



Panduan Developer

# AWS Infrastructure Composer



# AWS Infrastructure Composer: Panduan Developer

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

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

---

# Table of Contents

|   |    |
|---|----|
| Apa itu Komposer Infrastruktur? .....                   | 1  |
| Tulis arsitektur Anda .....                             | 2  |
| Tentukan template Anda .....                            | 4  |
| Integrasikan dengan alur kerja Anda .....               | 5  |
| Cara mengakses Infrastructure Composer .....            | 6  |
| Pelajari selengkapnya .....                             | 8  |
| Langkah selanjutnya .....                               | 8  |
| Konsep tanpa server .....                               | 8  |
| Konsep tanpa server .....                               | 9  |
| Kartu .....   | 10 |
| Kartu komponen yang disempurnakan .....                 | 11 |
| Contoh .....  | 12 |
| Kartu komponen standar .....                            | 13 |
| Koneksi kartu .....                                     | 17 |
| Koneksi antar kartu .....                               | 17 |
| Koneksi antara kartu komponen yang disempurnakan .....  | 18 |
| Koneksi ke dan dari kartu sumber daya IAC standar ..... | 20 |
| Memulai .....   | 21 |
| Ikuti tur konsol .....                                  | 21 |
| Langkah selanjutnya .....                               | 22 |
| Memuat dan memodifikasi .....                           | 22 |
| Langkah 1: Buka demo .....                              | 23 |
| Langkah 2: Jelajahi kanvas visual .....                 | 23 |
| Langkah 3: Perluas arsitektur Anda .....                | 27 |
| Langkah 4: Simpan aplikasi Anda .....                   | 28 |
| Langkah selanjutnya .....                               | 29 |
| Membangun .....   | 29 |
| Properti sumber daya .....                              | 30 |
| Langkah 1: Buat proyek Anda .....                       | 30 |
| Tambahkan kartu .....                                   | 33 |
| Langkah 3: Konfigurasi REST API .....                   | 34 |
| Langkah 4: Konfigurasi fungsi Anda .....                | 35 |
| Langkah 5: Hubungkan kartu Anda .....                   | 36 |
| Langkah 6: Atur kanvas .....                            | 37 |

|   |    |
|---|----|
| Tambahkan tabel DynamoDB .....                            | 38 |
| Langkah 8: Tinjau template Anda .....                     | 39 |
| Langkah 9: Integrasikan ke dalam alur kerja Anda .....    | 40 |
| Langkah selanjutnya .....                                 | 40 |
| Dimana menggunakan Infrastructure Composer .....          | 41 |
| Konsol Komposer Infrastruktur .....                       | 41 |
| Ikhtisar visual .....                                     | 42 |
| Kelola proyek Anda .....                                  | 45 |
| Connect ke lokal Anda IDE .....                           | 48 |
| Izinkan akses halaman web .....                           | 51 |
| Sinkronkan dan simpan secara lokal .....                  | 52 |
| Impor dari konsol Lambda .....                            | 55 |
| Ekspor kanvas .....                                       | 56 |
| CloudFormation modus konsol .....                         | 58 |
| Mengapa menggunakan mode ini? .....                       | 58 |
| Akses mode ini .....                                      | 59 |
| Visualisasikan penerapan .....                            | 59 |
| Buat template baru .....                                  | 60 |
| Perbarui tumpukan yang ada .....                          | 61 |
| AWS Toolkit for Visual Studio Code .....                  | 63 |
| Ikhtisar visual .....                                     | 64 |
| Akses dari VS Code .....                                  | 65 |
| Sinkronkan ke AWS Cloud .....                             | 66 |
| Komposer Infrastruktur dengan Amazon Q .....              | 68 |
| Cara menulis .....  | 71 |
| Tempatkan kartu di kanvas .....                           | 71 |
| Kartu grup bersama .....                                  | 72 |
| Mengelompokkan kartu komponen yang disempurnakan .....    | 72 |
| Mengelompokkan kartu komponen standar ke kartu lain ..... | 73 |
| Connect kartu .....                                       | 74 |
| Menghubungkan kartu komponen yang disempurnakan .....     | 75 |
| Menghubungkan kartu standar .....                         | 76 |
| Contoh .....  | 78 |
| Putuskan sambungan kartu .....                            | 80 |
| Kartu komponen yang disempurnakan .....                   | 80 |
| Kartu komponen standar .....                              | 80 |

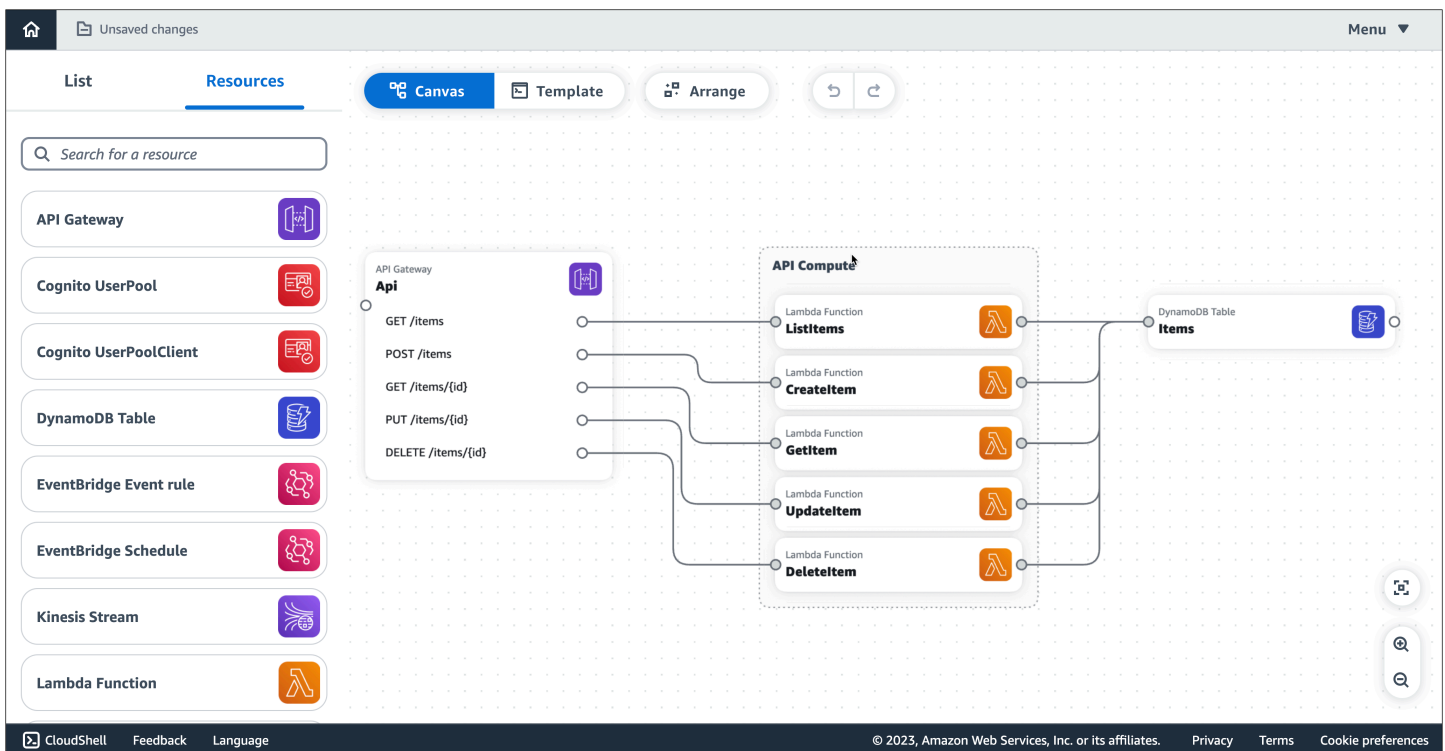
|   |     |
|---|-----|
| Atur kartu .....  | 82  |
| Konfigurasi dan modifikasi kartu .....                    | 83  |
| Kartu yang disempurnakan .....                            | 84  |
| Kartu standar .....                                       | 100 |
| Hapus kartu .....   | 101 |
| Kartu komponen yang disempurnakan .....                   | 101 |
| Kartu komponen standar .....                              | 101 |
| Lihat pembaruan kode .....                                | 102 |
| Keuntungan dari Change Inspector .....                    | 103 |
| Prosedur .....  | 103 |
| Pelajari selengkapnya .....                               | 105 |
| Referensi file eksternal .....                            | 106 |
| Praktik terbaik .....                                     | 107 |
| Buat referensi file eksternal .....                       | 107 |
| Memuat proyek .....                                       | 108 |
| Buat aplikasi menggunakan AWS SAM CLI .....               | 109 |
| Referensi sebuah OpenAPI spesifikasi .....                | 112 |
| Integrasikan dengan Amazon VPC .....                      | 115 |
| Identifikasi sumber daya dan informasi .....              | 116 |
| Konfigurasi fungsi .....                                  | 122 |
| Parameter dalam template yang diimpor .....               | 122 |
| Menambahkan parameter baru ke template yang diimpor ..... | 125 |
| Konfigurasi fungsi Lambda dengan template VPC lain .....  | 126 |
| Menyebarkan ke Cloud AWS .....                            | 130 |
| AWS SAM Konsep penting .....                              | 130 |
| Langkah selanjutnya .....                                 | 130 |
| Mengatur AWS SAM CLI .....                                | 131 |
| Instal AWS CLI .....                                      | 131 |
| Instal AWS SAM CLI .....                                  | 131 |
| Akses AWS SAM CLI .....                                   | 131 |
| Langkah selanjutnya .....                                 | 132 |
| Membangun dan menyebarkan .....                           | 132 |
| Hapus tumpukan .....                                      | 140 |
| Pemecahan Masalah .....                                   | 142 |
| Pesan kesalahan .....                                     | 142 |
| “Tidak dapat membuka folder ini” .....                    | 142 |

