola di AWS App Studio

Biasanya, Anda mengonfigurasi entitas di App Studio dengan sambungan ke tabel database eksternal, dan Anda harus membuat dan memetakan setiap bidang entitas dengan kolom di tabel database yang terhubung. Saat Anda membuat perubahan pada model data, tabel database eksternal dan entitas harus diperbarui, dan bidang yang diubah harus dipetakan ulang. Meskipun metode ini fleksibel dan memungkinkan penggunaan berbagai jenis sumber data, dibutuhkan lebih banyak perencanaan awal dan pemeliharaan berkelanjutan.

Entitas terkelola adalah jenis entitas yang App Studio mengelola seluruh proses penyimpanan dan konfigurasi data untuk Anda. Saat Anda membuat entitas terkelola, tabel DynamoDB yang

sesuai dibuat di akun terkait. AWS Ini memastikan manajemen data yang aman dan transparan di dalamnya AWS. Dengan entitas terkelola, Anda mengonfigurasi skema entitas di App Studio, dan tabel DynamoDB terkait juga diperbarui secara otomatis.

Menggunakan entitas terkelola dalam beberapa aplikasi

Setelah Anda membuat entitas terkelola di aplikasi App Studio, entitas tersebut dapat digunakan di aplikasi App Studio lainnya. Ini berguna untuk mengonfigurasi penyimpanan data untuk aplikasi dengan model dan skema data yang identik dengan menyediakan satu sumber daya dasar untuk dipelihara.

Saat menggunakan entitas terkelola dalam beberapa aplikasi, semua pembaruan skema ke tabel DynamoDB yang sesuai harus dibuat menggunakan aplikasi asli tempat entitas terkelola dibuat. Setiap perubahan skema yang dibuat untuk entitas dalam aplikasi lain tidak akan memperbarui tabel DynamoDB yang sesuai.

Batasan entitas yang dikelola

Pembatasan pembaruan kunci primer: Anda tidak dapat mengubah nama atau jenis kunci utama entitas setelah dibuat, karena ini adalah perubahan destruktif di DynamoDB, dan akan mengakibatkan hilangnya data yang ada.

Mengganti nama kolom: Ketika Anda mengganti nama kolom di DynamoDB, Anda benar-benar membuat kolom baru sementara kolom asli tetap dengan data asli. Data asli tidak secara otomatis disalin ke kolom baru atau dihapus dari kolom asli. Anda dapat mengganti nama bidang entitas terkelola, yang dikenal sebagai nama sistem, tetapi Anda akan kehilangan akses ke kolom asli dan datanya. Tidak ada batasan dengan mengganti nama nama tampilan.

Mengubah tipe data: Meskipun DynamoDB memungkinkan fleksibilitas untuk memodifikasi tipe data kolom setelah pembuatan tabel, perubahan tersebut dapat sangat memengaruhi data yang ada serta logika dan akurasi kueri. Perubahan tipe data memerlukan transformasi semua data yang ada agar sesuai dengan format baru, yang kompleks untuk tabel besar dan aktif. Selain itu, tindakan data dapat mengembalikan hasil yang tidak terduga hingga migrasi data selesai. Anda dapat mengganti tipe data bidang, tetapi data yang ada tidak akan dimigrasikan ke tipe data baru.

Sorting Kolom: DynamoDB memungkinkan pengambilan data yang diurutkan melalui Sort Keys. Sort Keys harus didefinisikan sebagai bagian dari Composite Primary Keys bersama dengan Partition Key. Batasan termasuk Kunci Sortir wajib, pengurutan terbatas dalam satu partisi, dan tidak ada penyortiran global di seluruh partisi. Pemodelan data yang cermat dari Sort Keys diperlukan untuk menghindari partisi panas. Kami tidak akan mendukung Sorting for Preview milestone.

Bergabung: Gabungan tidak didukung di DynamoDB. Tabel didenormalisasi dengan desain untuk menghindari operasi gabungan yang mahal. Untuk memodelkan one-to-many hubungan, tabel anak berisi atribut yang mereferensikan kunci utama tabel induk. Kueri data multi-tabel melibatkan mencari item dari tabel induk untuk mengambil detail. Kami tidak akan mendukung Gabungan asli untuk entitas Terkelola sebagai bagian dari tonggak Pratinjau. Sebagai solusinya, kami akan memperkenalkan langkah otomatisasi yang dapat melakukan penggabungan data dari 2 entitas. Ini akan sangat mirip dengan pencarian satu tingkat. Kami tidak akan mendukung Sorting for Preview milestone.

Env Stage: Kami akan mengizinkan penerbitan untuk menguji tetapi menggunakan toko terkelola yang sama di kedua lingkungan

Parameter halaman dan otomatisasi

Parameter adalah fitur canggih di AWS App Studio yang digunakan untuk meneruskan nilai dinamis antara berbagai komponen, halaman, dan otomatisasi dalam aplikasi Anda. Dengan menggunakan parameter, Anda dapat membuat pengalaman yang fleksibel dan sadar konteks, membuat aplikasi Anda lebih responsif dan dipersonalisasi. Artikel ini mencakup dua jenis parameter: parameter halaman dan parameter otomatisasi.

Topik

Parameter halaman

Parameter halaman adalah cara untuk mengirim informasi antar halaman dan sering digunakan saat menavigasi dari satu halaman ke halaman lain dalam aplikasi App Studio untuk mempertahankan konteks atau meneruskan data. Parameter halaman biasanya terdiri dari nama dan nilai.

Kasus penggunaan parameter halaman

Parameter halaman digunakan untuk meneruskan data antara halaman dan komponen yang berbeda dalam aplikasi App Studio Anda. Mereka sangat membantu untuk kasus penggunaan berikut:

1. Pencarian dan pemfilteran: Saat pengguna menelusuri di beranda aplikasi Anda, istilah penelusuran dapat diteruskan sebagai parameter ke halaman hasil, sehingga hanya menampilkan

item yang difilter yang relevan. Misalnya, jika pengguna mencari*noise-cancelling headphones*, parameter dengan nilai *noise-cancelling headphones* dapat diteruskan ke halaman daftar produk.

2. Melihat detail item: Jika pengguna mengklik daftar, seperti produk, pengenal unik item tersebut dapat diteruskan sebagai parameter ke halaman detail. Ini memungkinkan halaman detail untuk menampilkan semua informasi tentang item tertentu. Misalnya, ketika pengguna mengklik produk headphone, ID unik produk diteruskan sebagai parameter ke halaman detail produk.
3. Melewati konteks pengguna dalam navigasi halaman: Saat pengguna menavigasi antar halaman, parameter dapat meneruskan konteks penting, seperti lokasi pengguna, kategori produk pilihan, konten keranjang belanja, dan pengaturan lainnya. Misalnya, saat pengguna menelusuri kategori produk yang berbeda di aplikasi Anda, lokasi dan kategori pilihan mereka dipertahankan sebagai parameter, memberikan pengalaman yang dipersonalisasi dan konsisten.
4. Tautan dalam: Gunakan parameter halaman untuk membagikan atau menandai tautan ke halaman tertentu di dalam aplikasi.
5. Tindakan data: Anda dapat membuat tindakan data yang menerima nilai parameter untuk memfilter dan menanyakan sumber data Anda berdasarkan parameter yang diteruskan. Misalnya, pada halaman daftar produk, Anda dapat membuat tindakan data yang menerima `category` parameter untuk mengambil produk yang relevan.

Pertimbangan keamanan parameter halaman

Meskipun parameter halaman menyediakan cara yang ampuh untuk meneruskan data antar halaman, Anda harus menggunakannya dengan hati-hati, karena mereka berpotensi mengekspos informasi sensitif jika tidak digunakan dengan benar. Berikut adalah pertimbangan keamanan penting yang perlu diingat:

1. Hindari mengekspos data sensitif di URLs
   a. Risiko: URLs, termasuk parameter tindakan data, sering terlihat di log server, riwayat browser, dan tempat lain. Karena itu, penting untuk menghindari mengekspos data sensitif, seperti kredensi pengguna, informasi identitas pribadi (PII), atau data rahasia lainnya, dalam nilai parameter halaman.
   b. Mitigasi: Pertimbangkan untuk menggunakan pengidentifikasi yang dapat dipetakan dengan aman ke data sensitif. Misalnya, alih-alih meneruskan nama pengguna atau alamat email sebagai parameter, Anda dapat meneruskan pengenal unik acak yang dapat digunakan untuk mengambil nama atau email pengguna.

Parameter otomatisasi

Parameter otomatisasi adalah fitur canggih di App Studio yang dapat digunakan untuk membuat otomatisasi yang fleksibel dan dapat digunakan kembali dengan meneruskan nilai dinamis dari berbagai sumber, seperti UI, otomatisasi lain, atau tindakan data. Mereka bertindak sebagai placeholder yang diganti dengan nilai aktual ketika otomatisasi dijalankan, memungkinkan Anda untuk menggunakan otomatisasi yang sama dengan input yang berbeda setiap kali.

Dalam otomatisasi, parameter memiliki nama unik, dan Anda dapat mereferensikan nilai parameter menggunakan variabel params diikuti dengan nama parameter, misalnya,`{{params.customerId}}`.

Artikel ini memberikan pemahaman mendalam tentang parameter otomatisasi, termasuk konsep dasar, penggunaan, dan praktik terbaiknya.

Manfaat parameter otomatisasi

Parameter otomatisasi memberikan beberapa manfaat, termasuk daftar berikut:

1. Reusability: Dengan menggunakan parameter, Anda dapat membuat otomatisasi yang dapat digunakan kembali yang dapat disesuaikan dengan nilai input yang berbeda, memungkinkan Anda untuk menggunakan kembali logika otomatisasi yang sama dengan input yang berbeda.
2. Fleksibilitas: Alih-alih nilai hard-coding ke dalam otomatisasi, Anda dapat menentukan parameter dan memberikan nilai yang berbeda bila diperlukan, membuat otomatisasi Anda lebih dinamis dan mudah beradaptasi.
3. Pemisahan kekhawatiran: Parameter membantu memisahkan logika otomatisasi dari nilai-nilai spesifik yang digunakan, mempromosikan organisasi kode dan pemeliharaan.
4. Validasi: Setiap parameter memiliki tipe data, seperti string, nomor, atau boolean, yang divalidasi saat runtime. Ini memastikan bahwa permintaan dengan tipe data yang salah ditolak tanpa perlu kode validasi kustom.
5. Parameter opsional dan wajib: Anda dapat menetapkan parameter otomatisasi sebagai opsional atau wajib. Parameter yang diperlukan harus disediakan saat menjalankan otomatisasi, sedangkan parameter opsional dapat memiliki nilai default atau dihilangkan. Fleksibilitas ini memungkinkan Anda membuat otomatisasi yang lebih serbaguna yang dapat menangani skenario berbeda berdasarkan parameter yang disediakan.

Skenario dan kasus penggunaan

Skenario: Mengambil detail produk

Bayangkan Anda memiliki otomatisasi yang mengambil detail produk dari database berdasarkan ID produk. Otomatisasi ini dapat memiliki parameter yang disebut`productId`.

`productId`Parameter bertindak sebagai placeholder yang dapat Anda isi dengan nilai ID produk aktual saat menjalankan otomatisasi. Alih-alih hard-coding ID produk tertentu ke dalam otomatisasi, Anda dapat menentukan `productId` parameter dan meneruskan nilai ID produk yang berbeda setiap kali Anda menjalankan otomatisasi.

Anda dapat memanggil otomatisasi ini dari sumber data komponen, meneruskan ID produk yang dipilih sebagai `productId` parameter menggunakan sintaks braket kurawal ganda:. `{{ui.productsTable.selectedRow.id}}` Dengan cara ini, ketika pengguna memilih produk dari tabel (`ui.productsTable`), otomatisasi akan mengambil detail untuk produk yang dipilih dengan meneruskan id dari baris yang dipilih sebagai parameter. `productId`

Atau, Anda dapat menjalankan otomatisasi ini dari otomatisasi lain yang mengulang daftar produk dan mengambil detail untuk setiap produk dengan meneruskan id produk sebagai parameter. `productId` Dalam skenario ini, nilai `productId` parameter akan diberikan secara dinamis dari `{{product.id}}` ekspresi di setiap iterasi loop.

Dengan menggunakan `productId` parameter dan sintaks braket keriting ganda, Anda dapat membuat otomatisasi ini lebih fleksibel dan dapat digunakan kembali. Alih-alih membuat otomatisasi terpisah untuk setiap produk, Anda dapat memiliki otomatisasi tunggal yang dapat mengambil detail untuk produk apa pun hanya dengan memberikan ID produk yang sesuai sebagai nilai parameter dari sumber yang berbeda, seperti komponen UI atau otomatisasi lainnya.

Skenario: Menangani parameter opsional dengan nilai fallback

Mari pertimbangkan skenario di mana Anda memiliki entitas “Tugas” dengan kolom “Pemilik” yang diperlukan, tetapi Anda ingin bidang ini menjadi opsional dalam otomatisasi dan memberikan nilai fallback jika pemilik tidak dipilih.

1. Buat otomatisasi dengan parameter bernama `0wner` yang memetakan ke `0wner` bidang `Task` entitas.
2. Karena `0wner` bidang diperlukan dalam entitas, `0wner` parameter akan disinkronkan dengan pengaturan yang diperlukan.

3. Untuk membuat `Owner` parameter opsional dalam otomatisasi, matikan `required` pengaturan untuk parameter ini.
4. Dalam logika otomatisasi Anda, Anda dapat menggunakan ekspresi seperti`{{params.Owner || currentUser.userId}}`. Ekspresi ini memeriksa apakah `Owner` parameter disediakan. Jika tidak disediakan, itu akan mundur ke ID pengguna saat ini sebagai pemilik.
5. Dengan cara ini, jika pengguna tidak memilih pemilik dalam bentuk atau komponen, otomatisasi akan secara otomatis menetapkan pengguna saat ini sebagai pemilik untuk tugas tersebut.

Dengan mengaktifkan `required` pengaturan untuk `Owner` parameter dan menggunakan ekspresi fallback, Anda dapat memisahkannya dari persyaratan bidang entitas, menjadikannya opsional dalam otomatisasi, dan memberikan nilai default saat parameter tidak disediakan.

Mendefinisikan jenis parameter otomatisasi

Dengan menggunakan tipe parameter untuk menentukan tipe data dan menetapkan persyaratan, Anda dapat mengontrol input untuk otomatisasi Anda. Ini membantu memastikan otomatisasi Anda berjalan dengan andal dengan input yang diharapkan.

Menyinkronkan tipe dari entitas

Menyinkronkan tipe dan persyaratan parameter secara dinamis dari definisi bidang entitas menyederhanakan otomatisasi bangunan yang berinteraksi dengan data entitas, memastikan bahwa parameter selalu mencerminkan jenis dan persyaratan bidang entitas terbaru.

Prosedur berikut merinci langkah-langkah umum untuk menyinkronkan tipe parameter dari entitas:

1. Buat entitas dengan bidang yang diketik (misalnya Boolean, Number, dll.) Dan tandai bidang sesuai kebutuhan.
2. Buat otomatisasi baru.
3. Tambahkan parameter ke otomatisasi, dan saat memilih Jenis, pilih bidang entitas yang ingin Anda sinkronkan. Tipe data dan pengaturan yang diperlukan akan secara otomatis menyinkronkan dari bidang entitas yang dipetakan
4. Jika diperlukan, Anda dapat mengganti pengaturan “wajib” dengan mengaktifkan/menonaktifkannya untuk setiap parameter. Ini berarti status yang diperlukan tidak akan disinkronkan dengan bidang entitas, tetapi jika tidak, status tersebut akan tetap disinkronkan.

Mendefinisikan tipe secara manual

Anda juga dapat menentukan jenis parameter secara manual tanpa menyinkronkan dari entitas

Dengan mendefinisikan jenis parameter kustom, Anda dapat membuat otomatisasi yang menerima jenis input tertentu dan menangani parameter opsional atau wajib sesuai kebutuhan, tanpa bergantung pada pemetaan bidang entitas.

1. Buat entitas dengan bidang yang diketik (misalnya Boolean, Number, dll.) Dan tandai bidang sesuai kebutuhan.
2. Buat otomatisasi baru.
3. Tambahkan parameter ke otomatisasi, dan saat memilih Jenis, pilih jenis yang diinginkan.