|   |       |
|---|-------|
| “Template yang tidak kompatibel” .....  | 142   |
| “Folder yang disediakan berisi template.yaml yang ada” .....                          | 143   |
| “Browser Anda tidak memiliki izin untuk menyimpan proyek Anda di folder itu...” ..... | 144   |
| Keamanan .....  | 145   |
| Perlindungan data .....   | 145   |
| Enkripsi data .....   | 147   |
| Enkripsi bergerak .....   | 147   |
| Manajemen kunci .....   | 147   |
| Privasi lalu lintas antar jaringan .....  | 147   |
| AWS Identity and Access Management .....  | 147   |
| Audiens .....   | 148   |
| Mengautentikasi dengan identitas .....  | 148   |
| Mengelola akses menggunakan kebijakan .....   | 152   |
| Bagaimana AWS Infrastructure Composer bekerja dengan IAM .....                        | 154   |
| Validasi kepatuhan .....  | 161   |
| Ketahanan .....   | 162   |
| Riwayat dokumen .....   | 164   |
| .....   | clxxi |

# Apa itu AWS Infrastructure Composer?

AWS Infrastructure Composer memungkinkan Anda untuk menulis aplikasi modern secara visual. AWS Lebih khusus lagi, Anda dapat menggunakan Infrastructure Composer untuk memvisualisasikan, membangun, dan menyebarkan aplikasi modern dari semua AWS layanan yang didukung oleh AWS CloudFormation tanpa perlu menjadi ahli dalam. AWS CloudFormation

Saat Anda menyusun AWS CloudFormation infrastruktur Anda, melalui drag-and-drop antarmuka yang menyenangkan, Infrastructure Composer membuat templat infrastruktur Anda sebagai kode (IaC), sambil mengikuti praktik terbaik. AWS Gambar berikut menunjukkan betapa mudahnya menyeret, melepas, mengkonfigurasi, dan menghubungkan sumber daya pada kanvas visual Infrastructure Composer.



Infrastructure Composer dapat digunakan dari konsol Infrastructure Composer, the AWS Toolkit for Visual Studio Code, dan dalam mode CloudFormation konsol.

## Topik

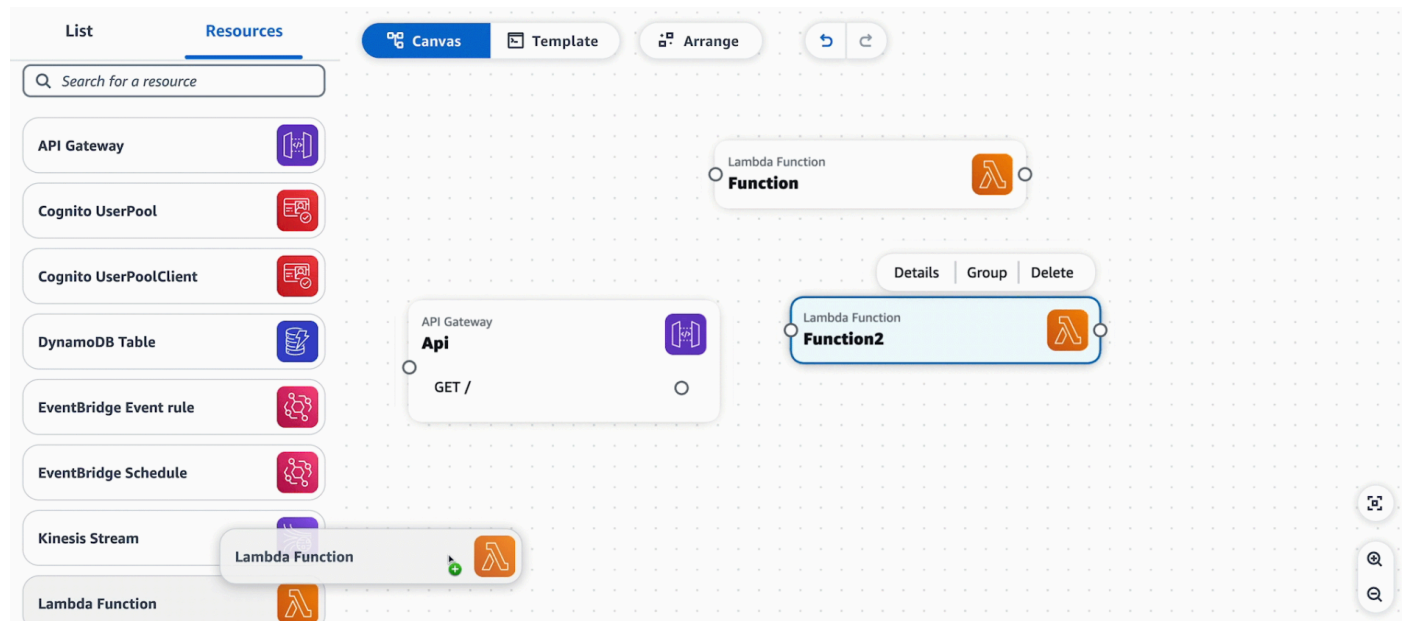
- [Tulis arsitektur aplikasi Anda](#)
- [Tentukan infrastruktur Anda sebagai templat kode \(IaC\)](#)
- [Integrasikan dengan alur kerja yang ada](#)

- [Cara mengakses Infrastructure Composer](#)
- [Pelajari selengkapnya](#)
- [Langkah selanjutnya](#)
- [Konsep tanpa server untuk AWS Infrastructure Composer](#)

## Tulis arsitektur aplikasi Anda

### Bangun dengan kartu

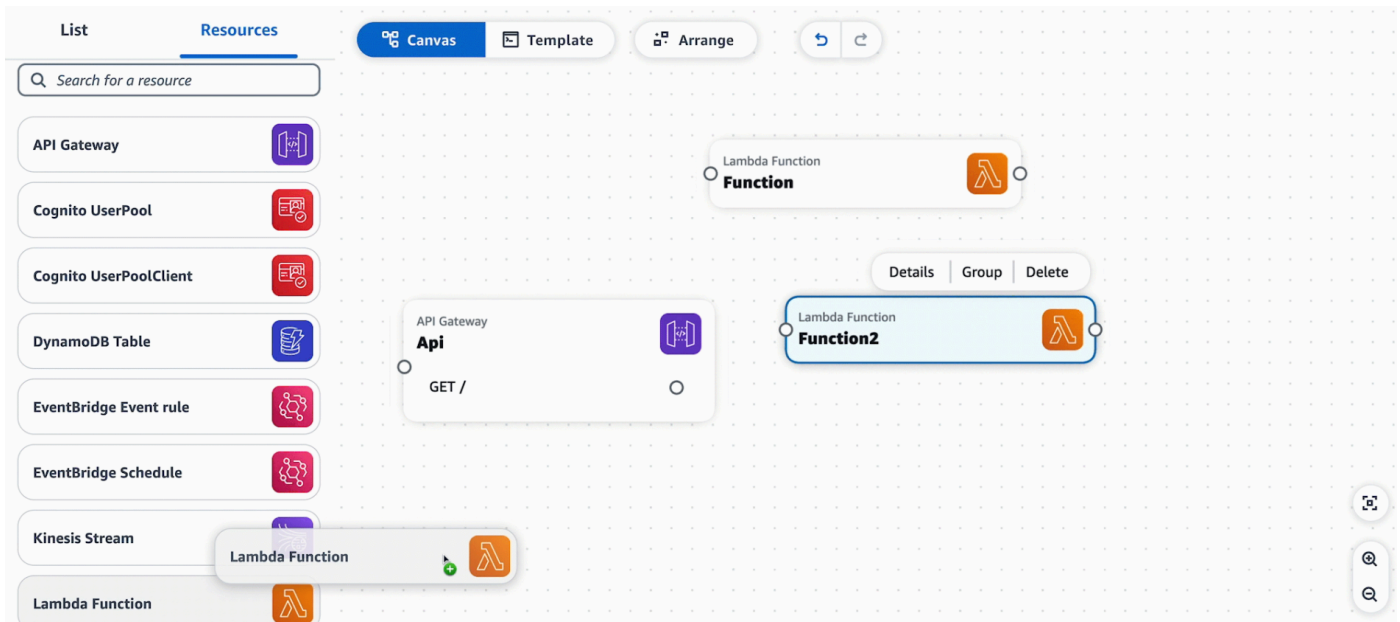
Tempatkan kartu pada kanvas Infrastructure Composer untuk memvisualisasikan dan membangun arsitektur aplikasi Anda.



### Connect kartu bersama-sama

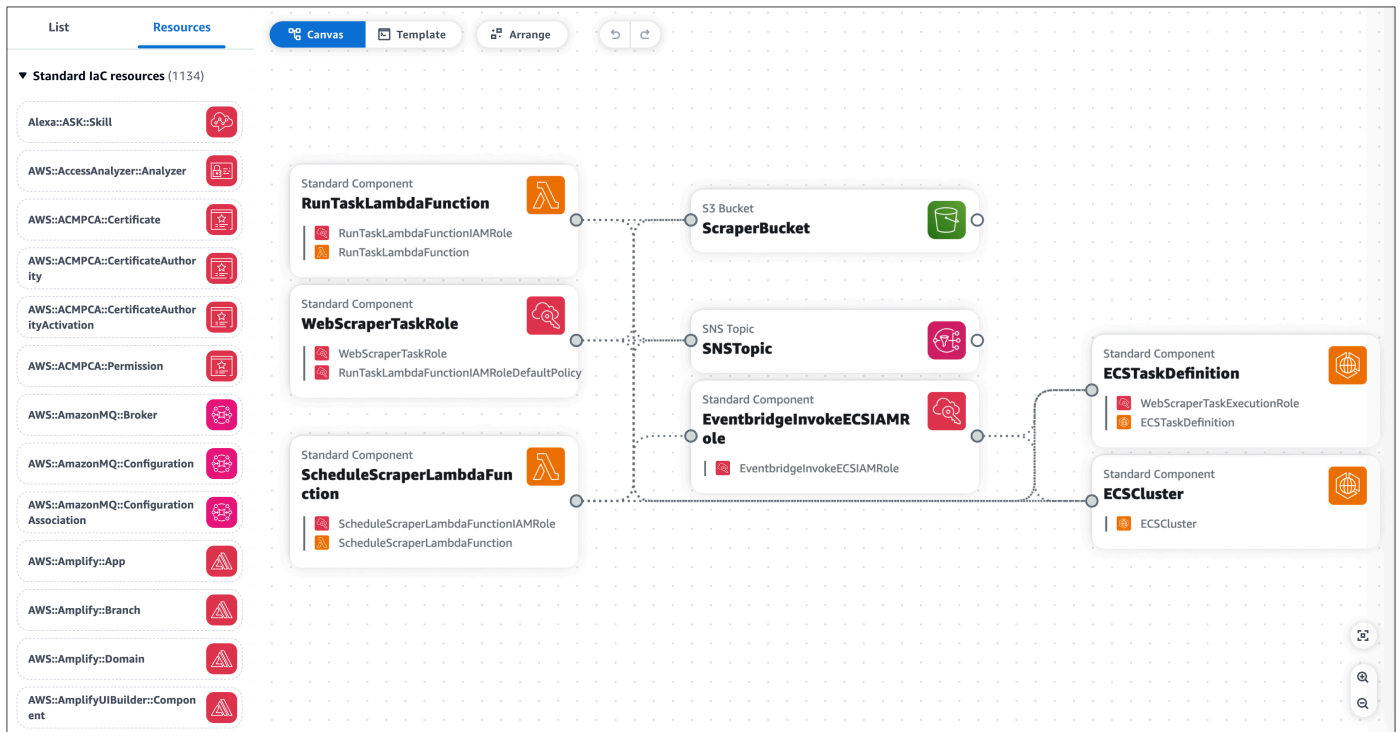
Konfigurasi bagaimana sumber daya Anda berinteraksi satu sama lain dengan menghubungkannya secara visual. Tentukan properti mereka lebih lanjut melalui panel properti yang dikuratori.





### Bekerja dengan AWS CloudFormation sumber daya apa pun

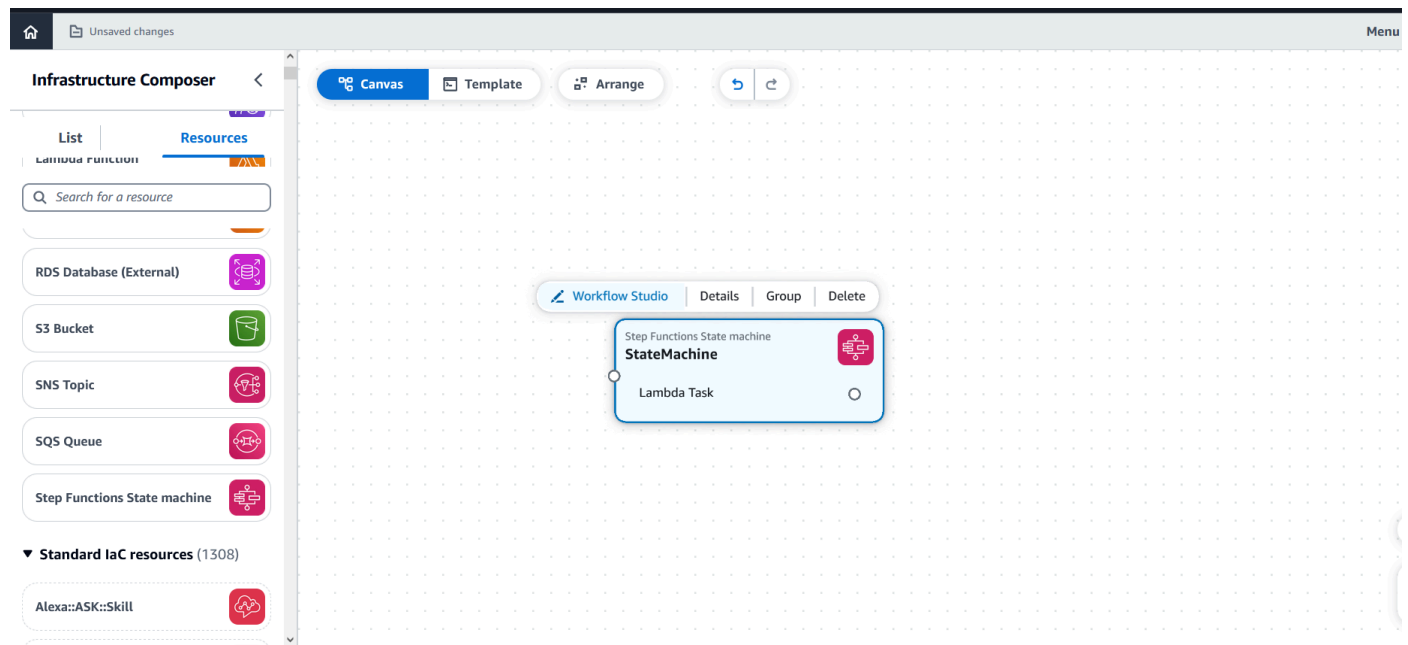
Seret AWS CloudFormation sumber daya apa pun ke kanvas untuk membuat arsitektur aplikasi Anda. Infrastructure Composer menyediakan template IaC awal yang dapat Anda gunakan untuk menentukan properti sumber daya Anda. Untuk mempelajari selengkapnya, lihat [Mengkonfigurasi dan memodifikasi kartu di Infrastructure Composer](#).