Mengkonfigurasi nilai dinamis untuk diteruskan ke parameter otomatisasi

Setelah Anda menentukan parameter untuk otomatisasi, Anda dapat meneruskan nilai kepada mereka saat menjalankan otomatisasi. Anda dapat meneruskan nilai parameter dengan dua cara:

1. Pemicu komponen: Jika Anda menjalankan otomatisasi dari pemicu komponen, seperti klik tombol, Anda dapat menggunakan JavaScript ekspresi untuk meneruskan nilai dari konteks komponen. Misalnya, jika Anda memiliki kolom input teks bernama`emailInput`, Anda dapat meneruskan nilainya ke parameter email dengan ekspresi berikut:`ui.emailInput.value`.
2. Otomatisasi lain: Jika Anda menjalankan otomatisasi dari otomatisasi lain, Anda dapat menggunakan JavaScript ekspresi untuk meneruskan nilai dari konteks otomatisasi. Misalnya, Anda dapat melewatkan nilai parameter lain atau hasil dari langkah tindakan sebelumnya.

Jenis keamanan

Dengan mendefinisikan parameter dengan tipe data tertentu, seperti String, Number, atau Boolean, Anda dapat memastikan bahwa nilai yang diteruskan ke otomatisasi Anda adalah tipe yang diharapkan.

> **Note**
>
> Di App Studio, tanggal adalah tanggal string ISO, dan tanggal tersebut akan divalidasi juga.

Keamanan tipe ini membantu mencegah ketidakcocokan tipe, yang dapat menyebabkan kesalahan atau perilaku tak terduga dalam logika otomatisasi Anda. Misalnya, jika Anda mendefinisikan parameter sebagai a`Number`, Anda dapat yakin bahwa nilai apa pun yang diteruskan ke parameter itu akan menjadi angka, dan Anda tidak perlu melakukan pemeriksaan atau konversi tipe tambahan dalam otomatisasi Anda.

Validasi

Anda dapat menambahkan aturan validasi ke parameter Anda, memastikan bahwa nilai yang diteruskan ke otomatisasi Anda memenuhi kriteria tertentu.

Meskipun App Studio tidak menyediakan pengaturan validasi bawaan untuk parameter, Anda dapat menerapkan validasi kustom dengan menambahkan JavaScript tindakan ke otomatisasi Anda yang menimbulkan kesalahan jika batasan tertentu dilanggar.

Untuk bidang entitas, subset aturan validasi, seperti tindakan minimum/maximum values, are supported. However, those are not validated at the automation level, only at the data layer, when running Create/Update/Delete Rekam.

Praktik terbaik untuk parameter otomatisasi

Untuk memastikan bahwa parameter otomatisasi Anda dirancang dengan baik, dapat dipelihara, dan mudah digunakan, ikuti praktik terbaik berikut:

1. Gunakan nama parameter deskriptif: Pilih nama parameter yang dengan jelas menggambarkan tujuan atau konteks parameter.
2. Berikan deskripsi parameter: Manfaatkan bidang Deskripsi saat menentukan parameter untuk menjelaskan tujuan, kendala, dan harapannya. Deskripsi ini akan muncul di JSDoc komentar saat mereferensikan parameter, serta di antarmuka pengguna mana pun di mana pengguna perlu memberikan nilai untuk parameter saat menjalankan otomatisasi.
3. Gunakan tipe data yang sesuai: Pertimbangkan dengan cermat tipe data setiap parameter berdasarkan nilai input yang diharapkan, misalnya: String, Number, Boolean, Object.
4. Validasi nilai parameter: Terapkan pemeriksaan validasi yang sesuai dalam otomatisasi Anda untuk memastikan bahwa nilai parameter memenuhi persyaratan tertentu sebelum melanjutkan dengan tindakan lebih lanjut.
5. Gunakan nilai fallback atau default: Meskipun App Studio saat ini tidak mendukung pengaturan nilai default untuk parameter, Anda dapat menerapkan nilai fallback atau default saat menggunakan parameter dalam logika otomatisasi Anda. Misalnya, Anda dapat menggunakan

ekspresi seperti `{{ params.param1 || "default value" }}` untuk memberikan nilai default jika `param1` parameter tidak disediakan atau memiliki nilai palsu.

6. Pertahankan konsistensi parameter: Jika Anda memiliki beberapa otomatisasi yang memerlukan parameter serupa, cobalah untuk menjaga konsistensi dalam nama parameter dan tipe data di seluruh otomatisasi tersebut.
7. Penggunaan parameter dokumen: Pertahankan dokumentasi yang jelas untuk otomatisasi Anda, termasuk deskripsi setiap parameter, tujuannya, nilai yang diharapkan, dan contoh atau kasus tepi yang relevan.
8. Tinjau dan refactor sesering mungkin: Tinjau otomatisasi dan parameternya secara berkala, refactoring atau konsolidasi parameter sesuai kebutuhan untuk meningkatkan kejelasan, pemeliharaan, dan penggunaan kembali.
9. Batasi jumlah parameter: Meskipun parameter memberikan fleksibilitas, terlalu banyak parameter dapat membuat otomatisasi menjadi rumit dan sulit digunakan. Bertujuan untuk mencapai keseimbangan antara fleksibilitas dan kesederhanaan dengan membatasi jumlah parameter hanya untuk apa yang diperlukan.
10. Pertimbangkan pengelompokan parameter: Jika Anda menemukan diri Anda mendefinisikan beberapa parameter terkait, pertimbangkan untuk mengelompokkannya menjadi satu parameter. *`Object`*
11. Kekhawatiran terpisah: Hindari menggunakan satu parameter untuk berbagai tujuan atau menggabungkan nilai yang tidak terkait menjadi satu parameter. Setiap parameter harus mewakili perhatian atau potongan data yang berbeda.
12. Gunakan alias parameter: Jika Anda memiliki parameter dengan nama panjang atau kompleks, pertimbangkan untuk menggunakan alias atau versi singkatan dalam logika otomatisasi untuk keterbacaan dan pemeliharaan yang lebih baik.

Dengan mengikuti praktik terbaik ini, Anda dapat memastikan bahwa parameter otomatisasi Anda dirancang dengan baik, dapat dipelihara, dan mudah digunakan, yang pada akhirnya meningkatkan kualitas dan efisiensi otomatisasi Anda secara keseluruhan.

Menggunakan JavaScript untuk menulis ekspresi di App Studio

Di AWS App Studio, Anda dapat menggunakan JavaScript ekspresi untuk mengontrol perilaku dan tampilan aplikasi secara dinamis. JavaScript Ekspresi baris tunggal ditulis dalam kurung kurawal ganda`{{ }}`, dan dapat digunakan dalam berbagai konteks seperti otomatisasi, komponen UI, dan

kueri data. Ekspresi ini dievaluasi saat runtime dan dapat digunakan untuk melakukan perhitungan, memanipulasi data, dan mengontrol logika aplikasi.

App Studio menyediakan dukungan native untuk tiga library JavaScript open source: Luxon, UUID, Lodash, serta integrasi SDK untuk mendeteksi kesalahan JavaScript sintaks dan pengecekan tipe dalam konfigurasi aplikasi Anda.

> **Important**
>
> App Studio tidak mendukung penggunaan JavaScript pustaka pihak ketiga atau kustom.

Sintaks dasar

JavaScript ekspresi dapat mencakup variabel, literal, operator, dan panggilan fungsi. Ekspresi biasanya digunakan untuk melakukan perhitungan atau mengevaluasi kondisi.

Lihat contoh berikut:

- `{{ 2 + 3 }}`akan mengevaluasi ke 5.
- `{{ "Hello, " + "World!" }}`akan mengevaluasi untuk “Halo, Dunia!”.
- `{{ Math.max(5, 10) }}`akan mengevaluasi hingga 10.
- `{{ Math.random() * 10 }}`mengembalikan angka acak (dengan desimal) antara [0-10).

Interpolasi

Anda juga dapat menggunakan JavaScript untuk menginterpolasi nilai dinamis dalam teks statis. Ini dicapai dengan melampirkan JavaScript ekspresi dalam kurung kurawal ganda, seperti contoh berikut:

```
Hello {{ currentUser.firstName }}, welcome to App Studio!
```

Dalam contoh ini, `currentUser.firstName` adalah JavaScript ekspresi yang mengambil nama depan pengguna saat ini, yang kemudian secara dinamis dimasukkan ke dalam pesan ucapan.

Rangkaian

Anda dapat menggabungkan string dan variabel menggunakan + operator di JavaScript, seperti pada contoh berikut.

```
{{ currentRow.FirstName + " " + currentRow.LastName }}
```

Ekspresi ini menggabungkan nilai-nilai `currentRow.FirstName` dan `currentRow.LastName` dengan spasi di antaranya, menghasilkan nama lengkap dari baris saat ini.

Tanggal dan waktu

JavaScript menyediakan berbagai fungsi dan objek untuk bekerja dengan tanggal dan waktu. Sebagai contoh:

- `{{ new Date().toLocaleDateString() }}`: Mengembalikan tanggal saat ini dalam format lokal.
- `{{ DateTime.now().toISODate() }}`: Mengembalikan tanggal saat ini dalam YYYY-MM-DD format, untuk digunakan dalam komponen Tanggal.

Blok kode

Selain ekspresi, Anda juga dapat menulis blok JavaScript kode multi-baris. Tidak seperti ekspresi, blok kode tidak memerlukan kurawal kurawal. Sebagai gantinya, Anda dapat menulis JavaScript kode Anda langsung di dalam editor blok kode.

> **Note**
>
> Sementara ekspresi dievaluasi dan nilainya ditampilkan, blok kode dijalankan, dan outputnya (jika ada) ditampilkan.

Variabel dan fungsi global

App Studio menyediakan akses ke variabel dan fungsi global tertentu yang dapat digunakan dalam JavaScript ekspresi dan blok kode Anda. Misalnya, `currentUser` adalah variabel global yang mewakili pengguna yang saat ini masuk, dan Anda dapat mengakses properti seperti `currentUser.role` mengambil peran pengguna.

Merujuk atau memperbarui nilai komponen UI

Anda dapat menggunakan ekspresi dalam komponen dan tindakan otomatisasi untuk mereferensikan dan memperbarui nilai komponen UI. Dengan mereferensikan dan memperbarui nilai komponen secara terprogram, Anda dapat membuat antarmuka pengguna dinamis dan interaktif yang merespons input pengguna dan perubahan data.

Mereferensikan nilai komponen UI

Anda dapat membuat aplikasi interaktif dan berbasis data dengan menerapkan perilaku dinamis dengan mengakses nilai dari komponen UI.

Anda dapat mengakses nilai dan properti komponen UI pada halaman yang sama dengan menggunakan `ui` namespace dalam ekspresi. Dengan mereferensikan nama komponen, Anda dapat mengambil nilainya atau melakukan operasi berdasarkan statusnya.

> **Note**
>
> `ui`Namespace hanya akan menampilkan komponen pada halaman saat ini, karena komponen dicakup ke halaman masing-masing.

Sintaks dasar untuk merujuk ke komponen dalam aplikasi App Studio adalah:`{{ui.`*`componentName`*`}}`.

Daftar berikut berisi contoh penggunaan `ui` namespace untuk mengakses nilai komponen UI:

- `{{ui.`*`textInputName`*`.value}}`: Merupakan nilai komponen input teks bernama*`textInputName`*.
- `{{ui.`*`formName`*`.isValid}}`: Periksa apakah semua bidang dalam formulir bernama *`formName`* valid berdasarkan kriteria validasi yang Anda berikan.
- `{{ui.`*`tableName`*`.currentRow.`*`columnName`*`}}`: Merupakan nilai kolom tertentu di baris saat ini dari komponen tabel bernama*`tableName`*.
- `{{ui.`*`tableName`*`.selectedRowData.`*`fieldName`*`}}`: Merupakan nilai bidang tertentu dari baris yang dipilih dalam komponen tabel bernama*`tableName`*. Anda kemudian dapat menambahkan nama bidang seperti `ID` (`{{ui.`*`tableName`*`.selectedRowData.`*`ID`*`}}`) untuk mereferensikan nilai bidang itu dari baris yang dipilih.

Daftar berikut berisi contoh yang lebih spesifik dari referensi nilai komponen:

- `{{ui.inputText1.value.trim().length > 0}}`: Periksa apakah nilai *inputText1* komponen, setelah memotong spasi putih depan atau belakang, memiliki string yang tidak kosong. Ini dapat berguna untuk memvalidasi input pengguna atau mengaktifkan/menonaktifkan komponen lain berdasarkan nilai bidang teks input.
- `{{ui.multiSelect1.value.join(", ")}}`: Untuk komponen multi-pilih bernama*multiSelect1*, ekspresi ini mengubah array nilai opsi yang dipilih menjadi string yang dipisahkan koma. Ini dapat membantu untuk menampilkan opsi yang dipilih dalam format yang mudah digunakan atau meneruskan pilihan ke komponen atau otomatisasi lain.
- `{{ui.multiSelect1.value.includes("option1")}}`: Ekspresi ini memeriksa *option1* apakah nilai disertakan dalam array opsi yang dipilih untuk *multiSelect1* komponen. Ia mengembalikan true jika *option1* dipilih, dan false sebaliknya. Ini dapat berguna untuk merender komponen secara kondisional atau mengambil tindakan berdasarkan pilihan opsi tertentu.
- `{{ui.s3Upload1.files.length > 0}}`: Untuk komponen unggahan file Amazon S3 bernama*s3Upload1*, ekspresi ini memeriksa apakah ada file yang telah diunggah dengan memeriksa panjang array file. Ini dapat berguna untuk mengaktifkan/menonaktifkan komponen atau tindakan lain berdasarkan apakah file telah diunggah.
- `{{ui.s3Upload1.files.filter(file => file.type === "image/png").length}}`: Ekspresi ini menyaring daftar file yang diunggah dalam *s3Upload1* komponen untuk hanya menyertakan file gambar PNG, dan mengembalikan jumlah file tersebut. Ini dapat membantu untuk memvalidasi atau menampilkan informasi tentang jenis file yang diunggah.

Memperbarui nilai komponen UI

Untuk memperbarui atau memanipulasi nilai komponen, gunakan `RunComponentAction` dalam otomatisasi. Berikut adalah contoh sintaks yang dapat Anda gunakan untuk memperbarui nilai komponen input teks bernama *myInput* menggunakan `RunComponentAction` tindakan:

```
RunComponentAction(ui.myInput, "setValue", "New Value")
```

Dalam contoh ini, `RunComponentAction` langkah memanggil `setValue` tindakan pada *myInput* komponen, meneruskan nilai baru,*New Value*.

Bekerja dengan data tabel

Anda dapat mengakses data tabel dan nilai untuk melakukan operasi. Anda dapat menggunakan ekspresi berikut untuk mengakses data tabel:

- `currentRow`: Digunakan untuk mengakses data tabel dari baris saat ini dalam tabel. Misalnya, menyetel nama tindakan tabel, mengirimkan nilai dari baris ke otomatisasi yang dimulai dari tindakan, atau menggunakan nilai dari kolom yang ada dalam tabel untuk membuat kolom baru.
- `ui.`*`tableName`*`.selectedRow`dan `ui.`*`tableName`*`.selectedRowData` keduanya digunakan untuk mengakses data tabel dari komponen lain pada halaman. Misalnya, mengatur nama tombol di luar tabel berdasarkan baris yang dipilih. Nilai yang dikembalikan adalah sama, tetapi perbedaan antara `selectedRow` dan `selectedRowData` adalah sebagai berikut:
  - `selectedRow`: Namespace ini menyertakan nama yang ditampilkan di header kolom untuk setiap bidang. Anda harus menggunakan `selectedRow` saat mereferensikan nilai dari kolom yang terlihat di tabel. Misalnya, jika Anda memiliki kolom kustom atau dihitung dalam tabel Anda yang tidak ada sebagai bidang dalam entitas.
  - `selectedRowData`: Namespace ini mencakup bidang dalam entitas yang digunakan sebagai sumber untuk tabel. Anda harus menggunakan `selectedRowData` untuk mereferensikan nilai dari entitas yang tidak terlihat dalam tabel, tetapi berguna untuk komponen atau otomatisasi lain di aplikasi Anda.