## Akses kemampuan tambahan dengan unggulan Layanan AWS

Fitur Infrastructure Composer Layanan AWS yang biasa digunakan atau dikonfigurasi bersama saat membangun aplikasi. Untuk mempelajari selengkapnya, lihat [Integrasikan dengan Amazon VPC](#).

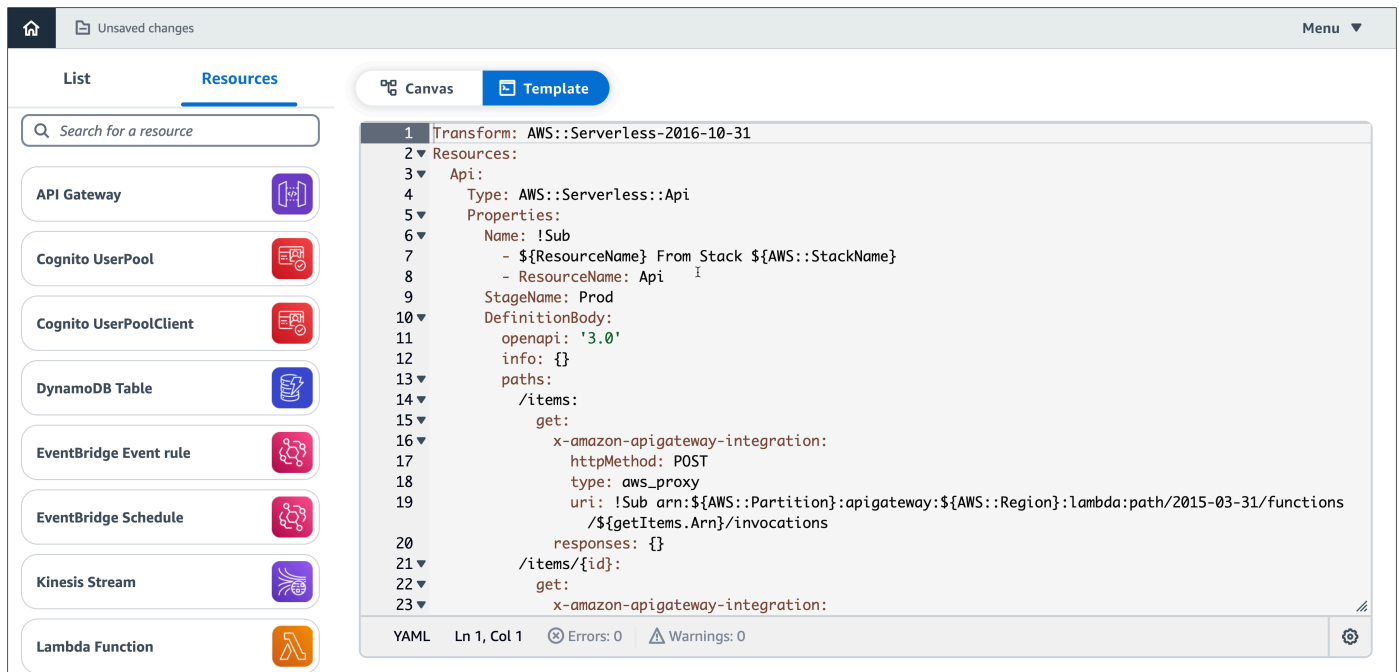
Berikut ini adalah contoh AWS Step Functions fitur, yang menyediakan integrasi untuk meluncurkan Step Functions Workflow Studio langsung di dalam kanvas Infrastructure Composer.



## Tentukan infrastruktur Anda sebagai templat kode (IAC)

Infrastructure Composer membuat kode infrastruktur Anda

Saat Anda menulis, Infrastructure Composer secara otomatis membuat template AWS CloudFormation dan AWS Serverless Application Model (AWS SAM) Anda, mengikuti praktik AWS terbaik. Anda dapat melihat dan memodifikasi template Anda langsung dari dalam Infrastructure Composer. Infrastructure Composer secara otomatis menyinkronkan perubahan antara kanvas visual dan kode template Anda.



The screenshot displays the AWS Infrastructure Composer interface. On the left, there is a 'List' view showing various AWS resources with their respective icons: API Gateway, Cognito UserPool, Cognito UserPoolClient, DynamoDB Table, EventBridge Event rule, EventBridge Schedule, Kinesis Stream, and Lambda Function. The 'Resources' tab is active. On the right, the 'Canvas' view shows a YAML template for an API Gateway resource. The template is as follows:

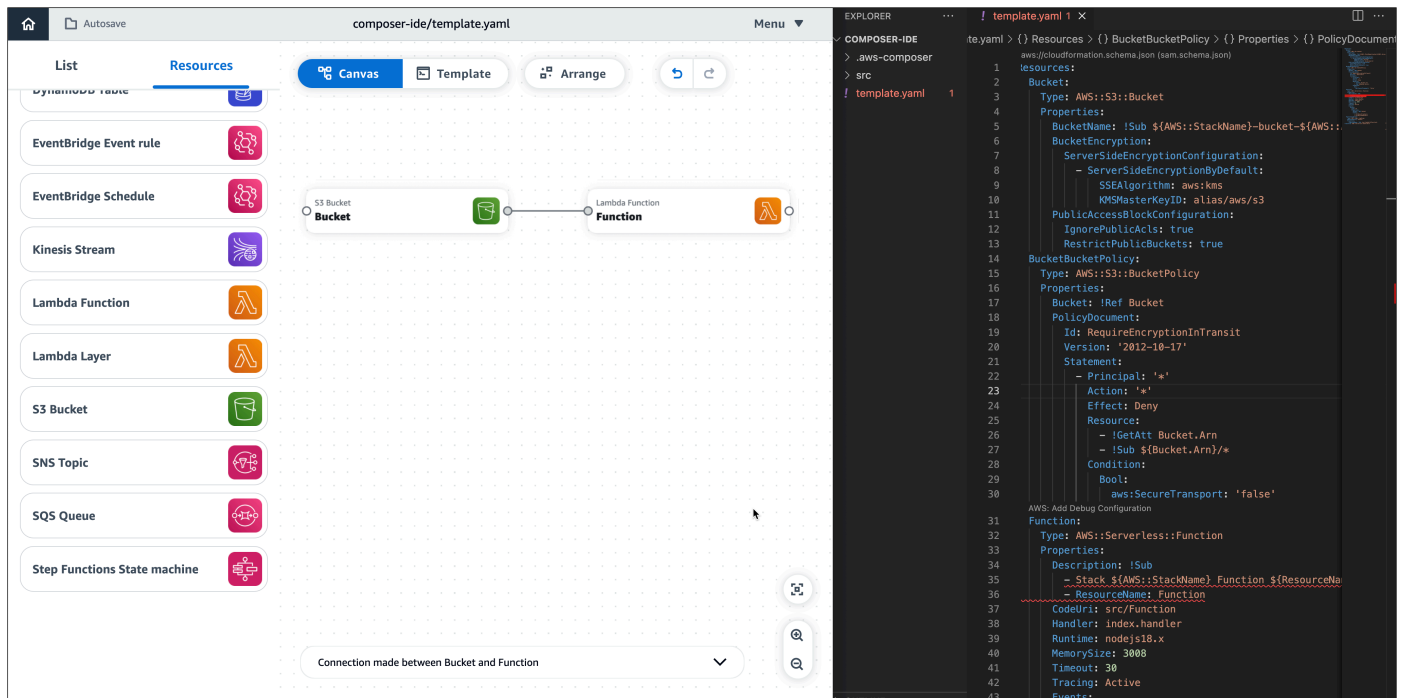
```
1 Transform: AWS::Serverless-2016-10-31
2 Resources:
3   Api:
4     Type: AWS::Serverless::Api
5     Properties:
6       Name: !Sub
7         - ${ResourceName} From Stack ${AWS::StackName}
8         - ResourceName: Api
9     StageName: Prod
10    DefinitionBody:
11      openapi: '3.0'
12      info: {}
13      paths:
14        /items:
15          get:
16            x-amazon-apigateway-integration:
17              httpMethod: POST
18              type: aws_proxy
19              uri: !Sub arn:${AWS::Partition}:apigateway:${AWS::Region}:lambda:path/2015-03-31/functions
20                /${getItems.Arn}/invocations
21            responses: {}
22        /items/{id}:
23          get:
24            x-amazon-apigateway-integration:
```

The status bar at the bottom indicates 'YAML Ln 1, Col 1', 'Errors: 0', and 'Warnings: 0'.

## Integrasikan dengan alur kerja yang ada

Impor templat dan proyek yang ada

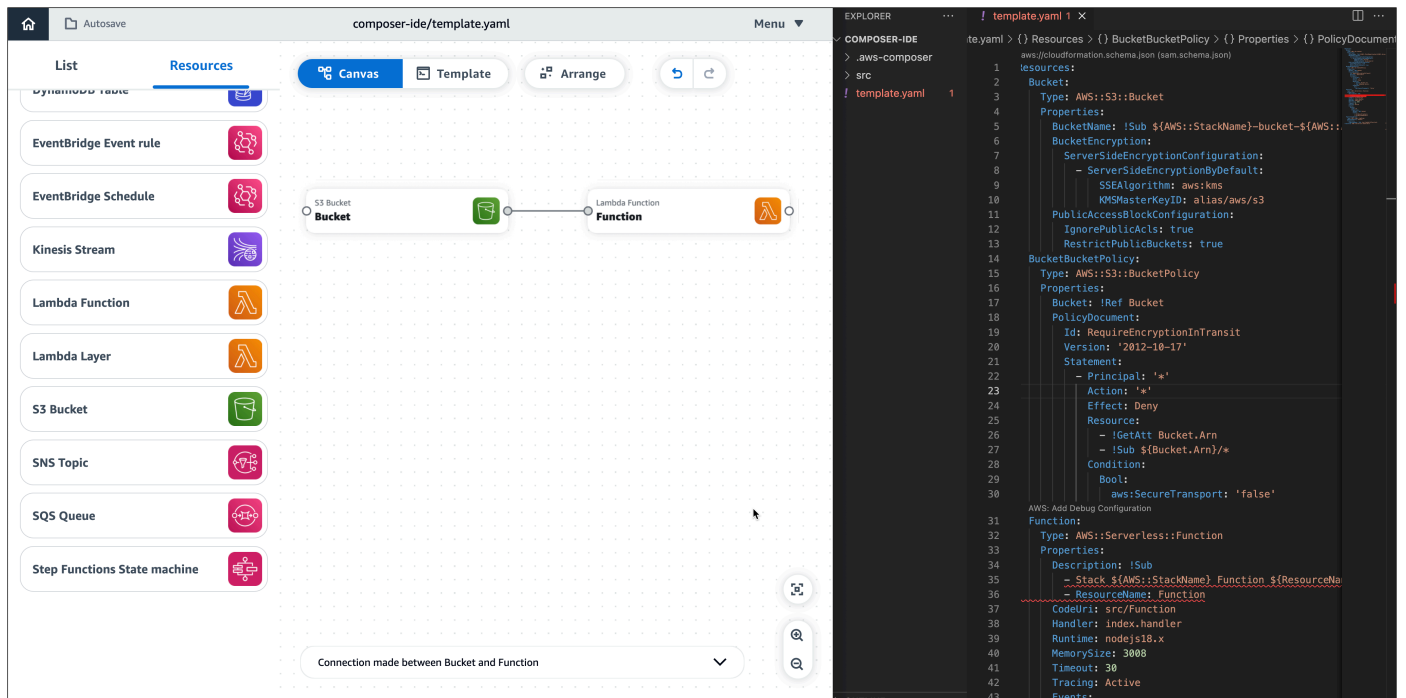
Impor yang ada AWS CloudFormation dan AWS SAM templat untuk memvisualisasikannya agar lebih memahami dan memodifikasi desainnya. Ekspor template yang Anda buat dalam Infrastructure Composer dan integrasikan ke dalam alur kerja yang ada menuju penerapan.



## Cara mengakses Infrastructure Composer

Dari konsol Infrastructure Composer

Akses Infrastructure Composer melalui konsol Infrastructure Composer untuk memulai dengan cepat. Selain itu, Anda dapat menggunakan mode sinkronisasi lokal untuk secara otomatis menyinkronkan dan menyimpan Infrastructure Composer dengan mesin lokal Anda.



Dari AWS CloudFormation konsol

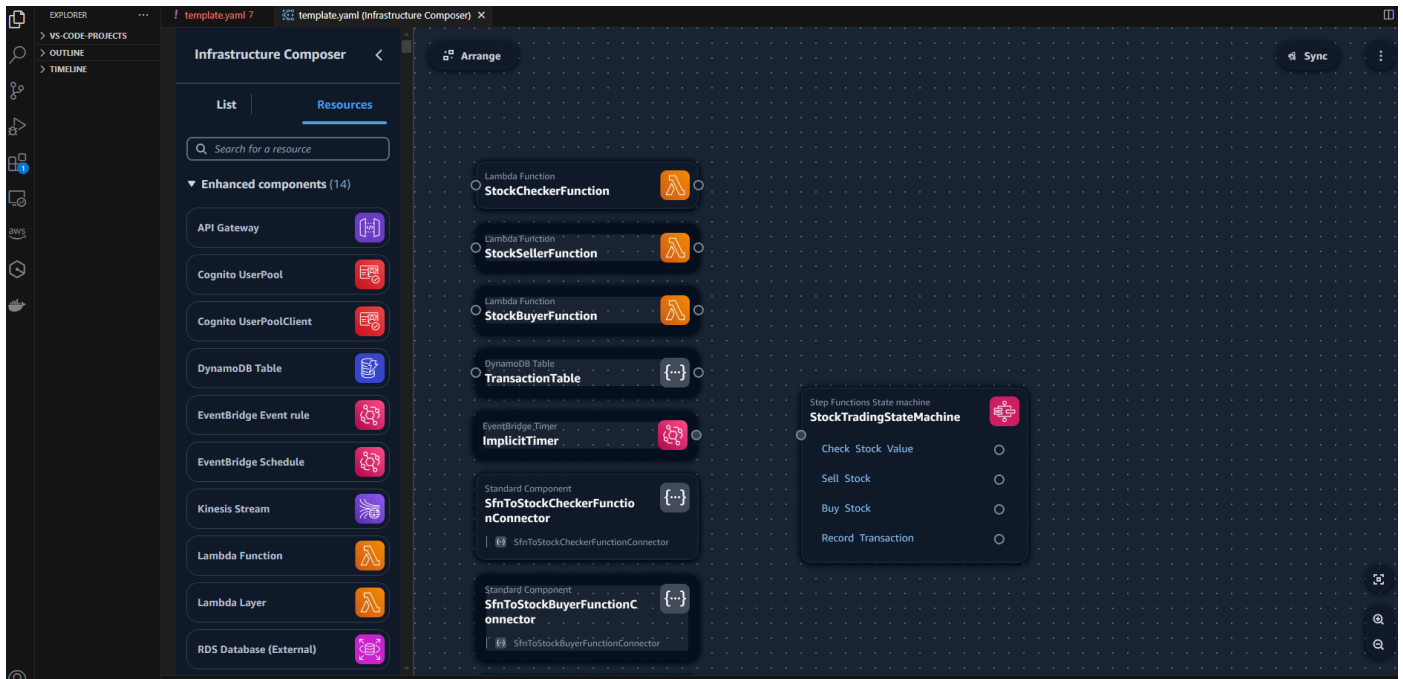
Konsol Infrastructure Composer juga mendukung [mode CloudFormation konsol](#), peningkatan dari CloudFormation Designer yang terintegrasi dengan alur kerja AWS CloudFormation tumpukan. Alat baru ini sekarang menjadi alat yang direkomendasikan untuk memvisualisasikan CloudFormation template Anda.