Daftar berikut berisi contoh mengakses data tabel dalam ekspresi:

- `{{ui.`*`tableName`*`.selectedRow.`*`columnNameWithNoSpace`*`}}`: Mengembalikan nilai *`columnNameWithNoSpace`* kolom dari baris yang dipilih dalam tabel.
- `{{ui.`*`tableName`*`.selectedRow.['`*`Column Name With Space`*`']}}`: Mengembalikan nilai *`Column Name With Space`* kolom dari baris yang dipilih dalam tabel.
- `{{ui.`*`tableName`*`.selectedRowData.`*`fieldName`*`}}`: Mengembalikan nilai bidang *`fieldName`* entitas dari baris yang dipilih dalam tabel.
- `{{ui.`*`tableName`*`.selectedRows[0].`*`columnMappingName`*`}}`: Referensi nama kolom baris yang dipilih dari komponen atau ekspresi lain pada halaman yang sama.
- `{{currentRow.`*`firstName`*` + ' ' + currentRow.`*`lastNamecolumnMapping`*`}}`: Menggabungkan nilai dari beberapa kolom untuk membuat kolom baru dalam tabel.
- `{{ { "Blocked": "#", "Delayed": "#", "On track": "#" } [currentRow.`*`statuscolumnMapping`*`] + " " +`

`currentRow.statuscolumnMapping}}`: Sesuaikan nilai tampilan bidang dalam tabel berdasarkan nilai status yang disimpan.

- `{{currentRow.colName}}`,`{{currentRow["First Name"]}}`,`{{currentRow}}`, atau`{{ui.tableName.selectedRows[0]}}`: Lewati konteks baris yang direferensikan dalam tindakan baris.

Mengakses otomatisasi

Anda dapat menggunakan otomatisasi untuk menjalankan logika dan operasi sisi server di App Studio. Dalam tindakan otomatisasi, Anda dapat menggunakan ekspresi untuk memproses data, menghasilkan nilai dinamis, dan menggabungkan hasil dari tindakan sebelumnya.

Mengakses parameter otomatisasi

Anda dapat meneruskan nilai dinamis dari komponen UI dan otomatisasi lainnya ke dalam otomatisasi, membuatnya dapat digunakan kembali dan fleksibel. Ini dilakukan dengan menggunakan parameter otomatisasi dengan `params` namespace sebagai berikut:

`{{params.parameterName}}`: Referensi nilai yang diteruskan ke otomatisasi dari komponen UI atau sumber lain. Misalnya, `{{params.ID}}` akan mereferensikan parameter bernama*ID*.

Memanipulasi parameter otomatisasi

Anda dapat menggunakan JavaScript untuk memanipulasi parameter otomatisasi. Lihat contoh berikut:

- `{{params.firstName}} {{params.lastName}}`: Menggabungkan nilai yang diteruskan sebagai parameter.
- `{{params.numberParam1 + params.numberParam2}}`: Tambahkan dua parameter angka.
- `{{params.valueProvided?.length > 0 ? params.valueProvided : 'Default'}}`: Periksa apakah parameter tidak nol atau tidak terdefinisi, dan memiliki panjang bukan nol. Jika benar, gunakan nilai yang disediakan; jika tidak, tetapkan nilai default.
- `{{params.rootCause || "No root cause provided"}}`: Jika `params.rootCause` parameternya salah (null, undefined, atau string kosong), gunakan nilai default yang disediakan.
- `{{Math.min(params.numberOfProducts, 100)}}`: Batasi nilai parameter ke nilai maksimum (dalam hal ini,`100`).

- `{{ DateTime.fromISO(params.startDate).plus({ days: 7 }).toISO() }}`: Jika `params.`*`startDate`* parameternya`"2023-06-15T10:30:00.000Z"`, ekspresi ini akan dievaluasi`"2023-06-22T10:30:00.000Z"`, yang merupakan tanggal satu minggu setelah tanggal mulai.

Mengakses hasil otomatisasi dari tindakan sebelumnya

Otomatisasi memungkinkan aplikasi untuk menjalankan logika sisi server dan operasi, seperti query database, berinteraksi dengan, atau melakukan transformasi data. APIs `results`Namespace menyediakan akses ke output dan data yang dikembalikan oleh tindakan sebelumnya dalam otomatisasi yang sama. Perhatikan poin-poin berikut tentang mengakses hasil otomatisasi:

1. Anda hanya dapat mengakses hasil dari langkah-langkah otomatisasi sebelumnya dalam otomatisasi yang sama.
2. Jika Anda memiliki tindakan bernama *`action1`* dan *`action2`* dalam urutan itu, *`action1`* tidak dapat mereferensikan hasil apa pun, dan hanya *`action2`* dapat mengakses`results.action1`.
3. Ini juga berfungsi dalam tindakan sisi klien. Misalnya, jika Anda memiliki tombol yang memicu otomatisasi menggunakan `InvokeAutomation` tindakan. Anda kemudian dapat memiliki langkah navigasi dengan `Run If` kondisi seperti menavigasi `results.`*`myInvokeAutomation1`*`.fileType === "pdf"` ke halaman dengan penampil PDF jika otomatisasi menunjukkan file tersebut adalah PDF.

Daftar berikut berisi sintaks untuk mengakses hasil otomatisasi dari tindakan sebelumnya menggunakan namespace. `results`

- `{{results.`*`stepName`*`.data}}`: Ambil array data dari langkah otomatisasi bernama*`stepName`*.
- `{{results.`*`stepName`*`.output}}`: Ambil output dari langkah otomatisasi bernama*`stepName`*.

Cara Anda mengakses hasil langkah otomatisasi tergantung pada jenis tindakan dan data yang dikembalikan. Tindakan yang berbeda dapat mengembalikan properti atau struktur data yang berbeda. Berikut adalah beberapa contoh umum:

- Untuk tindakan data, Anda dapat mengakses array data yang dikembalikan menggunakan`results.`*`stepName`*`.data`.
- Untuk tindakan panggilan API, Anda dapat mengakses badan respons menggunakan`results.`*`stepName`*`.body`.

- Untuk tindakan Amazon S3, Anda dapat mengakses konten file menggunakan. `results.stepName.Body.transformToWebStream()`

Lihat dokumentasi untuk jenis tindakan tertentu yang Anda gunakan untuk memahami bentuk data yang mereka kembalikan dan cara mengaksesnya di dalam `results` namespace. Daftar berikut berisi beberapa contoh

- `{{results.getDataStep.data.filter(row => row.status === "pending").length}}`: Dengan asumsi *getDataStep* adalah tindakan `Invoke Data Action` otomatisasi yang mengembalikan array baris data, ekspresi ini menyaring array data untuk memasukkan hanya baris di mana bidang status sama dengan`pending`, dan mengembalikan panjang (hitungan) dari array yang difilter. Hal ini dapat berguna untuk query atau pengolahan data berdasarkan kondisi tertentu.
- `{{params.email.split("@")[0]}}`: Jika `params.email` parameter berisi alamat email, ekspresi ini membagi string pada simbol @ dan mengembalikan bagian sebelum simbol @, secara efektif mengekstrak bagian nama pengguna dari alamat email.
- `{{new Date(params.timestamp * 1000)}}`: Ekspresi ini mengambil parameter timestamp Unix (`params.timestamp`) dan mengubahnya menjadi objek Date. JavaScript Ini mengasumsikan bahwa stempel waktu dalam hitungan detik, sehingga mengalikannya dengan 1000 untuk mengubahnya menjadi milidetik, yang merupakan format yang diharapkan oleh konstruktor. `Date` Ini dapat berguna untuk bekerja dengan nilai tanggal dan waktu dalam otomatisasi.
- `{{results.stepName.Body}}`: Untuk tindakan `Amazon S3 GetObject` otomatisasi bernama*stepName*, ekspresi ini mengambil konten file, yang dapat dikonsumsi oleh komponen UI seperti Image atau PDF Viewer untuk menampilkan file yang diambil. Perhatikan bahwa ekspresi ini perlu dikonfigurasi dalam keluaran Otomasi otomatisasi untuk digunakan dalam komponen.

Ketergantungan data dan pertimbangan waktu

Saat membuat aplikasi kompleks di App Studio, sangat penting untuk memahami dan mengelola dependensi data antara komponen data yang berbeda, seperti formulir, tampilan detail, dan komponen yang didukung otomatisasi. Komponen dan otomatisasi data mungkin tidak menyelesaikan pengambilan atau eksekusi data mereka pada saat yang sama, yang dapat menyebabkan masalah waktu, kesalahan, dan perilaku yang tidak terduga. Dengan mengetahui

potensi masalah waktu dan mengikuti praktik terbaik, Anda dapat menciptakan pengalaman pengguna yang lebih andal dan konsisten di aplikasi App Studio.

Beberapa masalah potensial adalah sebagai berikut:

1. Konflik waktu render: Komponen data dapat dirender dalam urutan yang tidak selaras dengan dependensi datanya, berpotensi menyebabkan inkonsistensi atau kesalahan visual.
2. Waktu pengoperasian otomatisasi: Tugas otomatisasi dapat selesai sebelum komponen dimuat penuh, yang menyebabkan kesalahan eksekusi runtime.
3. Komponen macet: Komponen yang didukung oleh otomatisasi dapat mogok pada respons yang tidak valid atau ketika otomatisasi belum selesai berjalan.

Contoh: Detail pesanan dan informasi pelanggan

Contoh ini menunjukkan bagaimana dependensi antara komponen data dapat menyebabkan masalah waktu dan potensi kesalahan dalam tampilan data.

Pertimbangkan aplikasi dengan dua komponen data berikut pada halaman yang sama:

- Sebuah komponen Detail (`orderDetails`) yang mengambil data pesanan.
- Komponen Detail (`customerDetails`) yang menampilkan detail pelanggan yang terkait dengan pesanan.

Dalam aplikasi ini, ada dua bidang dalam komponen `orderDetails` detail, dikonfigurasi dengan nilai-nilai berikut:

```
// 2 text fields within the orderDetails detail component

// Info from orderDetails Component
{{ui.orderDetails.data[0].name}}

// Info from customerDetails component
{{ui.customerDetails.data[0].name}} // Problematic reference
```

Dalam contoh ini, `orderDetails` komponen mencoba menampilkan nama pelanggan dengan mereferensikan data dari komponen. `customerDetails` Ini bermasalah, karena `orderDetails` komponen dapat dirender sebelum `customerDetails` komponen mengambil datanya. Jika

pengambilan data `customerDetails` komponen tertunda atau gagal, `orderDetails` komponen akan menampilkan informasi yang tidak lengkap atau salah.

Ketergantungan data dan praktik terbaik waktu

Gunakan praktik terbaik berikut untuk mengurangi masalah ketergantungan dan waktu data di aplikasi App Studio Anda:

1. Gunakan rendering bersyarat: Hanya merender komponen atau menampilkan data saat Anda mengonfirmasi bahwa itu tersedia. Gunakan pernyataan bersyarat untuk memeriksa keberadaan data sebelum menampilkannya. Cuplikan berikut menunjukkan contoh pernyataan bersyarat:

   ```
   {{ui.someComponent.data ? ui.someComponent.data.fieldName : "Loading..."}}
   ```

2. Kelola visibilitas komponen anak: Untuk komponen seperti Stepflow, Formulir, atau Detail yang merender turunan sebelum datanya dimuat, setel visibilitas komponen turunan secara manual. Cuplikan berikut menunjukkan contoh pengaturan visibilitas berdasarkan ketersediaan data komponen induk:

   ```
   {{ui.parentComponent.data ? true : false}}
   ```

3. Gunakan kueri gabungan: Jika memungkinkan, gunakan kueri gabungan untuk mengambil data terkait dalam satu kueri. Ini mengurangi jumlah pengambilan data terpisah dan meminimalkan masalah waktu antara komponen data.
4. Terapkan penyerahan kesalahan dalam otomatisasi: Terapkan penanganan kesalahan yang kuat di otomatisasi Anda untuk mengelola skenario dengan anggun di mana data yang diharapkan tidak tersedia atau respons yang tidak valid diterima.
5. Gunakan rantai opsional: Saat mengakses properti bersarang, gunakan rantai opsional untuk mencegah kesalahan jika properti induk tidak terdefinisi. Cuplikan berikut menunjukkan contoh rantai opsional:

   ```
   {{ui.component.data?.[0]?.fieldSystemName}}
   ```

Membangun aplikasi dengan banyak pengguna

Beberapa pengguna dapat bekerja pada satu aplikasi App Studio, namun hanya satu pengguna yang dapat mengedit aplikasi sekaligus. Lihat bagian berikut untuk informasi tentang mengundang

pengguna lain untuk mengedit aplikasi, dan perilaku saat beberapa pengguna mencoba mengedit aplikasi secara bersamaan.

Mengundang pembangun untuk mengedit aplikasi

Gunakan petunjuk berikut untuk mengundang builder lain untuk mengedit aplikasi App Studio.

Untuk mengundang pembangun lain untuk mengedit aplikasi

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda.
2. Pilih Bagikan.
3. Di tab Pengembangan, gunakan kotak teks untuk mencari dan memilih grup atau pengguna individual yang ingin Anda undang untuk mengedit aplikasi.
4. Untuk setiap pengguna atau grup, pilih dropdown dan pilih izin yang akan diberikan kepada pengguna atau grup tersebut.

    - Pemilik bersama: Pemilik bersama memiliki izin yang sama dengan pemilik aplikasi.
    - Hanya mengedit: Pengguna dengan peran Edit saja memiliki izin yang sama dengan pemilik dan pemilik bersama, kecuali yang berikut ini:
        - Mereka tidak dapat mengundang pengguna lain untuk mengedit aplikasi.
        - Mereka tidak dapat mempublikasikan aplikasi ke lingkungan Pengujian atau Produksi.
        - Mereka tidak dapat menambahkan sumber data ke aplikasi.
        - Mereka tidak dapat menghapus atau menduplikasi aplikasi.

Mencoba mengedit aplikasi yang sedang diedit oleh pengguna lain

Satu aplikasi App Studio hanya dapat diedit oleh satu pengguna pada satu waktu. Lihat contoh berikut untuk memahami apa yang terjadi ketika beberapa pengguna mencoba mengedit aplikasi secara bersamaan.

Dalam contoh ini, saat ini `User A` sedang mengedit aplikasi, dan telah membagikannya`User B`. `User B`kemudian mencoba mengedit aplikasi yang sedang diedit oleh`User A`.

Saat `User B` mencoba mengedit aplikasi, kotak dialog akan muncul memberi tahu mereka yang `User A` sedang mengedit aplikasi, dan bahwa melanjutkan akan `User A` keluar dari studio aplikasi, dan semua perubahan akan disimpan. `User B`dapat memilih untuk membatalkan dan membiarkan

`User A` melanjutkan, atau melanjutkan dan masuk ke studio aplikasi untuk mengedit aplikasi. Dalam contoh ini, mereka memilih untuk mengedit aplikasi.

Ketika `User B` memilih untuk mengedit aplikasi, `User A` menerima pemberitahuan yang `User B` telah mulai mengedit aplikasi, dan sesi mereka telah berakhir. Perhatikan `User A` bahwa jika aplikasi terbuka di tab browser yang tidak aktif, mereka mungkin tidak menerima notifikasi. Dalam hal ini, jika mereka mencoba kembali ke aplikasi dan mencoba mengedit, mereka akan menerima pesan kesalahan dan dipandu untuk menyegarkan halaman, yang akan mengembalikannya ke daftar aplikasi.

Melihat atau memperbarui setelan keamanan konten aplikasi Anda

Setiap aplikasi di App Studio memiliki pengaturan keamanan konten yang dapat digunakan untuk membatasi media eksternal atau sumber daya seperti gambar, iFrames, dan PDFs agar tidak dimuat, atau hanya diizinkan dari domain tertentu atau URLs (termasuk bucket Amazon S3). Anda juga dapat menentukan domain tempat aplikasi Anda dapat mengunggah objek ke Amazon S3.

Pengaturan keamanan konten default untuk semua aplikasi adalah memblokir pemuatan semua media dari sumber eksternal, termasuk bucket Amazon S3, dan memblokir pengunggahan objek ke Amazon S3. Oleh karena itu, untuk memuat gambar, iFrame PDFs, atau media serupa, Anda harus mengedit pengaturan untuk memungkinkan sumber media. Selain itu, untuk memungkinkan mengunggah objek ke Amazon S3, Anda harus mengedit pengaturan untuk mengizinkan domain yang dapat diunggah.