Dari konsol Lambda

Dengan Infrastructure Composer, Anda juga dapat mengimpor fungsi Lambda dari konsol Lambda. Untuk mempelajari selengkapnya, lihat [Impor fungsi ke Infrastructure Composer dari konsol Lambda](#).

Dari AWS Toolkit for Visual Studio Code

Akses Infrastructure Composer melalui ekstensi Toolkit for VS Code untuk membawa Infrastructure Composer ke lingkungan pengembangan lokal Anda.



## Pelajari selengkapnya

Untuk terus belajar tentang Infrastructure Composer, lihat sumber daya berikut:

- [Kartu Komposer Infrastruktur](#)
- [Buat dan buat aplikasi tanpa server secara visual | Jam Kantor Tanpa Server](#) - Ikhtisar dan demo Infrastructure Composer.

## Langkah selanjutnya

Untuk mengatur Infrastructure Composer, lihat [Memulai dengan konsol Infrastructure Composer](#).

## Konsep tanpa server untuk AWS Infrastructure Composer

Pelajari tentang konsep dasar tanpa server sebelum menggunakan. AWS Infrastructure Composer

# Konsep tanpa server

## Arsitektur berbasis peristiwa

Aplikasi tanpa server terdiri dari AWS layanan individual, seperti AWS Lambda untuk komputasi dan Amazon DynamoDB untuk manajemen basis data, yang masing-masing melakukan peran khusus. Layanan ini kemudian terintegrasi secara longgar satu sama lain melalui arsitektur berbasis peristiwa. Untuk mempelajari lebih lanjut tentang arsitektur berbasis peristiwa, lihat [Apa itu Arsitektur Berbasis Acara?](#) .

## Infrastruktur sebagai Kode (IaC)

Infrastructure as Code (IaC) adalah cara memperlakukan infrastruktur dengan cara yang sama seperti pengembang memperlakukan kode, menerapkan ketelitian pengembangan kode aplikasi yang sama untuk penyediaan infrastruktur. Anda menentukan infrastruktur Anda dalam file template, menyebarkannya ke AWS, dan AWS membuat sumber daya untuk Anda. Dengan IaC, Anda menentukan dalam kode apa yang AWS ingin Anda berikan. Untuk informasi selengkapnya, lihat [Infrastruktur sebagai Kode](#) di Pengantar DevOps pada AWS AWS Whitepaper.

## Teknologi tanpa server

Dengan teknologi AWS tanpa server, Anda dapat membangun dan menjalankan aplikasi tanpa harus mengelola server Anda sendiri. Semua manajemen server dilakukan oleh AWS, memberikan banyak manfaat seperti penskalaan otomatis dan ketersediaan tinggi bawaan, memungkinkan Anda membawa ide Anda ke produksi dengan cepat. Menggunakan teknologi tanpa server, Anda dapat fokus pada inti produk Anda tanpa harus khawatir tentang mengelola dan mengoperasikan server. Untuk mempelajari lebih lanjut tentang tanpa server, lihat [Serverless on. AWS](#)

Untuk pengenalan dasar tentang layanan tanpa AWS server inti, lihat Serverless [101: Memahami layanan tanpa server di Serverless Land](#).

# Kartu Komposer Infrastruktur

Infrastructure Composer menyederhanakan proses penulisan infrastruktur sebagai kode (IaC) untuk sumber daya. AWS CloudFormation Untuk menggunakan Infrastructure Composer secara efektif, ada dua konsep dasar yang harus Anda pahami terlebih dahulu: kartu Infrastructure Composer dan koneksi [kartu](#).

Dalam Infrastructure Composer, kartu mewakili AWS CloudFormation sumber daya. ada dua kategori umum kartu:

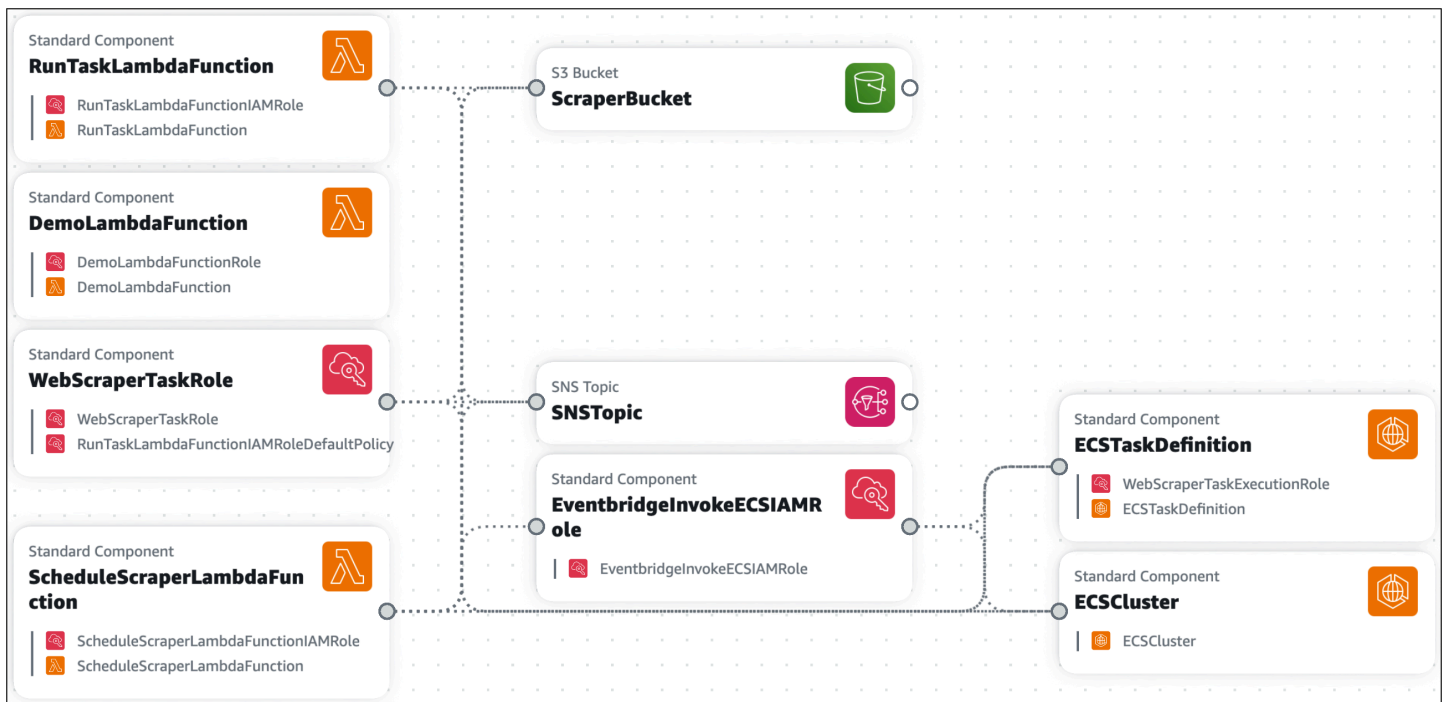
- [Kartu komponen yang disempurnakan](#) — Kumpulan AWS CloudFormation sumber daya yang telah digabungkan menjadi satu kartu kurasi yang meningkatkan kemudahan penggunaan, fungsionalitas, dan dirancang untuk berbagai kasus penggunaan. Kartu komponen yang disempurnakan adalah kartu pertama yang tercantum dalam palet Sumber Daya di Infrastructure Composer.
- [Kartu sumber daya IAC standar](#) — AWS CloudFormation Sumber daya tunggal. Setiap kartu sumber daya IAC standar, setelah diseret ke kanvas, diberi label komponen Standar dan dapat digabungkan menjadi beberapa sumber daya.