> **Note**
>
> Pengaturan keamanan konten digunakan untuk mengonfigurasi header Kebijakan Keamanan Konten (CSP) di aplikasi Anda. CSP adalah standar keamanan yang membantu mengamankan aplikasi Anda dari cross-site scripting (XSS), clickjacking, dan serangan injeksi kode lainnya. Untuk informasi selengkapnya tentang CSP, lihat Kebijakan Keamanan Konten (CSP) di Dokumen Web MDN.

Untuk memperbarui setelan keamanan konten aplikasi

1. Jika perlu, navigasikan ke studio aplikasi aplikasi Anda dengan memilih untuk mengeditnya dari daftar aplikasi.
2. Pilih Pengaturan aplikasi.

3. Pilih tab Pengaturan Keamanan Konten untuk melihat pengaturan berikut:

   - Sumber bingkai: Digunakan untuk mengelola domain tempat aplikasi Anda dapat memuat frame dan iframe (seperti konten interaktif atau PDFs). Setelan ini memengaruhi komponen atau sumber daya aplikasi berikut:
     - Komponen embed iFrame
     - Komponen penampil PDF
   - Sumber gambar: Digunakan untuk mengelola domain tempat aplikasi Anda dapat memuat gambar. Setelan ini memengaruhi komponen atau sumber daya aplikasi berikut:
     - Logo dan spanduk aplikasi
     - Komponen penampil gambar
   - Connect source: Digunakan untuk mengelola domain tempat aplikasi Anda dapat mengunggah objek Amazon S3.

4. Untuk setiap pengaturan, pilih pengaturan yang diinginkan dari dropdown:

   - Blokir semua frames/images/connections: Jangan izinkan media apa pun (gambar, bingkai, PDFs) dimuat, atau objek apa pun untuk diunggah ke Amazon S3.
   - Izinkan semua frames/images/connections: Izinkan semua media (gambar, bingkai, PDFs) dari semua domain dimuat, atau izinkan pengunggahan objek ke Amazon S3 untuk semua domain.
   - Izinkan domain tertentu: Izinkan memuat media dari atau mengunggah media ke domain tertentu. Domain atau URLs ditentukan sebagai daftar ekspresi yang dipisahkan spasi, di mana wildcard (`*`) dapat digunakan untuk subdomain, alamat host, atau nomor port untuk menunjukkan bahwa semua nilai hukum masing-masing valid. Menentukan `http` juga cocok`https`. Daftar berikut berisi contoh entri yang valid:
     - `blob:`: Cocokkan semua gumpalan, yang mencakup data file yang dikembalikan oleh tindakan otomatisasi, seperti `GetObject` mengembalikan item dari bucket Amazon S3, atau gambar yang dihasilkan oleh Amazon Bedrock.

       > **Important**
       >
       > Anda harus menyertakan `blob:` ekspresi yang Anda berikan untuk mengizinkan data file yang dikembalikan oleh tindakan, meskipun ekspresi Anda`*`, Anda harus memperbaruinya `* blob:`

     - `http://*.example.com`: Cocokkan semua upaya untuk memuat dari subdomain mana pun dari. `example.com` Juga cocok dengan `https` sumber daya.

- `https://source1.example.com https//source2.example.com`: Cocokkan semua upaya untuk memuat dari keduanya `https://source1.example.com` dan `https://source2.example.com`
- `https://example.com/subdirectory/`: Cocokkan semua upaya untuk memuat file di bawah direktori subdirektori. Misalnya, `https://example.com/subdirectory/path/to/file.jpeg`. Itu tidak cocok`https://example.com/path/to/file.jpeg`.

5. Pilih Simpan untuk menyimpan perubahan Anda.

Memecahkan masalah dan men-debug App Studio

Topik

Memecahkan masalah penyiapan, izin, dan orientasi App Studio

Topik ini mencakup informasi tentang pemecahan masalah umum saat menyiapkan atau melakukan orientasi ke App Studio, dan mengelola izin.

Pengaturan App Studio gagal saat memilih opsi Buat instance akun untuk saya

Masalah: Menyiapkan App Studio dengan instance Buat akun untuk saya akan gagal jika Anda memiliki instans Pusat Identitas IAM tingkat akun di AWS Wilayah mana pun, karena Pusat Identitas IAM hanya mendukung satu instance.

Solusi: Arahkan ke konsol Pusat Identitas IAM di https://console.aws.amazon.com/singlesignon/untuk memeriksa apakah Anda memiliki instans Pusat Identitas IAM. Periksa setiap AWS Wilayah yang didukung hingga Anda menemukan instans. Anda dapat menggunakan instance itu saat menyiapkan App Studio, atau menghapus instance IAM Identity Center dan coba lagi dengan opsi Create an account instance for me.

> **Warning**
>
> Menghapus instance IAM Identity Center akan memengaruhi kasus penggunaan yang ada. Pastikan instance tidak digunakan sebelum menghapus, atau gunakan instance untuk menyiapkan App Studio.

Tidak dapat mengakses App Studio setelah pengaturan

Masalah: Saat menyiapkan App Studio, Anda mungkin telah menyediakan grup Pusat Identitas IAM yang bukan anggotanya. Anda harus menjadi anggota setidaknya satu grup untuk mengakses App Studio.

Solusi: Arahkan ke konsol Pusat Identitas IAM di https://console.aws.amazon.com/singlesignon/untuk menambahkan diri Anda ke grup yang ditambahkan ke App Studio saat menyiapkan.

Tidak yakin nama pengguna atau kata sandi apa yang akan digunakan saat masuk ke App Studio

Masalah: Anda mungkin tidak yakin bagaimana cara masuk ke App Studio, baik karena Anda belum menyiapkan kredensyal Pusat Identitas IAM Anda, atau Anda lupa nama pengguna atau kata sandi Pusat Identitas IAM Anda.

Solusi: Saat menyiapkan App Studio tanpa instance IAM Identity Center, email dan nama pengguna disediakan untuk setiap pengguna yang akan digunakan untuk membuat pengguna IAM Identity Center. Setiap alamat email yang diberikan dikirim email dengan undangan untuk bergabung dengan IAM Identity Center. Setiap pengguna harus menerima undangan dan membuat kata sandi untuk kredensil pengguna IAM Identity Center mereka. Setiap pengguna kemudian dapat menggunakan nama pengguna dan kata sandi IAM Identity Center untuk masuk ke App Studio.

Jika Anda telah menyiapkan kredensyal dan lupa nama pengguna atau kata sandi Anda, Anda harus meminta administrator Anda untuk menggunakan konsol Pusat Identitas IAM untuk melihat dan memberikan nama pengguna Anda, atau mengatur ulang kata sandi Anda.

Saya mendapatkan kesalahan Sistem saat mengatur App Studio

Masalah: Anda mendapatkan kesalahan berikut saat menyiapkan App Studio:

```
System error. We encountered a problem. Report the issue and the App Studio service
 team will get back to you.
```

Kesalahan ini terjadi ketika layanan mengalami kesalahan yang tidak diketahui.

Solusi: Hubungi tim dukungan dengan bergabung dengan komunitas Slack dengan memilih Bergabunglah dengan kami di Slack di bagian Pelajari navigasi sebelah kiri, atau di spanduk atas saat mengedit aplikasi.

Saya tidak dapat menemukan URL instance App Studio saya

Jika Anda tidak dapat menemukan URL untuk mengakses instance App Studio, hubungi administrator yang menyiapkan App Studio. Administrator dapat melihat URL di konsol App Studio di AWS Management Console.

Saya tidak dapat mengubah grup atau peran di App Studio

Masalah: Anda tidak dapat melihat tautan Peran di navigasi sebelah kiri. Ini karena hanya pengguna dengan peran Admin yang dapat mengubah grup dan peran di App Studio.

Solusi: Jangkau pengguna dengan peran Admin untuk mengubah grup atau peran, atau hubungi administrator Anda untuk ditambahkan ke grup Admin.

Bagaimana cara turun dari App Studio

Anda tidak dapat keluar dari App Studio saat ini. Disarankan untuk menghapus semua sumber daya seperti aplikasi dan konektor, dan mengubah peran grup menjadi Pengguna Aplikasi untuk mencegah akses atau penggunaan. Anda juga harus menghapus sumber daya pihak ketiga yang digunakan secara eksklusif untuk App Studio, seperti peran IAM atau tabel database.

Memecahkan masalah dan men-debug aplikasi

Topik berikut mencakup informasi untuk pemecahan masalah dan debugging aplikasi App Studio.

Topik

Memecahkan masalah asisten pembuat AI dan obrolan

Topik ini berisi panduan pemecahan masalah untuk masalah umum saat menggunakan asisten pembuat AI.

Kesalahan saat membuat aplikasi dengan AI

Saat menggunakan prompt AI untuk membuat aplikasi, kesalahan berikut dapat terjadi:

```
We apologize, but we cannot proceed with your request. The request may contain content
 that violates our policies and guidelines. Please revise your prompt before trying
 again.
```

Masalah: Permintaan diblokir karena konten yang berpotensi berbahaya.

Solusi: Ulangi prompt dan coba lagi.

Aplikasi yang dihasilkan menggunakan AI adalah aplikasi kosong atau komponen yang hilang.

Masalah: Ini dapat disebabkan oleh kesalahan layanan yang tidak terduga.

Solusi: Coba lagi buat aplikasi menggunakan AI, atau buat komponen secara manual di aplikasi yang dihasilkan.

Pemecahan masalah di studio aplikasi

Topik ini berisi panduan pemecahan masalah dan debugging untuk masalah saat membuat aplikasi.

Menggunakan panel debug

Untuk membantu proses debug langsung saat Anda membangun aplikasi, App Studio menyediakan panel debug builder yang dapat dilipat yang mencakup halaman, otomatisasi, dan tab data studio aplikasi. Panel ini menunjukkan kesalahan dan peringatan. Meskipun peringatan berfungsi sebagai saran yang dapat ditindaklanjuti, seperti sumber daya yang belum dikonfigurasi, kesalahan harus diselesaikan agar berhasil melihat pratinjau atau mempublikasikan aplikasi Anda. Setiap kesalahan atau peringatan menyertakan tautan Tampilan yang dapat digunakan untuk menavigasi ke lokasi masalah.

Panel debug secara otomatis memperbarui dengan kesalahan atau peringatan baru saat terjadi, dan kesalahan atau peringatan secara otomatis hilang setelah diselesaikan. Status pesan peringatan dan kesalahan ini tetap ada saat Anda meninggalkan pembuat.

JavaScript sintaks ekspresi dan penanganan tipe data

App Studio menampilkan deteksi JavaScript kesalahan, menyoroti kesalahan dengan menggarisbawahi kode Anda dengan garis merah. Kesalahan kompilasi ini, yang akan mencegah

aplikasi berhasil dibangun, menunjukkan masalah seperti kesalahan ketik, referensi tidak valid, operasi yang tidak valid, dan keluaran yang salah untuk tipe data yang diperlukan. Lihat daftar berikut untuk masalah umum:

1. Kesalahan yang disebabkan oleh penggantian nama resource: Saat JavaScript ekspresi mereferensikan nama resource di App Studio, mengubah nama tersebut akan menyebabkan ekspresi menjadi salah dan menghasilkan error. Anda dapat melihat kesalahan ini di panel debug.
2. Masalah tipe data: Ketidakcocokan tipe data akan menghasilkan kesalahan di aplikasi Anda. Misalnya, jika otomatisasi dikonfigurasi untuk menerima parameter tipe`String`, tetapi komponen dikonfigurasi untuk mengirim nilai tipe`Integer`, kesalahan akan terjadi. Periksa apakah tipe data cocok antara sumber daya yang sesuai, termasuk komponen, otomatisasi, dan entitas data serta tindakan. Anda mungkin perlu mengubah jenis nilai dalam JavaScript ekspresi.

Memecahkan masalah aplikasi pratinjau

Topik ini berisi informasi tentang masalah pemecahan masalah saat mencoba mempratinjau aplikasi.

Pratinjau gagal dimuat dengan kesalahan berikut: **`Your app failed to build and cannot be previewed`**

Masalah: Aplikasi Anda harus berhasil dibangun agar dapat dipratinjau. Kesalahan ini terjadi ketika ada kesalahan kompilasi yang mencegah aplikasi Anda berhasil membangun.

Solusi: Tinjau dan selesaikan kesalahan dengan menggunakan panel debug di studio aplikasi.

Pratinjau membutuhkan waktu lama untuk dimuat

Masalah: Jenis pembaruan aplikasi tertentu memerlukan waktu yang lama untuk dikompilasi dan dibuat.

Solusi: Biarkan tab terbuka, dan tunggu pembaruan dibangun. Anda akan melihat Disimpan di sudut kanan atas studio aplikasi aplikasi, dan pratinjau akan dimuat ulang.

Pratinjau tidak mencerminkan perubahan terbaru

Masalah: Hal ini dapat terjadi ketika sesi pengeditan aplikasi Anda diambil alih oleh pengguna lain, tetapi Anda tidak diberi tahu. Ini dapat menyebabkan aplikasi yang diedit tidak cocok dengan lingkungan pratinjau.

Solusi: Segarkan tab browser studio aplikasi dan ambil alih sesi pengeditan jika diperlukan.

Pemecahan masalah di lingkungan Pengujian

Topik ini berisi informasi tentang pemecahan masalah aplikasi yang dipublikasikan ke lingkungan Pengujian.

> **Note**
>
> Respons HTTP 500 dari otomatisasi atau tindakan data dapat disebabkan oleh crash runtime dalam ekspresi Anda, kegagalan konektor, atau pembatasan dari sumber data yang terhubung ke aplikasi Anda. Gunakan instruksi Menggunakan konsol browser Anda untuk men-debug untuk melihat log debug yang akan menampilkan detail kesalahan yang mendasarinya.

Menggunakan panel debug

Mirip dengan panel debug bangunan yang digunakan saat membuat aplikasi, App Studio menyediakan panel debug yang dapat dilipat di lingkungan Pengujian. Panel ini menampilkan pesan informasi seperti waktu buka halaman, navigasi pengguna, dan acara aplikasi. Ini juga berisi kesalahan dan peringatan. Panel debug secara otomatis memperbarui dengan pesan baru saat peristiwa terjadi.

Menggunakan konsol browser Anda untuk men-debug

Karena tindakan tidak dipanggil saat mempratinjau aplikasi Anda, aplikasi Anda harus dipublikasikan ke lingkungan Pengujian untuk menguji penanganan panggilan dan responsnya. Jika terjadi kesalahan selama eksekusi otomatisasi Anda atau jika Anda ingin memahami mengapa aplikasi berperilaku dengan cara tertentu, Anda dapat menggunakan konsol browser Anda untuk debugging waktu nyata.

Untuk menggunakan konsol browser Anda untuk men-debug aplikasi di lingkungan Pengujian

1. Tambahkan `?debug=true` ke akhir URL dan tekan enter. Perhatikan bahwa jika URL sudah memiliki string kueri (berisi?), sebagai gantinya tambahkan `&debug=true` ke akhir URL.
2. Buka konsol browser Anda untuk memulai debugging dengan menjelajahi input dan output tindakan atau API Anda.

  - Di Chrome: Klik kanan di browser Anda dan pilih Periksa. Untuk informasi selengkapnya tentang debugging dengan Chrome DevTools, lihat DevTools dokumentasi Chrome.
  - Di Firefox: Tekan dan tahan atau klik kanan pada elemen halaman web, lalu pilih Inspect Element. Untuk informasi selengkapnya tentang debugging dengan Firefox DevTools, lihat Dokumen DevTools Pengguna Firefox.