## Note

Tergantung pada kartunya, kartu sumber daya Standar IAC dapat diberi label kartu komponen Standar setelah diseret ke kanvas visual. Ini berarti kartu adalah kumpulan dari satu atau lebih kartu sumber daya IAC standar.

Sementara beberapa jenis kartu tersedia dari palet Resources, kartu juga dapat muncul di kanvas ketika Anda mengimpor template AWS CloudFormation atau AWS Serverless Application Model (AWS SAM) yang ada ke Infrastructure Composer. Gambar berikut adalah contoh aplikasi impor yang berisi berbagai jenis kartu:





## Topik

- [Kartu komponen yang disempurnakan di Infrastructure Composer](#)
- [Kartu komponen standar di Infrastructure Composer](#)
- [Koneksi kartu di Infrastructure Composer](#)

## Kartu komponen yang disempurnakan di Infrastructure Composer

Kartu komponen yang disempurnakan dibuat dan dikelola oleh Infrastructure Composer. Setiap kartu berisi AWS CloudFormation sumber daya yang biasa digunakan bersama saat membangun aplikasi AWS. Kode infrastruktur mereka dibuat oleh Infrastructure Composer mengikuti praktik AWS terbaik. Kartu komponen yang disempurnakan adalah cara yang bagus untuk mulai merancang aplikasi Anda.

Kartu komponen yang disempurnakan tersedia dari palet Sumber Daya, di bawah bagian Komponen yang Ditingkatkan.

Kartu komponen yang disempurnakan dapat sepenuhnya dikonfigurasi dan digunakan dalam Infrastructure Composer untuk merancang dan membangun aplikasi tanpa server Anda. Sebaiknya gunakan kartu komponen yang disempurnakan saat mendesain aplikasi Anda tanpa kode yang ada.

Tabel ini menampilkan komponen kami yang disempurnakan dengan tautan ke spesifikasi templat AWS CloudFormation or AWS Serverless Application Model (AWS SAM) dari sumber daya unggulan kartu:

| Kartu  | Referensi   |
|--|---|
| APIGerbang Amazon  | <a href="#">AWS: :Tanpa server:: API</a>          |
| Amazon Cognito UserPool                                  | <a href="#">AWS: :Kognito:: UserPool</a>          |
| Amazon Cognito UserPoolClient                            | <a href="#">AWS: :Kognito:: UserPoolClient</a>    |
| Tabel Amazon DynamoDB                                    | <a href="#">AWS: :DynamoDB: :Tabel</a>            |
| Aturan EventBridge Acara Amazon                          | <a href="#">AWS: :Acara: :Aturan</a>              |
| EventBridge Jadwal                                       | <a href="#">AWS: :Penjadwal: :Jadwal</a>          |
| Aliran Kinesis Amazon                                    | <a href="#">AWS: :Kinesis: :Aliran</a>            |
| AWS Lambda Fungsi  | <a href="#">AWS: :Tanpa server: :Fungsi</a>       |
| Lapisan Lambda   | <a href="#">AWS: :Tanpa server:: LayerVersion</a> |
| Bucket Layanan Penyimpanan Sederhana Amazon (Amazon S3)  | <a href="#">AWS: :S3: :Ember</a>                  |
| Topik Layanan Pemberitahuan Sederhana Amazon (AmazonSNS) | <a href="#">AWS::SNS: :Topik</a>                  |
| Antrian Layanan Antrian Sederhana Amazon (AmazonSQS)     | <a href="#">AWS::SQS: :Antrian</a>                |
| AWS Step Functions Mesin negara                          | <a href="#">AWS: :Tanpa server:: StateMachine</a> |

## Contoh

Berikut ini adalah contoh komponen yang disempurnakan S3 Bucket:



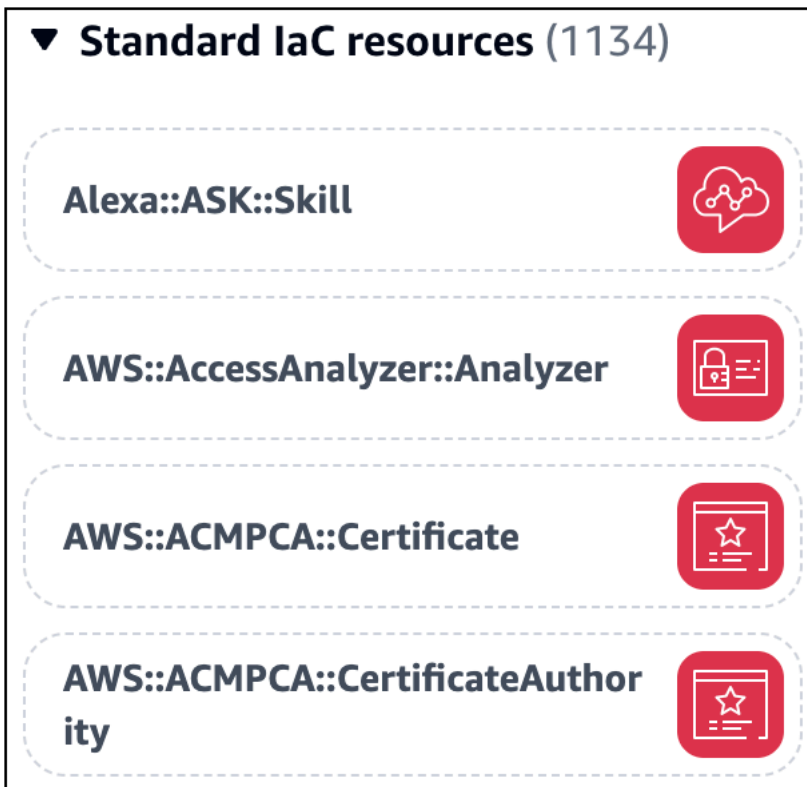
Saat Anda menyeret kartu komponen S3 Bucket ke kanvas dan melihat template Anda, Anda akan melihat dua AWS CloudFormation sumber daya berikut ditambahkan ke template Anda:

- `AWS::S3::Bucket`
- `AWS::S3::BucketPolicy`

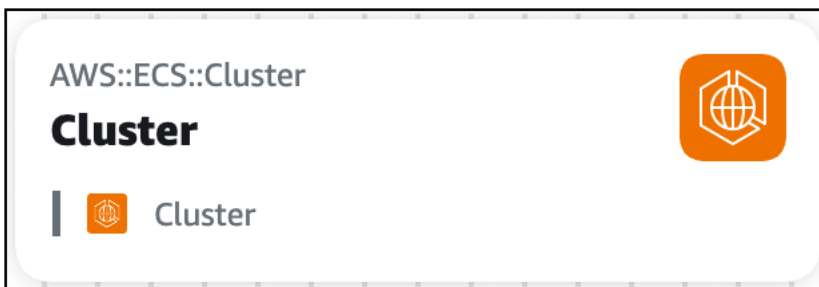
Kartu komponen yang disempurnakan S3 Bucket mewakili dua AWS CloudFormation sumber daya yang keduanya diperlukan untuk bucket Amazon Simple Storage Service (Amazon S3) untuk berinteraksi dengan layanan lain di aplikasi Anda.