Daftar berikut berisi beberapa masalah umum yang menghasilkan kesalahan:

- Kesalahan runtime
  - Masalah: Jika otomatisasi atau ekspresi dikonfigurasi dengan tidak benar, itu dapat menyebabkan kesalahan saat otomatisasi dijalankan. Kesalahan umum adalah mengganti nama aset, menghasilkan ekspresi yang salah, kesalahan JavaScript kompilasi lainnya, atau upaya untuk menggunakan data atau aset yang ada. `undefined`
  - Solusi: Periksa setiap penggunaan input kode kustom (ekspresi, JavaScript, dan JSON) dan pastikan tidak ada kesalahan kompilasi di editor kode atau panel debug.
- Masalah konektor
  - Masalah: Karena aplikasi App Studio tidak berkomunikasi dengan layanan eksternal dengan konektor hingga dipublikasikan, kesalahan dapat terjadi di lingkungan Pengujian yang tidak terjadi selama pratinjau. Jika tindakan dalam otomatisasi yang menggunakan konektor gagal, itu bisa dari kesalahan konfigurasi dalam tindakan yang mengirimkan permintaan ke konektor, atau dengan konfigurasi konektor itu sendiri.
  - Solusi: Anda harus menggunakan output Mocked untuk menguji otomatisasi di awal lingkungan pratinjau untuk mencegah kesalahan ini. Pastikan konektor Anda dikonfigurasi dengan benar, untuk informasi selengkapnya, lihatKonektor pemecahan masalah. Terakhir, Anda dapat menggunakan CloudWatch untuk meninjau log. Untuk informasi selengkapnya, lihat Debugging dengan log dari aplikasi yang diterbitkan di Amazon CloudWatch Logs. Di log `ConnectorService` namespace, harus ada pesan kesalahan atau metadata yang berasal dari konektor.

Debugging dengan log dari aplikasi yang diterbitkan di Amazon CloudWatch Logs

Amazon CloudWatch Logs memantau AWS sumber daya Anda dan aplikasi yang Anda jalankan AWS secara real time. Anda dapat menggunakan CloudWatch Log untuk mengumpulkan dan

melacak metrik, yang merupakan variabel yang dapat Anda ukur untuk sumber daya dan aplikasi Anda.

Untuk men-debug aplikasi App Studio, CloudWatch Log berguna untuk melacak kesalahan yang terjadi selama eksekusi aplikasi, mengaudit informasi, dan menyediakan konteks tentang tindakan pengguna dan interaksi eksklusif. Log menawarkan data historis, yang dapat Anda gunakan untuk mengaudit penggunaan aplikasi dan pola akses, serta meninjau kesalahan yang dihadapi oleh pengguna.

> **Note**
>
> CloudWatch Log tidak menyediakan jejak real-time dari nilai parameter yang diteruskan dari UI aplikasi.

Gunakan prosedur berikut untuk mengakses log dari aplikasi App Studio Anda di CloudWatch Log.

1. Di studio aplikasi App Studio untuk aplikasi Anda, cari dan catat ID aplikasi Anda dengan melihat di URL. ID aplikasi mungkin terlihat seperti ini:`802a3bd6-ed4d-424c-9f6b-405aa42a62c5`.
2. Buka CloudWatch konsol di https://console.aws.amazon.com/cloudwatch/.
3. Pada panel navigasi, pilih Grup log.
4. Di sini Anda akan menemukan lima grup log per aplikasi. Bergantung pada jenis informasi yang Anda minati, pilih grup dan tulis kueri untuk data yang ingin Anda temukan.

   Daftar berikut berisi grup log dan informasi tentang kapan harus menggunakan masing-masing:

   1. `/aws/appstudio/`*`teamId`*`/`*`appId`*`/TEST/app`: Gunakan untuk men-debug respons otomatisasi, kesalahan komponen, atau JavaScript kode yang terkait dengan versi aplikasi yang saat ini dipublikasikan ke lingkungan Pengujian.
   2. `/aws/appstudio/`*`teamId`*`/`*`appId`*`/TEST/audit`: Gunakan untuk men-debug kesalahan JavaScript kode, seperti visibilitas atau transformasi bersyarat, kegagalan kueri, dan kesalahan pengguna login atau izin yang terkait dengan versi aplikasi Anda yang saat ini dipublikasikan ke lingkungan Pengujian.
   3. `/aws/appstudio/`*`teamId`*`/setup`: Gunakan untuk memantau tindakan pembangun atau admin.

4. `/aws/appstudio/teamId/appId/PRODUCTION/app`: Gunakan untuk men-debug respons otomatisasi, kegagalan kueri, kesalahan komponen, atau JavaScript kode yang terkait dengan versi aplikasi yang saat ini dipublikasikan ke lingkungan Produksi.
5. `/aws/appstudio/teamId/appId/PRODUCTION/audit`: Gunakan untuk men-debug kesalahan JavaScript kode, seperti visibilitas atau transformasi bersyarat, serta kesalahan pengguna login atau izin yang terkait dengan versi aplikasi Anda yang saat ini dipublikasikan ke lingkungan Produksi.

> **Note**
>
> Sebagian besar log yang akan digunakan untuk debugging dikategorikan di bawah namespace. `DebugLogClient`

5. Setelah Anda berada di grup log, Anda dapat memilih aliran log terbaru, atau yang memiliki waktu peristiwa terakhir yang paling dekat dengan waktu yang diinginkan, atau Anda dapat memilih untuk mencari semua aliran log untuk mencari di semua peristiwa di grup log tersebut. Untuk informasi selengkapnya tentang melihat data CloudWatch log di Log, lihat Melihat data log yang dikirim ke CloudWatch Log.

Menggunakan kueri Wawasan CloudWatch Log untuk memfilter dan mengurutkan log

Anda dapat menggunakan Wawasan CloudWatch Log untuk menanyakan beberapa grup log sekaligus. Setelah Anda mengidentifikasi daftar grup log yang berisi informasi sesi, navigasikan ke Wawasan CloudWatch Log dan pilih grup log. Kemudian, lebih mempersempit entri log target dengan menyesuaikan kueri. Berikut adalah beberapa contoh pertanyaan:

Daftar log yang berisi kata kunci: ***error***

```
fields @timestamp, @message
| filter @message like 'error'
| sort @timestamp desc
```

Log debug dari lingkungan Pengujian:

```
fields @timestamp, @message
| filter namespace = "DebugLogClient"
| sort @timestamp desc
```

Keseluruhan kesalahan 504/404/500 dihitung selama interval 5 menit:

```
filter @message like '/api/automation' and (@message like ': 404' or @message like ':
 500' or @message like ': 504')
| fields @timestamp, method, path, statusCode
| stats count(*) as errorCount by bin(5m)
```

Untuk informasi selengkapnya tentang Wawasan CloudWatch Log, Menganalisis data CloudWatch log dengan Wawasan Log di Panduan Pengguna CloudWatch Log Amazon.

Konektor pemecahan masalah

Topik ini berisi panduan pemecahan masalah untuk masalah konektor umum. Anda harus menjadi anggota grup admin untuk melihat atau mengedit konektor.

Periksa apakah peran IAM Anda memiliki kebijakan dan tag kepercayaan khusus yang benar

Saat menyiapkan peran IAM untuk konektor Anda, pastikan bahwa kebijakan kepercayaan kustom dikonfigurasi dengan benar untuk menyediakan akses ke App Studio. Kebijakan kepercayaan khusus ini masih diperlukan jika AWS sumber daya berada di AWS akun yang sama yang digunakan untuk menyiapkan App Studio.

- Pastikan nomor AWS akun di `Principal` bagian ini adalah ID AWS akun akun yang digunakan untuk menyiapkan App Studio. Nomor akun ini tidak selalu merupakan akun tempat sumber daya berada.
- Pastikan `"aws:PrincipalTag/IsAppStudioAccessRole": "true"` ditambahkan dengan benar di `sts:AssumeRole` bagian.

```
{
  "Version": "2012-10-17",
  "Statement": [
  {
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::111122223333:root"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
```

```
        "aws:PrincipalTag/IsAppStudioAccessRole": "true"
      }
    }
  }
  ]
}
```

Juga pastikan bahwa tag dengan kunci dan nilai berikut telah ditambahkan ke peran IAM, untuk informasi selengkapnya tentang menambahkan tag, lihat peran IAM Tag:

> **Note**
>
> Perhatikan bahwa nilai tag adalah`IsAppStudioDataAccessRole`, yang sedikit berbeda dari nilai dalam kebijakan kepercayaan khusus (`IsAppStudioAccessRole`).

- Kunci: `IsAppStudioDataAccessRole`
- Nilai: `true`

Periksa konfigurasi sumber daya dalam produk atau layanan yang terhubung dengan konektor Anda. Beberapa sumber daya, seperti tabel Amazon Redshift, memerlukan konfigurasi tambahan untuk digunakan dengan App Studio.

Periksa konfigurasi konektor Anda. Untuk AWS layanan, buka konektor di App Studio dan pastikan Nama Sumber Daya Amazon (ARN) yang benar disertakan dan AWS Wilayah yang ditentukan adalah yang berisi sumber daya Anda.

Periksa apakah peran IAM Anda memiliki izin yang benar

Untuk memberikan akses App Studio ke AWS sumber daya, Anda harus menetapkan izin yang sesuai untuk peran IAM yang digunakan oleh konektor Anda. Izin yang diperlukan unik untuk layanan, sumber daya, dan tindakan yang akan dilakukan. Misalnya, membaca data dari tabel Amazon Redshift memerlukan izin yang berbeda dengan mengunggah objek ke bucket Amazon S3. Lihat topik yang sesuai Connect ke AWS layanan untuk informasi lebih lanjut.

Memecahkan masalah konektor Amazon Redshift

Bagian ini mencakup panduan pemecahan masalah untuk masalah umum dengan konektor Amazon Redshift. Untuk informasi tentang mengonfigurasi konektor dan sumber daya Amazon Redshift, lihat. Connect ke Amazon Redshift

1. Pastikan `Isolated Session` sakelar disetel ke editor `OFF` Amazon Redshift. Pengaturan ini diperlukan untuk memungkinkan visibilitas perubahan data yang dibuat oleh pengguna lain, seperti aplikasi App Studio.
2. Pastikan izin yang sesuai diberikan pada tabel Amazon Redshift.
3. Dalam konfigurasi konektor, pastikan bahwa jenis komputasi yang sesuai (`Provisioned`atau`Serverless`) dipilih agar sesuai dengan jenis tabel Amazon Redshift.

Memecahkan masalah konektor Aurora

Bagian ini mencakup panduan pemecahan masalah untuk masalah umum dengan konektor Aurora. Untuk informasi tentang mengonfigurasi konektor dan sumber daya Aurora, lihat. Connect ke Amazon Aurora

1. Pastikan bahwa versi Aurora yang sesuai dan didukung dipilih saat membuat tabel.
2. Verifikasi bahwa Amazon RDS Data API diaktifkan, karena ini merupakan persyaratan untuk mengizinkan App Studio melakukan operasi pada tabel Aurora. Untuk informasi selengkapnya, lihat Mengaktifkan Amazon RDS Data API.
3. Verifikasi bahwa AWS Secrets Manager izin diberikan.

Pemecahan masalah konektor DynamoDB

Bagian ini mencakup panduan pemecahan masalah untuk masalah umum dengan konektor DynamoDB. Untuk informasi tentang mengonfigurasi konektor dan sumber daya DynamoDB, lihat. Connect ke Amazon DynamoDB

Jika skema tabel DynamoDB Anda tidak muncul saat membuat konektor, mungkin karena tabel DynamoDB Anda dienkripsi dengan kunci yang dikelola pelanggan (CMK) dan data tabel tidak dapat diakses tanpa izin untuk mendeskripsikan kunci dan mendekripsi tabel. Untuk membuat konektor DynamoDB dengan tabel yang dienkripsi dengan CMK, Anda harus menambahkan `kms:decrypt` dan izin ke peran IAM Anda. `kms:describeKey`

Memecahkan masalah konektor Amazon S3

Bagian ini mencakup panduan pemecahan masalah untuk masalah umum dengan konektor Amazon S3. Untuk informasi tentang mengonfigurasi konektor dan sumber daya Amazon S3, lihat. Connect ke Amazon Simple Storage Service (Amazon S3)

Panduan pemecahan masalah umum termasuk memeriksa hal-hal berikut:

1. Pastikan konektor Amazon S3 dikonfigurasi dengan AWS Wilayah tempat sumber daya Amazon S3 berada.
2. Pastikan peran IAM dikonfigurasi dengan benar.
3. Di bucket Amazon S3, pastikan konfigurasi CORS memberikan izin yang sesuai. Untuk informasi selengkapnya, lihat Langkah 1: Buat dan konfigurasikan sumber daya Amazon S3.

Kesalahan unggahan file Amazon S3: Gagal menghitung URL yang telah ditetapkan sebelumnya

Anda mungkin mengalami kesalahan berikut saat mencoba mengunggah file ke bucket Amazon S3 menggunakan komponen Unggah S3:

```
Error while uploading file to S3: Failed to calculate presigned URL.
```

Kesalahan ini biasanya disebabkan oleh konfigurasi peran IAM yang salah, atau konfigurasi CORS yang salah pada bucket Amazon S3 dan dapat diatasi dengan memperbaiki konfigurasi tersebut dengan informasi di dalamnya. Connect ke Amazon Simple Storage Service (Amazon S3)

Memecahkan masalah penerbitan dan berbagi aplikasi

Topik ini berisi panduan pemecahan masalah untuk masalah umum saat menerbitkan atau berbagi aplikasi App Studio.

Saya tidak melihat peran aplikasi yang baru dibuat di kotak dialog Bagikan

Peran tingkat aplikasi yang baru dibuat hanya akan muncul di kotak dialog Bagikan setelah aplikasi diterbitkan ulang. Publikasikan aplikasi setelah peran baru dibuat untuk menggunakannya.

Saya tidak mendapatkan email ketika publikasi aplikasi saya selesai

Hanya pemilik aplikasi yang menerima email saat aplikasi dipublikasikan.

Pengguna akhir aplikasi saya tidak dapat mengakses aplikasi yang dipublikasikan

Jika pengguna akhir Anda tidak dapat mengakses aplikasi yang dipublikasikan, dan menerima `Forbidden` pesan saat mencoba mengaksesnya, kemungkinan aplikasi yang dipublikasikan tidak

dibagikan dengan pengguna yang mencoba mengaksesnya. Aplikasi yang dipublikasikan harus dibagikan dengan grup untuk memberikan akses ke pengguna dalam grup.

Untuk mempelajari lebih lanjut tentang berbagi aplikasi, lihatBerbagi aplikasi yang diterbitkan.

Keamanan di AWS App Studio

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi 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. Efektivitas keamanan kami diuji dan diverifikasi secara rutin oleh auditor pihak ketiga sebagai bagian dari program kepatuhan AWS. Untuk mempelajari tentang program kepatuhan yang berlaku untuk App Studio, lihat AWS Layanan dalam Lingkup berdasarkan Program Kepatuhan.
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor-faktor lain termasuk sensitivitas data Anda, persyaratan organisasi Anda, serta undang-undang dan peraturan yang berlaku.

Dokumentasi ini akan membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan App Studio. Topik berikut menunjukkan cara mengonfigurasi App Studio untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga akan mempelajari cara menggunakan AWS layanan lain yang dapat membantu Anda memantau dan mengamankan sumber daya App Studio.

Topik

Pertimbangan dan mitigasi keamanan

Pertimbangan keamanan

Ketika berhadapan dengan konektor data, model data, dan aplikasi yang dipublikasikan, beberapa masalah keamanan muncul terkait dengan paparan data, kontrol akses, dan potensi kerentanan. Daftar berikut mencakup masalah keamanan utama.

Konfigurasi peran IAM yang tidak tepat

Konfigurasi peran IAM yang salah untuk konektor data dapat menyebabkan akses dan kebocoran data yang tidak sah. Memberikan akses yang terlalu permisif ke peran IAM konektor data dapat memungkinkan pengguna yang tidak sah untuk mengakses dan memodifikasi data sensitif.

Menggunakan peran IAM untuk melakukan operasi data

Karena pengguna akhir aplikasi App Studio mengambil peran IAM yang disediakan dalam konfigurasi konektor untuk melakukan tindakan, pengguna akhir tersebut mungkin mendapatkan akses ke data yang biasanya tidak dapat mereka akses.

Menghapus konektor data dari aplikasi yang diterbitkan

Ketika konektor data dihapus, kredensyal rahasia terkait tidak secara otomatis dihapus dari aplikasi yang dipublikasikan yang sudah menggunakan konektor itu. Dalam skenario ini, jika aplikasi telah diterbitkan dengan konektor tertentu, dan salah satu konektor tersebut dihapus dari App Studio, aplikasi yang diterbitkan akan terus bekerja menggunakan kredensyal konektor yang disimpan sebelumnya. Penting untuk dicatat bahwa aplikasi yang dipublikasikan akan tetap tidak terpengaruh dan beroperasi meskipun konektor dihapus.

Mengedit konektor data pada aplikasi yang dipublikasikan

Ketika konektor data diedit, perubahan tidak secara otomatis tercermin dalam aplikasi yang dipublikasikan yang menggunakan konektor itu. Jika aplikasi telah diterbitkan dengan konektor tertentu, dan salah satu konektor tersebut dimodifikasi di App Studio, aplikasi yang diterbitkan akan terus menggunakan konfigurasi dan kredensyal konektor yang disimpan sebelumnya. Untuk memasukkan perubahan konektor yang diperbarui, aplikasi harus dipublikasikan ulang. Sampai aplikasi diterbitkan ulang, itu akan tetap salah dan tidak operasional, atau tidak terpengaruh dan operasional tetapi tidak akan mencerminkan modifikasi konektor terbaru.

Rekomendasi mitigasi risiko keamanan

Bagian ini mencantumkan rekomendasi mitigasi untuk menghindari risiko keamanan yang dirinci di bagian pertimbangan keamanan sebelumnya.

1. Konfigurasi peran IAM yang tepat: Pastikan bahwa peran IAM untuk konektor data dikonfigurasi dengan benar dengan prinsip hak istimewa paling sedikit untuk mencegah akses dan kebocoran data yang tidak sah.
2. Akses aplikasi terbatas: Hanya bagikan aplikasi Anda dengan pengguna yang berwenang untuk melihat atau melakukan tindakan pada data aplikasi.
3. Penerbitan aplikasi: Pastikan aplikasi dipublikasikan ulang setiap kali konektor diperbarui atau dihapus.

Perlindungan data di AWS App Studio

Model tanggung jawab AWS bersama model berlaku untuk perlindungan data di AWS App Studio. Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Lihat informasi yang lebih lengkap tentang privasi data dalam Pertanyaan Umum Privasi Data. Lihat informasi tentang perlindungan data di Eropa di pos blog Model Tanggung Jawab Bersama dan GDPR AWS di Blog Keamanan AWS .

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensil dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan sumber daya. AWS Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang penggunaan CloudTrail jejak untuk menangkap AWS aktivitas, lihat Bekerja dengan CloudTrail jejak di AWS CloudTrail Panduan Pengguna.

- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola tingkat lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-3 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi selengkapnya tentang titik akhir FIPS yang tersedia di Standar Pemrosesan Informasi Federal (FIPS) 140-3.

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan AWS App Studio atau lainnya Layanan AWS menggunakan konsol, API AWS CLI, atau AWS SDKs. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log penagihan atau log diagnostik. Saat Anda memberikan URL ke server eksternal, kami sangat menganjurkan supaya Anda tidak menyertakan informasi kredensial di dalam URL untuk memvalidasi permintaan Anda ke server itu.

Enkripsi data

App Studio menyimpan dan mentransfer data dengan aman dengan mengenkripsi data saat istirahat dan dalam perjalanan.

Enkripsi diam

Enkripsi saat istirahat didefinisikan sebagai perlindungan data dari akses tidak sah dengan mengenkripsi data saat disimpan. App Studio menyediakan enkripsi saat istirahat secara default menggunakan AWS KMS kunci, dan Anda tidak perlu melakukan konfigurasi tambahan apa pun untuk enkripsi data saat istirahat.

App Studio menyimpan data berikut untuk aplikasi Anda dengan aman: kode sumber, artefak build, metadata, dan informasi izin.

Saat menggunakan sumber data yang dienkripsi dengan Kunci Terkelola AWS KMS Pelanggan (CMK), sumber daya App Studio terus dienkripsi menggunakan kunci AWS terkelola, sedangkan data dalam sumber data terenkripsi dienkripsi oleh CMK. Untuk informasi selengkapnya tentang menggunakan sumber data terenkripsi di aplikasi App Studio, lihat. Gunakan sumber data terenkripsi dengan CMKs

App Studio menggunakan Amazon CloudFront untuk menayangkan aplikasi Anda kepada pengguna Anda. CloudFront menggunakan SSDs yang dienkripsi untuk titik lokasi tepi kehadiran (POPs), dan volume EBS terenkripsi untuk Regional Edge Cache (). RECs Kode fungsi dan konfigurasi dalam CloudFront Fungsi selalu disimpan dalam format terenkripsi pada lokasi tepi terenkripsi POPs, dan SSDs di lokasi penyimpanan lain yang digunakan oleh. CloudFront

Enkripsi bergerak

Enkripsi dalam transit didefinisikan sebagai perlindungan data dari intersepsi saat data ditransfer antar-endpoint komunikasi. App Studio menyediakan enkripsi untuk data dalam transit secara default. Semua komunikasi antara pelanggan dan App Studio, serta antara App Studio dan dependensi hilirnya dilindungi menggunakan koneksi TLS yang ditandatangani menggunakan proses penandatanganan Signature Version 4. Semua endpoint App Studio menggunakan sertifikat SHA-256 yang dikelola oleh AWS Certificate Manager Private Certificate Authority.

Manajemen kunci

App Studio tidak mendukung pengelolaan kunci enkripsi.

Privasi lalu lintas antar jaringan

Saat membuat instance di App Studio, Anda memilih AWS Wilayah tempat data dan sumber daya akan disimpan untuk instance tersebut. Artefak dan metadata pembuatan aplikasi tidak pernah meninggalkan Wilayah itu. AWS

Namun, perhatikan informasi berikut:

- Karena App Studio menggunakan Amazon CloudFront untuk melayani aplikasi Anda dan menggunakan Lambda @Edge untuk mengelola otentikasi ke aplikasi Anda, kumpulan data otentikasi terbatas, data otorisasi, metadata aplikasi akan diakses dari lokasi CloudFront tepi, yang mungkin berada di Wilayah yang berbeda.
- AWS App Studio mentransfer data di seluruh AWS Wilayah untuk mengaktifkan fitur AI generatif tertentu dalam layanan. Untuk informasi selengkapnya tentang fitur yang diaktifkan oleh transfer data lintas wilayah, jenis data yang bergerak di seluruh Wilayah, dan cara memilih keluar, lihatTransfer data lintas wilayah di AWS App Studio.

AWS App Studio dan AWS Identity and Access Management (IAM)

Di AWS App Studio, Anda mengelola akses dan izin dalam layanan dengan menetapkan grup di Pusat Identitas IAM ke peran yang sesuai di App Studio. Izin anggota grup ditentukan oleh peran yang ditetapkan, dan bukan dengan mengonfigurasi pengguna, peran, atau izin langsung di AWS Identity and Access Management (IAM). Untuk informasi selengkapnya tentang mengelola akses dan izin di App Studio, lihatMengelola akses dan peran di App Studio.

App Studio terintegrasi dengan IAM saat memverifikasi instans untuk tujuan penagihan, dan saat terhubung ke AWS akun untuk membuat dan menggunakan sumber daya di akun tersebut AWS . Untuk informasi tentang menghubungkan App Studio ke AWS layanan lain untuk digunakan dalam aplikasi Anda, lihatConnect ke AWS layanan.

Saat membuat instance di App Studio, Anda harus menghubungkan AWS akun sebagai akun penagihan dan manajemen untuk instans Anda. Untuk mengaktifkan fitur utama, App Studio juga membuat peran layanan IAM untuk menyediakan layanan dengan izin yang diperlukan untuk melaksanakan tugas atas nama Anda.

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 diotorisasi (memiliki izin) untuk menggunakan sumber daya App Studio. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Topik

- AWS kebijakan terkelola untuk AWS App Studio
- Peran terkait layanan untuk App Studio
- Contoh kebijakan berbasis identitas untuk App Studio AWS

Sebelum Anda menggunakan IAM untuk mengelola akses ke App Studio, pelajari fitur IAM yang tersedia untuk digunakan dengan App Studio.

Fitur IAM yang dapat Anda gunakan dengan AWS App Studio

Fitur IAM	Dukungan App Studio
Kebijakan berbasis identitas	Ya
Kebijakan berbasis sumber daya	Tidak
Tindakan kebijakan	Ya
Sumber daya kebijakan	Ya
Kunci kondisi kebijakan	Tidak
ACLs	Tidak
ABAC (tanda dalam kebijakan)	Tidak
Kredensial sementara	Ya
Izin principal	Ya
Peran layanan	Ya
Peran terkait layanan	Ya

Untuk mendapatkan tampilan tingkat tinggi tentang cara kerja App Studio dan AWS layanan lainnya dengan sebagian besar fitur IAM, lihat AWS layanan yang bekerja dengan IAM di Panduan Pengguna IAM.

Kebijakan berbasis identitas untuk App Studio

Mendukung kebijakan berbasis identitas: Ya

Kebijakan berbasis identitas adalah dokumen kebijakan izin JSON yang dapat Anda lampirkan ke sebuah identitas, seperti pengguna IAM, grup pengguna IAM, atau peran IAM. 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 Tentukan izin IAM kustom dengan kebijakan terkelola pelanggan 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 App Studio

Untuk melihat contoh kebijakan berbasis identitas App Studio, lihat. Contoh kebijakan berbasis identitas untuk App Studio AWS

Kebijakan berbasis sumber daya dalam App Studio

Mendukung kebijakan berbasis sumber daya: Tidak

Kebijakan berbasis sumber daya adalah dokumen kebijakan JSON yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan peran IAM dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus menentukan prinsipal dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau. Layanan AWS

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan secara spesifik seluruh akun atau entitas IAM di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya. Menambahkan prinsipal akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika prinsipal dan sumber daya berbeda Akun AWS, administrator IAM di akun tepercaya juga harus memberikan izin entitas utama (pengguna atau peran) untuk mengakses sumber daya. Mereka memberikan izin dengan melampirkan kebijakan berbasis identitas kepada entitas. Namun, jika kebijakan berbasis sumber daya memberikan akses ke principal dalam akun

yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk informasi selengkapnya, lihat Akses sumber daya lintas akun di IAM dalam Panduan Pengguna IAM.

Tindakan kebijakan untuk App Studio

Mendukung tindakan kebijakan: Ya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, di mana utama dapat melakukan tindakan pada sumber daya, 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.

Sertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Untuk melihat daftar tindakan App Studio, lihat Tindakan yang Ditentukan oleh AWS App Studio di Referensi Otorisasi Layanan.

Tindakan kebijakan di App Studio menggunakan awalan berikut sebelum tindakan:

```
appstudio
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan tersebut dengan koma.

```
"Action": [
      "appstudio:action1",
      "appstudio:action2"
         ]
```

Pernyataan berikut mencantumkan semua tindakan di App Studio:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "AWS App Studio permissions",
            "Effect": "Allow",
            "Action": [
                "appstudio:GetAccountStatus", // Required to get the current account's
 App Studio instance status
                "appstudio:GetEnablementJobStatus", // Required to get the status of an
 enablement job of an App Studio instance
                "appstudio:StartEnablementJob", // Required to start the enablement of
 an App Studio instance
                "appstudio:StartRollbackEnablementJob", // Required to disable an
 enabled App Studio instance
                "appstudio:StartTeamDeployment" // Required to start deployment in
 order to update the App Studio instance infrastructure
            ],
            "Resource": "*"
        }
    ]
}
```

Sumber daya kebijakan untuk App Studio

Mendukung sumber daya kebijakan: Ya

Izin App Studio hanya mendukung wildcard (*) dalam `Resource` elemen kebijakan.

Kunci kondisi kebijakan untuk App Studio

Mendukung kunci kondisi kebijakan khusus layanan: Tidak

App Studio tidak mendukung kunci kondisi kebijakan.

ACLs di App Studio

Mendukung ACLs: Tidak

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan JSON.

ABAC dengan App Studio

Mendukung ABAC (tag dalam kebijakan): Tidak

App Studio tidak mendukung kontrol akses berbasis atribut (ABAC).

Menggunakan kredensil sementara dengan App Studio

Mendukung kredensial sementara: Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensyal sementara. Untuk informasi tambahan, termasuk yang Layanan AWS bekerja dengan kredensyal sementara, lihat Layanan AWS yang bekerja dengan IAM di Panduan Pengguna IAM.

Anda menggunakan kredensi sementara jika Anda masuk AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS menggunakan tautan masuk tunggal (SSO) perusahaan Anda, proses tersebut secara otomatis membuat kredensil sementara. Anda juga akan secara otomatis membuat kredensial sementara ketika Anda masuk ke konsol sebagai seorang pengguna lalu beralih peran. Untuk informasi selengkapnya tentang peralihan peran, lihat Beralih dari pengguna ke peran IAM (konsol) dalam Panduan Pengguna IAM.

Anda dapat membuat kredensyal sementara secara manual menggunakan API AWS CLI atau AWS . Anda kemudian dapat menggunakan kredensyal sementara tersebut untuk mengakses. AWS AWS merekomendasikan agar Anda menghasilkan kredensyal sementara secara dinamis alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat Kredensial keamanan sementara di IAM.

Izin utama lintas layanan untuk App Studio

Mendukung sesi akses maju (FAS): Ya

Saat Anda menggunakan pengguna atau peran IAM untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari pemanggilan utama Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. Permintaan FAS hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk menyelesaikannya. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan ketika mengajukan permintaan FAS, lihat Sesi akses maju.

Peran layanan untuk App Studio

Mendukung peran layanan: Ya

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. Untuk informasi selengkapnya, lihat Buat sebuah peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

AWS App Studio menggunakan peran layanan IAM untuk beberapa fitur guna memberikan izin kepada App Studio untuk menjalankan tugas atas nama Anda. Konsol secara otomatis membuat peran layanan untuk fitur yang didukung saat Anda menyiapkan App Studio.

> ⚠ Warning
>
> Mengubah izin untuk peran layanan dapat merusak fungsionalitas App Studio. Edit peran layanan hanya jika App Studio memberikan panduan untuk melakukannya.

Peran terkait layanan untuk App Studio

Mendukung peran terkait layanan: Ya

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan tersebut dapat menjalankan peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Anda Akun AWS dan dimiliki oleh layanan. Administrator IAM dapat melihat, tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Untuk detail tentang pembuatan atau manajemen peran terkait layanan, lihat Layanan AWS yang berfungsi dengan IAM. Cari layanan dalam tabel yang memiliki `Yes` di kolom Peran terkait layanan. Pilih tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

AWS kebijakan terkelola untuk AWS App Studio

Untuk menambahkan izin ke pengguna, grup, dan peran, lebih mudah menggunakan kebijakan AWS terkelola daripada menulis kebijakan sendiri. Dibutuhkan waktu dan keahlian untuk membuat kebijakan yang dikelola pelanggan IAM yang hanya memberi tim Anda izin yang mereka butuhkan. Untuk memulai dengan cepat, Anda dapat menggunakan kebijakan AWS terkelola kami. Kebijakan ini mencakup kasus penggunaan umum dan tersedia di Akun AWS Anda. Untuk informasi selengkapnya tentang kebijakan AWS AWS terkelola, lihat kebijakan terkelola di Panduan Pengguna IAM.

AWS layanan memelihara dan memperbarui kebijakan AWS terkelola. Anda tidak dapat mengubah izin dalam kebijakan AWS terkelola. Layanan terkadang menambahkan izin tambahan ke kebijakan yang dikelola AWS untuk mendukung fitur-fitur baru. Jenis pembaruan ini akan memengaruhi semua identitas (pengguna, grup, dan peran) di mana kebijakan tersebut dilampirkan. Layanan kemungkinan besar akan memperbarui kebijakan yang dikelola AWS saat ada fitur baru yang diluncurkan atau saat ada operasi baru yang tersedia. Layanan tidak menghapus izin dari kebijakan AWS terkelola, sehingga pembaruan kebijakan tidak akan merusak izin yang ada.

Selain itu, AWS mendukung kebijakan terkelola untuk fungsi pekerjaan yang mencakup beberapa layanan. Misalnya, kebijakan ReadOnlyAccess AWS terkelola menyediakan akses hanya-baca ke semua AWS layanan dan sumber daya. Saat layanan meluncurkan fitur baru, AWS tambahkan izin hanya-baca untuk operasi dan sumber daya baru. Untuk melihat daftar dan deskripsi dari kebijakan fungsi tugas, lihat kebijakan yang dikelola AWS untuk fungsi tugas di Panduan Pengguna IAM.

AWS kebijakan terkelola: AppStudioServiceRolePolicy

Anda tidak dapat melampirkan `AppStudioServiceRolePolicy` ke entitas IAM Anda. Kebijakan ini dilampirkan ke peran terkait layanan yang memungkinkan App Studio melakukan tindakan atas nama Anda. Untuk informasi selengkapnya, lihat Peran terkait layanan untuk App Studio.

Kebijakan ini memberikan izin yang memungkinkan peran terkait layanan untuk mengelola sumber daya. AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AppStudioResourcePermissionsForCloudWatch",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:*:*:log-group:/aws/appstudio/*"
            ],
            "Condition": {
```

```
                "StringEquals": {
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
                }
            }
        },
        {
            "Sid": "AppStudioResourcePermissionsForSecretsManager",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:CreateSecret",
                "secretsmanager:DeleteSecret",
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecret",
                "secretsmanager:TagResource"
            ],
            "Resource": "arn:aws:secretsmanager:*:*:secret:appstudio-*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "IsAppStudioSecret"
                    ]
                },
                "StringEquals": {
                    "aws:ResourceAccount": "${aws:PrincipalAccount}",
                    "aws:ResourceTag/IsAppStudioSecret": "true"
                }
            }
        },
        {
            "Sid": "AppStudioResourcePermissionsForSSO",
            "Effect": "Allow",
            "Action": [
                "sso:GetManagedApplicationInstance",
                "sso-directory:DescribeUsers",
                "sso-directory:ListMembersInGroup"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
                }
            }
```

```
        }
    ]
}
```

App Studio memperbarui kebijakan AWS terkelola

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk App Studio sejak layanan ini mulai melacak perubahan ini.

Perubahan	Deskripsi	Tanggal
App Studio mulai melacak perubahan	App Studio mulai melacak perubahan untuk kebijakan AWS terkelolanya.	Juni 28, 2024

Peran terkait layanan untuk App Studio

App Studio menggunakan peran AWS Identity and Access Management terkait layanan (IAM). Peran terkait layanan adalah jenis peran IAM unik yang ditautkan langsung ke App Studio. Peran terkait layanan telah ditentukan sebelumnya oleh App Studio dan menyertakan semua izin yang diperlukan layanan untuk memanggil AWS layanan lain atas nama Anda.

Peran terkait layanan membuat pengaturan App Studio lebih mudah karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. App Studio mendefinisikan izin peran terkait layanan, dan kecuali ditentukan lain, hanya App Studio yang dapat mengambil perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, dan kebijakan izin tersebut tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran tertaut layanan hanya setelah menghapus sumber daya terkait terlebih dahulu. Ini melindungi sumber daya App Studio karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Daftar Isi

Izin peran terkait layanan untuk App Studio

App Studio menggunakan nama peran terkait layanan. `AWSServiceRoleForAppStudio` Ini adalah peran terkait layanan yang diperlukan oleh App Studio untuk mengelola AWS layanan secara terus-menerus, untuk mempertahankan pengalaman pembuatan aplikasi.

Peran `AWSServiceRoleForAppStudio` terkait layanan menggunakan kebijakan kepercayaan berikut, yang hanya mempercayai layanan: `appstudio-service.amazonaws.com`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "appstudio-service.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Untuk izin, peran `AWSServiceRoleForAppStudio` terkait layanan memberikan izin ke layanan berikut:

- Amazon CloudWatch: Untuk mengirim log dan metrik untuk penggunaan App Studio.
- AWS Secrets Manager: Untuk mengelola kredensyal konektor di App Studio, digunakan untuk menghubungkan aplikasi ke layanan lain.
- Pusat Identitas IAM: Untuk akses hanya-baca untuk mengelola akses pengguna.

Secara khusus, izin yang diberikan ditentukan oleh kebijakan `AWSServiceRoleForAppStudio` `AppStudioServiceRolePolicy` terkelola terlampir. Untuk informasi selengkapnya tentang kebijakan terkelola, termasuk izin yang disertakan, lihatAWS kebijakan terkelola: AppStudioServiceRolePolicy.

Membuat peran terkait layanan untuk App Studio

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda membuat instance App Studio, App Studio akan membuat peran terkait layanan untuk Anda.

Jika Anda menghapus peran terkait layanan ini, disarankan untuk membuat instance App Studio agar peran lain dibuat secara otomatis untuk Anda.

Meskipun tidak diperlukan, Anda juga dapat menggunakan konsol IAM atau membuat peran AWS CLI terkait layanan dengan membuat peran terkait layanan dengan nama `appstudio-service.amazonaws.com` layanan, seperti pada cuplikan kebijakan kepercayaan yang ditampilkan sebelumnya. Untuk informasi selengkapnya, lihat Membuat peran tertaut layanan dalam Panduan Pengguna IAM.

Mengedit peran terkait layanan untuk App Studio

App Studio tidak mengizinkan Anda mengedit peran `AWSServiceRoleForAppStudio` terkait layanan. Setelah membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat mengubah deskripsi peran dengan menggunakan IAM. Untuk informasi selengkapnya, lihat Mengedit peran tertaut layanan dalam Panduan Pengguna IAM.

Menghapus peran terkait layanan untuk App Studio

Anda tidak perlu menghapus `AWSServiceRoleForAppStudio` peran. Saat menghapus instance App Studio, App Studio akan membersihkan sumber daya dan menghapus peran yang ditautkan layanan secara otomatis.

Meskipun tidak disarankan, Anda dapat menggunakan konsol IAM atau AWS CLI untuk menghapus peran terkait layanan. Untuk melakukan ini, Anda harus terlebih dahulu membersihkan sumber daya untuk peran terkait layanan Anda dan kemudian Anda dapat menghapusnya.

> **Note**
>
> Jika App Studio menggunakan peran saat Anda mencoba menghapus sumber daya, penghapusan mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi.

Untuk menghapus peran terkait layanan secara manual menggunakan IAM

1. Hapus aplikasi dan konektor dari instans App Studio Anda.
2. Gunakan konsol IAM, CLI IAM, atau API CLI untuk menghapus peran tertaut layanan `AWSServiceRoleForAppStudio`. Untuk informasi selengkapnya, lihat Menghapus peran tertaut layanan dalam Panduan Pengguna IAM.

Contoh kebijakan berbasis identitas untuk App Studio AWS

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau memodifikasi resource App Studio. Mereka juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS API. Untuk memberikan izin kepada pengguna untuk melakukan tindakan di sumber daya yang mereka perlukan, administrator IAM dapat membuat kebijakan IAM. Administrator kemudian dapat menambahkan kebijakan IAM ke peran, dan pengguna dapat mengambil peran.

Untuk mempelajari cara membuat kebijakan berbasis identitas IAM dengan menggunakan contoh dokumen kebijakan JSON ini, lihat Membuat kebijakan IAM (konsol) di Panduan Pengguna IAM.

Untuk detail tentang tindakan dan jenis sumber daya yang ditentukan oleh App Studio, termasuk format ARNs untuk setiap jenis sumber daya, lihat Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS App Studio di Referensi Otorisasi Layanan.

Topik

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus sumber daya App Studio di akun Anda. Tindakan ini membuat Akun AWS Anda dikenai biaya. Ketika Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi ini:

- Mulailah dengan kebijakan AWS terkelola dan beralih ke izin hak istimewa paling sedikit — Untuk mulai memberikan izin kepada pengguna dan beban kerja Anda, gunakan kebijakan AWS terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Anda Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola AWS pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat Kebijakan yang dikelola AWS atau Kebijakan yang dikelola AWS untuk fungsi tugas dalam Panduan Pengguna IAM.

- Menerapkan izin dengan hak akses paling rendah – Ketika Anda menetapkan izin dengan kebijakan IAM, hanya berikan izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk mengajukan izin, lihat Kebijakan dan izin dalam IAM dalam Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan suatu kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Sebagai contoh, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui yang spesifik Layanan AWS, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat Elemen kebijakan JSON IAM: Kondisi dalam Panduan Pengguna IAM.
- Gunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda untuk memastikan izin yang aman dan fungsional – IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan tersebut mematuhi bahasa kebijakan IAM (JSON) dan praktik terbaik IAM. IAM Access Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat Validasi kebijakan dengan IAM Access Analyzer dalam Panduan Pengguna IAM.
- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan pengguna IAM atau pengguna root di Anda, Akun AWS aktifkan MFA untuk keamanan tambahan. Untuk meminta MFA ketika operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat Amankan akses API dengan MFA dalam Panduan Pengguna IAM.

Untuk informasi selengkapnya tentang praktik terbaik dalam IAM, lihat Praktik terbaik keamanan di IAM dalam Panduan Pengguna IAM.

Menggunakan konsol App Studio

Untuk mengakses konsol AWS App Studio, Anda harus memiliki set izin minimum. Izin ini harus memungkinkan Anda untuk membuat daftar dan melihat detail tentang resource App Studio di situs Anda Akun AWS. Jika Anda membuat kebijakan berbasis identitas yang lebih ketat daripada izin minimum yang diperlukan, konsol tidak akan berfungsi sebagaimana mestinya untuk entitas (pengguna atau peran) dengan kebijakan tersebut.

Anda tidak perlu mengizinkan izin konsol minimum untuk pengguna yang melakukan panggilan hanya ke AWS CLI atau AWS API. Sebagai gantinya, izinkan akses hanya ke tindakan yang sesuai dengan operasi API yang coba mereka lakukan.

Untuk memastikan bahwa pengguna dan peran masih dapat menggunakan konsol App Studio, lampirkan juga App Studio *ConsoleAccess* atau kebijakan *ReadOnly* AWS terkelola ke entitas. Untuk informasi selengkapnya, lihat Menambah izin untuk pengguna dalam Panduan Pengguna IAM.

Mengizinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan pengguna IAM melihat kebijakan inline dan terkelola yang dilampirkan ke identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan ini di konsol atau menggunakan API atau secara terprogram. AWS CLI AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

Contoh 1: Izinkan pengguna menyiapkan instans App Studio

Contoh berikut menunjukkan kebijakan berbasis identitas untuk mengizinkan peran menyiapkan instance App Studio.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "appstudio:GetAccountStatus",
            "appstudio:GetEnablementJobStatus",
            "appstudio:StartEnablementJob",
            "appstudio:StartRollbackEnablementJob",
            "appstudio:StartTeamDeployment"
        ],
        "Resource": "*"
    }]
}
```

Contoh 2: Tolak pengguna agar tidak menyiapkan instans App Studio

Contoh berikut menunjukkan kebijakan berbasis identitas untuk menolak peran menyelesaikan instance App Studio.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "Action": [
            "appstudio:*"
        ],
        "Resource": "*"
    }]
}
```

Validasi kepatuhan untuk AWS App Studio

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:

- Kepatuhan dan Tata Kelola Keamanan – Panduan implementasi solusi ini membahas pertimbangan arsitektur serta memberikan langkah-langkah untuk menerapkan fitur keamanan dan kepatuhan.
- Referensi Layanan yang Memenuhi Syarat HIPAA — Daftar layanan yang memenuhi syarat HIPAA. Tidak semua memenuhi Layanan AWS 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 App Studio AWS

Infrastruktur AWS global dibangun di sekitar Wilayah AWS dan Availability Zones. Wilayah AWS menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi, yang terhubung dengan latensi rendah, throughput tinggi, dan jaringan yang 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. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur pusat data tunggal atau multi tradisional.

Untuk informasi selengkapnya tentang Wilayah AWS dan Availability Zone, lihat Infrastruktur AWS Global.

Selain infrastruktur AWS global, AWS App Studio menawarkan beberapa fitur untuk membantu mendukung ketahanan data dan kebutuhan pencadangan Anda.

Keamanan Infrastruktur di AWS App Studio

Sebagai layanan terkelola, AWS App Studio dilindungi oleh prosedur keamanan jaringan AWS global yang dijelaskan dalam whitepaper Amazon Web Services: Tinjauan Proses Keamanan.

Anda menggunakan panggilan API yang AWS dipublikasikan untuk mengakses App Studio melalui jaringan. Klien harus mendukung setidaknya Transport Layer Security (TLS) 1.2, tetapi TLS 1.3 direkomendasikan. Klien juga harus mendukung suite cipher dengan perfect forward secrecy (PFS) seperti Ephemeral Diffie-Hellman (DHE) atau Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

Selain itu, permintaan harus ditandatangani menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan prinsipal IAM. Atau Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk menghasilkan kredensial keamanan sementara untuk menandatangani permintaan.

Analisis konfigurasi dan kerentanan di AWS App Studio

Konfigurasi dan kontrol TI adalah tanggung jawab bersama antara AWS dan Anda, pelanggan kami. Untuk informasi selengkapnya, lihat model tanggung jawab AWS bersama.

Pencegahan "confused deputy" lintas layanan

Masalah "confused deputy" adalah masalah keamanan saat entitas yang tidak memiliki izin untuk melakukan suatu tindakan dapat memaksa entitas yang memilik hak akses lebih tinggi untuk melakukan tindakan tersebut. Pada tahun AWS, peniruan lintas layanan dapat mengakibatkan masalah wakil yang membingungkan. Peniruan identitas lintas layanan dapat terjadi ketika satu layanan (layanan yang dipanggil) memanggil layanan lain (layanan yang dipanggil). Layanan pemanggilan dapat dimanipulasi menggunakan izinnya untuk bertindak pada sumber daya pelanggan lain dengan cara yang seharusnya tidak dilakukannya kecuali bila memiliki izin untuk mengakses. Untuk mencegah hal ini, AWS menyediakan alat yang membantu Anda melindungi data untuk semua layanan dengan principal layanan yang telah diberi akses ke sumber daya di akun Anda.

Sebaiknya gunakan kunci konteks kondisi `aws:SourceAccount`global `aws:SourceArn`dan dalam kebijakan sumber daya untuk membatasi izin yang memberikan layanan lain ke sumber daya. Gunakan `aws:SourceArn` jika Anda ingin hanya satu sumber daya yang akan dikaitkan dengan akses lintas layanan. Gunakan `aws:SourceAccount` jika Anda ingin mengizinkan sumber daya apa pun di akun tersebut dikaitkan dengan penggunaan lintas layanan.

Cara paling efektif untuk melindungi dari masalah "confused deputy" adalah dengan menggunakan kunci konteks kondisi global `aws:SourceArn` dengan ARN lengkap sumber daya. Jika Anda tidak mengetahui ARN lengkap sumber daya atau jika Anda menentukan beberapa sumber daya, gunakan kunci kondisi konteks global `aws:SourceArn` dengan karakter wildcard (*) untuk bagian ARN yang tidak diketahui. Misalnya, `arn:aws:`*`servicename`*`:*:`*`123456789012`*`:*`.

Jika nilai `aws:SourceArn` tidak berisi ID akun, seperti ARN bucket Amazon S3, Anda harus menggunakan kedua kunci konteks kondisi global tersebut untuk membatasi izin.

Nilai `aws:SourceArn` harus ResourceDescription.

Contoh berikut menunjukkan bagaimana Anda dapat menggunakan kunci konteks kondisi `aws:SourceAccount` global `aws:SourceArn` dan untuk mencegah masalah wakil yang membingungkan.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "servicename.amazonaws.com"
    },
    "Action": "servicename:ActionName",
    "Resource": [
      "arn:aws:servicename:::ResourceName/*"
    ],
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:servicename:*:123456789012:*"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

Transfer data lintas wilayah di AWS App Studio

AWS App Studio mentransfer data di seluruh AWS Wilayah untuk mengaktifkan fitur AI generatif tertentu dalam layanan. Topik ini berisi informasi tentang fitur yang diaktifkan oleh transfer data lintas wilayah, jenis data yang bergerak di seluruh Wilayah, dan cara memilih keluar.

Fitur-fitur berikut diaktifkan oleh transfer data lintas wilayah, dan tidak akan dapat diakses dalam instans Anda jika Anda memilih keluar:

1. Membuat aplikasi dengan AI, digunakan untuk memulai pembuatan aplikasi dengan mendeskripsikan aplikasi Anda dengan bahasa alami dan membuat sumber daya untuk Anda.
2. Obrolan AI di studio aplikasi, digunakan untuk mengajukan pertanyaan tentang pembuatan aplikasi, penerbitan, dan berbagi.

Data berikut ditransfer ke seluruh Wilayah:

1. Petunjuk atau masukan pengguna dari fitur yang dijelaskan sebelumnya.

Untuk memilih keluar dari transfer data lintas wilayah, dan fitur yang diaktifkan olehnya, gunakan prosedur berikut untuk mengisi formulir permintaan opt-out dari konsol:

1. Buka konsol App Studio di https://console.aws.amazon.com/appstudio/.
2. Pilih Opt out dari transfer data.
3. Masukkan ID AWS akun Anda, dan berikan alamat email Anda.
4. Pilih Kirim.
5. Setelah dikirimkan, permintaan Anda untuk memilih keluar dari transfer data lintas wilayah akan diproses, yang dapat memakan waktu hingga 60 hari.

Browser yang didukung untuk AWS App Studio

Topik ini berisi informasi tentang browser yang didukung dan direkomendasikan untuk AWS App Studio, termasuk dukungan browser untuk pengguna akhir yang mengakses aplikasi yang dipublikasikan, dan pembuat aplikasi.

Browser yang didukung dan direkomendasikan untuk membangun aplikasi

Untuk pengalaman pembuatan aplikasi yang optimal, App Studio mendukung dan sangat merekomendasikan penggunaan Google Chrome.

> **Note**
>
> Meskipun tidak disarankan, Anda juga dapat membangun aplikasi menggunakan browser web populer lainnya seperti Mozilla Firefox, Microsoft Edge, atau Apple Safari untuk macOS, tetapi perhatikan bahwa browser ini tidak didukung atau divalidasi secara resmi, dan Anda mungkin perlu memperbarui pengaturan untuk mengakses beberapa fitur pembangun. Untuk informasi selengkapnya, lihat Perbarui setelan browser untuk membangun aplikasi di App Studio.

App Studio tidak mendukung pembuatan aplikasi dari platform seluler.

Browser yang didukung dan direkomendasikan untuk pengguna akhir aplikasi

Untuk pengguna akhir yang mengakses aplikasi yang dipublikasikan, App Studio sangat merekomendasikan penggunaan Google Chrome atau Mozilla Firefox. Meskipun itu adalah browser yang direkomendasikan, pengguna akhir juga dapat mengakses aplikasi yang dipublikasikan dengan browser web populer lainnya, seperti Microsoft Edge atau Apple Safari untuk macOS.

Pengguna akhir juga dapat mengakses aplikasi yang dipublikasikan dari platform seluler.

Perbarui setelan browser untuk membangun aplikasi di App Studio

App Studio secara resmi mendukung dan merekomendasikan penggunaan Google Chrome untuk membangun aplikasi. Namun, jika Anda ingin menggunakan browser lain untuk membangun aplikasi, Anda mungkin harus memperbarui pengaturan atau cookie tertentu yang terkait dengan pelacakan lintas situs untuk mengakses halaman tertentu di App Studio.

Untuk Mozilla Firefox: Untuk melihat pratinjau aplikasi, perbarui pengaturan berikut: `Firefox Settings > Privacy & Security > Enhanced Tracking Protection` ke`Custom > Cookies > Cross-site tracking cookies`.

Untuk Apple Safari untuk macOS: Untuk membangun atau mempratinjau aplikasi, nonaktifkan pengaturan berikut:. `Settings > Privacy > Prevent cross-site tracking`

Kuota untuk AWS App Studio

Tabel berikut menjelaskan kuota dan batas untuk AWS App Studio.

Jumlah maksimum aplikasi dalam instans App Studio	20
Jumlah maksimum aplikasi yang dipublikasikan ke lingkungan Pengujian atau Produksi dalam instans App Studio. Sebuah aplikasi tunggal yang diterbitkan untuk Pengujian dan Produksi dihitung sebagai dua aplikasi yang diterbitkan.	6
Jumlah maksimum entitas terkelola per aplikasi	20
Jumlah maksimum baris yang dikembalikan per kueri	3000
Jumlah maksimum baris data sampel per entitas	500
Waktu kerja maksimum otomatisasi	2 menit. Otomatisasi yang berjalan lebih dari 2 menit akan gagal.
Ukuran input dan output otomatisasi maksimum	5GB per input atau output.
Ukuran data maksimum yang digunakan oleh otomatisasi atau tindakan data	450MB per otomatisasi atau tindakan data dijalankan.
Nama halaman dan nama komponen	Harus tidak kosong dan unik. Harus berisi hanya huruf, angka garis bawah (_) dan tanda dolar ($). Tidak dapat berisi spasi.

Riwayat dokumen untuk Panduan Pengguna AWS App Studio

Tabel berikut menjelaskan rilis dokumentasi untuk AWS App Studio.

Perubahan	Deskripsi	Tanggal
Topik yang diperbarui: Menggunakan JavaScript untuk menulis ekspresi	Menata ulang dan menambahkan informasi tentang referensi atau memperbarui komponen UI dan data tabel menggunak an ekspresi dalam aplikasi. Untuk informasi selengkapnya, lihat Menggunakan JavaScript untuk menulis ekspresi di App Studio.	Februari 18, 2025
Topik yang diperbarui: Pengaturan keamanan konten aplikasi	Menambahkan informasi tentang setelan keamanan konten aplikasi sumber konten. Anda dapat menggunakan setelan ini untuk membatasi domain tempat aplikasi Anda dapat mengunggah objek ke Amazon S3. Untuk informasi selengkapnya, lihat Melihat atau memperbarui setelan keamanan konten aplikasi Anda.	Februari 14, 2025
Topik baru: Memanggil fungsi Lambda di aplikasi App Studio	Menambahkan tutorial singkat yang merinci cara menjalank an fungsi Lambda di aplikasi App Studio. Untuk informasi	Januari 24, 2025

	lebih lanjut, lihat Memanggil fungsi Lambda.	
Topik baru: Connect ke Amazon SES	Menambahkan petunjuk untuk membuat konektor Amazon SES untuk menggunakan layanan di aplikasi App Studio. Untuk informasi selengkap nya, lihat Connect to Amazon Simple Email Service.	Januari 16, 2025
Topik yang diperbarui: Membuat dan menyiapka n instans App Studio untuk pertama kalinya	Menambahkan instruksi untuk menggunakan metode pembuatan yang mudah untuk membuat instance App Studio agar lebih cepat memulai. Untuk informasi selengkapnya, lihat Membuat dan menyiapka n instans App Studio untuk pertama kalinya.	Desember 13, 2024
Topik baru: Praktik terbaik untuk mengelola ketergant ungan data dan masalah waktu	Menambahkan dokumentasi tentang mengelola dependens i data dan masalah waktu dengan baik di aplikasi App Studio. Untuk informasi selengkapnya, lihat Ketergant ungan data dan pertimbangan waktu.	November 20, 2024

Topik yang diperbarui: Mengedit aplikasi Anda dengan AI	Menambahkan dokumentasi dengan informasi tentang mengedit aplikasi Anda menggunakan obrolan AI di studio aplikasi. Untuk informasi selengkapnya, lihat Membangun aplikasi App Studio Anda dengan AI generatif.	November 18, 2024
Topik yang diperbarui: Gunakan AI untuk menghasilkan JavaScript untuk Anda	Memperbarui referensi tindakan JavaScript otomatisasi untuk menyertakan informasi tentang penggunaan AI untuk menghasilkan JavaScript untuk Anda. Untuk informasi selengkapnya, lihat tindakan JavaScript otomatisasi.	November 18, 2024
Topik yang diperbarui: Bangun aplikasi peringkas teks AI dengan Amazon Bedrock	Memperbarui tutorial prompt Amazon Bedrock untuk menggunakan tindakan GenAI Prompt yang baru dirilis. Untuk informasi selengkapnya, lihat Membuat aplikasi peringkas teks AI dengan Amazon Bedrock.	November 18, 2024
Topik baru: Ubah warna aplikasi Anda dengan tema aplikasi	Menambahkan topik dengan informasi tentang mengubah warna di aplikasi Anda menggunakan tema aplikasi. Untuk informasi selengkapnya, lihat Mengubah warna di aplikasi Anda dengan tema aplikasi.	November 18, 2024

Topik baru: Praktik terbaik model data	Menambahkan topik dengan praktik terbaik untuk membuat model data yang aman, kuat, dan dapat diskalakan untuk digunakan di aplikasi App Studio. Untuk informasi selengkapnya, lihat Praktik terbaik saat mendesain model data.	November 15, 2024
Topik yang diperbarui: Menghubungkan ke AWS layanan	Memperbarui kebijakan kepercayaan untuk disertaka `nsts:ExternalId` , yang diperlukan untuk peran IAM yang digunakan untuk membuat konektor ke AWS layanan. Untuk informasi selengkapnya, lihat Connect to AWS services.	November 13, 2024
Topik baru: Kembalikan atau kembalikan ke versi aplikasi yang diterbitkan sebelumnya	Menambahkan topik dengan informasi tentang memutar kembali atau mengembal ikan aplikasi ke versi yang diterbitkan sebelumnya. Untuk informasi selengkapnya, lihat Mengembalikan ke versi yang diterbitkan sebelumnya.	November 13, 2024
Topik baru: Menghapus instance App Studio	Menambahkan topik yang menyertakan informasi tentang menghapus instance App Studio, termasuk petunjuk cara menghapusnya. Untuk informasi selengkapnya, lihat Menghapus instance App Studio.	November 12, 2024

Topik baru: Memperbarui pengaturan keamanan konten aplikasi	Menambahkan topik yang menyertakan informasi tentang setelan keamanan konten aplikasi di App Studio, termasuk cara memperbar uinya. Untuk informasi selengkapnya, lihat Melihat atau memperbarui setelan keamanan konten aplikasi Anda.	November 8, 2024
Topik yang diperbarui: Keamanan di AWS App Studio	Memperluas dokumentasi keamanan, termasuk informasi tentang perlindungan data dan cara App Studio berinteraksi dengan IAM. Untuk informasi selengkapnya, lihat Keamanan di AWS App Studio.	6 November 2024
Topik yang diperbarui: Kuota di AWS App Studio	Memperbarui kuota layanan App Studio dan membatasi dokumentasi untuk memperbai ki nilai yang salah dan menghapus beberapa kuota. Untuk informasi selengkap nya, lihat Kuota di AWS App Studio.	Oktober 21, 2024

Topik yang diperbarui: Menghubungkan App Studio ke AWS layanan lain	Memperbarui dokumentasi untuk menghubungkan ke AWS layanan agar lebih mematuhi praktik keamanan terbaik dengan memberikan instruksi dan contoh untuk memberikan izin minimal yang diperlukan kepada App Studio ke layanan atau sumber daya. Untuk informasi selengkapnya, lihat Connect to AWS services.	Oktober 18, 2024
Topik diperbarui: Ditambahkan dukungan versi untuk dokumentasi konektor Aurora	Menambahkan daftar versi yang didukung ke dokumentasi konektor Aurora. Untuk informasi selengkapnya, lihat Connect to Amazon Aurora.	Oktober 16, 2024
Topik baru: Browser yang didukung untuk App Studio	Menambahkan topik yang mencakup dukungan browser dan rekomendasi untuk menggunakan App Studio. Untuk informasi selengkapnya, lihat Browser yang didukung.	Oktober 10, 2024
Topik baru: Cara kerja AWS App Studio	Menambahkan topik yang membahas konsep utama pengembangan aplikasi di App Studio, termasuk diagram dan tangkapan layar. Untuk informasi selengkapnya, lihat Cara kerja App Studio.	Oktober 10, 2024

Topik baru: Memesan dan mengatur halaman	Menambahkan topik yang mencakup informasi tentang menyusun ulang dan menyembunyikan atau menampilkan halaman dalam navigasi pratinjau atau aplikasi yang dipublikasikan. Untuk informasi selengkapnya, lihat Mengurutkan dan mengatur halaman.	September 24, 2024
Topik baru: Kuota di AWS App Studio	Menambahkan topik yang menyertakan kuota dan batasan layanan yang terkait dengan App Studio. Untuk informasi selengkapnya, lihat Kuota di AWS App Studio.	September 11, 2024
Topik yang diperbarui: Connect ke tabel DynamoDB terenkripsi	Menambahkan informasi, seperti izin yang diperluka n, untuk menggunakan tabel DynamoDB yang dienkrips i AWS KMS dengan kunci yang dikelola pelanggan () dengan App Studio. CMKs Untuk informasi selengkapnya, lihat Connect to DynamoDB.	September 6, 2024
Topik yang diperbarui: Connect ke DynamoDB, Amazon Redshift, dan Aurora	Menambahkan izin minimum yang diperlukan untuk ditambahkan ke peran IAM untuk menggunakan sumber daya DynamoDB, Amazon Redshift, dan Aurora dengan aplikasi App Studio. Untuk informasi selengkapnya, lihat Connect to AWS services.	September 5, 2024

Topik yang diperbarui: Connect ke Amazon Aurora	Memperbarui dokumentasi untuk membuat dan mengonfigurasi database dan tabel Amazon Aurora untuk digunakan dengan aplikasi App Studio. Untuk informasi selengkapnya, lihat Connect to Amazon Aurora.	September 5, 2024
Topik baru dan yang diperbarui: Pemecahan masalah dan debugging	Memperluas dokumentasi pemecahan masalah dan debugging untuk membantu menyelesaikan masalah umum dengan App Studio, termasuk informasi debugging untuk membangun aplikasi. Untuk informasi selengkapnya, lihat Memecahkan masalah dan men-debug App Studio.	Agustus 26, 2024
Topik baru: Tutorial: Membangun aplikasi peringkas teks AI dengan Amazon Bedrock	Ikuti langkah-langkah dalam tutorial untuk membuat aplikasi yang menerima prompt input dari pengguna akhir, mengirimkannya ke Amazon Bedrock dan mengembalikan serta menampilkan versi yang diringkas. Untuk informasi selengkapnya, lihat Membuat aplikasi peringkas teks AI dengan Amazon Bedrock.	Agustus 20, 2024

Topik yang diperbarui: Mempratinjau, menerbitkan, dan berbagi aplikasi App Studio	Memperluas dokumentasi pratinjau, penerbitan, dan berbagi untuk menambah kejelasan, mencocokkan pengalaman dalam layanan, dan memberikan informasi tambahan seputar lingkungan penerbitan dan melihat aplikasi di dalamnya. Untuk informasi selengkapnya, lihat Mempratinjau, menerbitkan, dan berbagi aplikasi.	Agustus 2, 2024
Topik baru: Membangun aplikasi dengan banyak pengguna	Memperluas dokumentasi pratinjau, penerbitan, dan berbagi untuk menambah kejelasan, mencocokkan pengalaman dalam layanan, dan memberikan informasi tambahan seputar lingkungan penerbitan dan melihat aplikasi di dalamnya. Untuk informasi selengkapnya, lihat Membangun aplikasi dengan beberapa pengguna.	Agustus 2, 2024
Topik yang diperbarui: Menghubungkan App Studio ke AWS layanan	Menambahkan informasi tentang membuat dan menyediakan peran IAM untuk menyediakan akses ke AWS sumber daya saat membuat konektor AWS Layanan Lain. Untuk informasi selengkapnya, lihat Connect to AWS services menggunakan konektor AWS Layanan lain.	Juli 29, 2024

Topik yang diperbarui: Tambahkan instruksi untuk membuat pengguna AWS administratif sebagai bagian dari pengaturan	Menambahkan instruksi dalam pengaturan dokumentasi App Studio untuk membuat pengguna administratif untuk mengelola AWS sumber daya. Juga membuat pembaruan di seluruh dokumentasi konektor untuk merekomendasikan penggunaan pengguna itu.	Juli 24, 2024
Topik baru: Connect ke Amazon Bedrock	Menambahkan topik dengan instruksi untuk membuat konektor untuk Amazon Bedrock. Builder dapat menggunakan konektor untuk membangun aplikasi yang menggunakan Amazon Bedrock. Untuk informasi selengkapnya, lihat Connect to Amazon Bedrock.	Juli 24, 2024
Rilis awal	Rilis awal Panduan Pengguna AWS App Studio	Juli 10, 2024

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.