## Kartu komponen standar di Infrastructure Composer

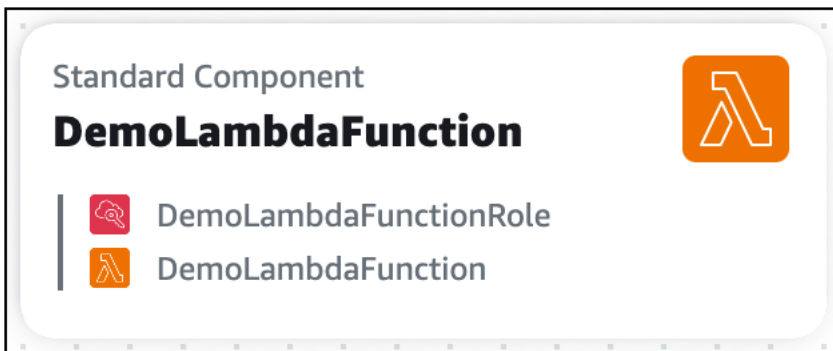
Sebelum kartu komponen standar ditempatkan pada kanvas visual Infrastructure Composer, kartu ini terdaftar sebagai kartu sumber daya Standar (IAC) pada palet Resources di Infrastructure Composer. Kartu sumber daya standar (IAC) mewakili satu AWS CloudFormation sumber daya. Setiap kartu sumber daya IAC standar, setelah ditempatkan pada kanvas visual, menjadi kartu berlabel komponen Standar, dan dapat digabungkan untuk mewakili beberapa AWS CloudFormation sumber daya.



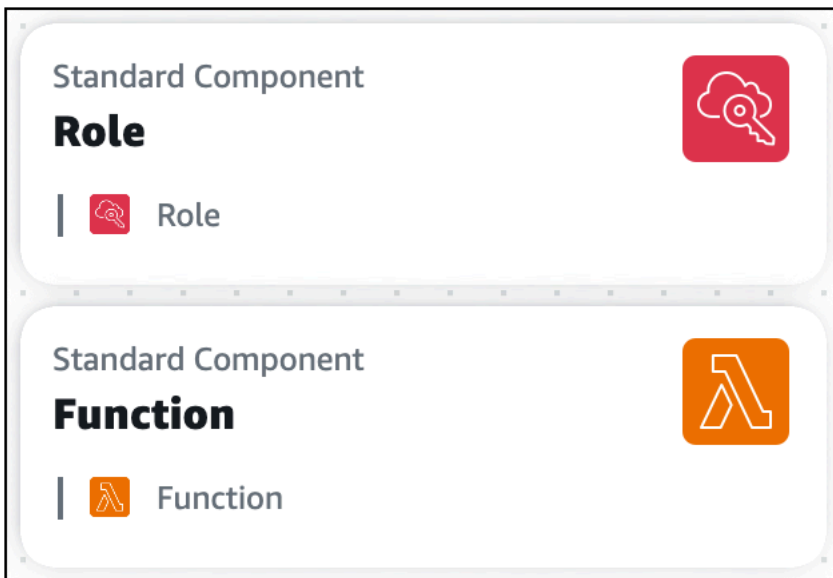
Setiap kartu sumber daya IAC standar dapat diidentifikasi berdasarkan jenis AWS CloudFormation sumber dayanya. Berikut ini adalah contoh kartu sumber daya IAC standar yang mewakili jenis `AWS::ECS::Cluster` AWS CloudFormation sumber daya:



Setiap kartu komponen standar memvisualisasikan AWS CloudFormation sumber daya yang dikandungnya. Berikut ini adalah contoh kartu komponen standar yang mencakup dua sumber daya IAc standar:



Saat Anda mengonfigurasi properti kartu komponen standar Anda, Infrastructure Composer dapat menggabungkan kartu terkait bersama-sama. Misalnya, berikut adalah dua kartu komponen standar:



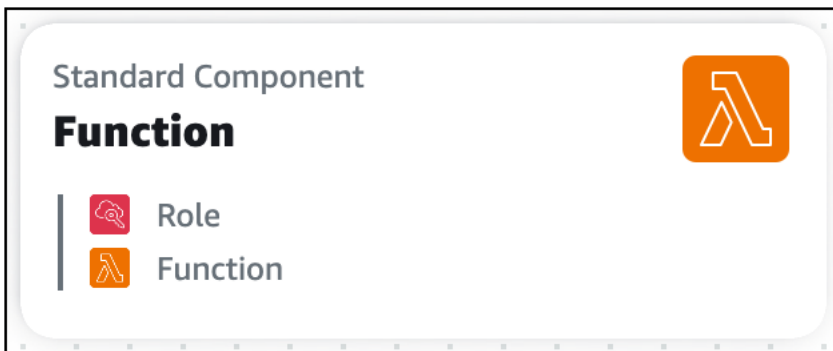
Di panel properti Resource dari kartu komponen standar yang mewakili `AWS::Lambda::Function` sumber daya, kami merujuk peran AWS Identity and Access Management (IAM) dengan ID logisnya:

The screenshot displays the AWS Infrastructure Composer interface. On the left, a grid of standard component cards is shown. Two cards are visible: a red 'Role' card and a blue 'Function' card. The 'Function' card is highlighted with a blue border. On the right, the 'Resource properties' panel is open, showing details for an 'AWS::Lambda::Function' resource. The panel includes an 'Editing' dropdown set to 'Function', a 'Logical ID' field containing 'Function', and a 'Resource configuration' section with a code editor. The code editor contains the following text:

```
Code: {}  
Role: !Ref Role
```

At the bottom right of the 'Resource properties' panel, there is a 'Resource reference' button with an external link icon.

Setelah menyimpan template kami, dua kartu komponen standar bergabung menjadi satu kartu komponen standar.



## Koneksi kartu di Infrastructure Composer

Dalam AWS Infrastructure Composer, koneksi antara dua kartu ditampilkan secara visual oleh garis. Baris-baris ini mewakili hubungan yang digerakkan oleh peristiwa dalam aplikasi Anda.

Topik

- [Koneksi antar kartu](#)
- [Koneksi antara kartu komponen yang disempurnakan](#)
- [Koneksi ke dan dari kartu sumber daya IAC standar](#)

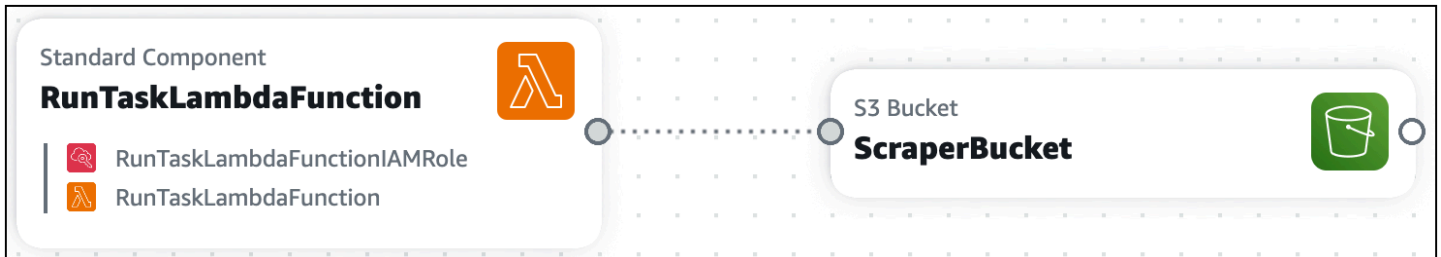
### Koneksi antar kartu

Cara Anda menghubungkan kartu bersama-sama bervariasi tergantung pada jenis kartu. Setiap kartu yang disempurnakan memiliki setidaknya satu port konektor. Untuk menghubungkannya, Anda cukup memilih satu port konektor dan menyeretnya ke port kartu lain, dan Infrastructure Composer akan menghubungkan dua sumber daya atau menampilkan pesan yang menyatakan konfigurasi ini tidak didukung.



Seperti yang terlihat di atas, garis antara kartu komponen yang disempurnakan solid. Sebaliknya, kartu sumber daya IAC standar (juga disebut sebagai kartu komponen standar) tidak memiliki port konektor. Untuk kartu ini, Anda harus menentukan hubungan berbasis peristiwa ini di template

aplikasi Anda, dan Infrastructure Composer akan secara otomatis mendeteksi koneksi mereka dan memvisualisasikannya dengan garis putus-putus di antara kartu Anda.

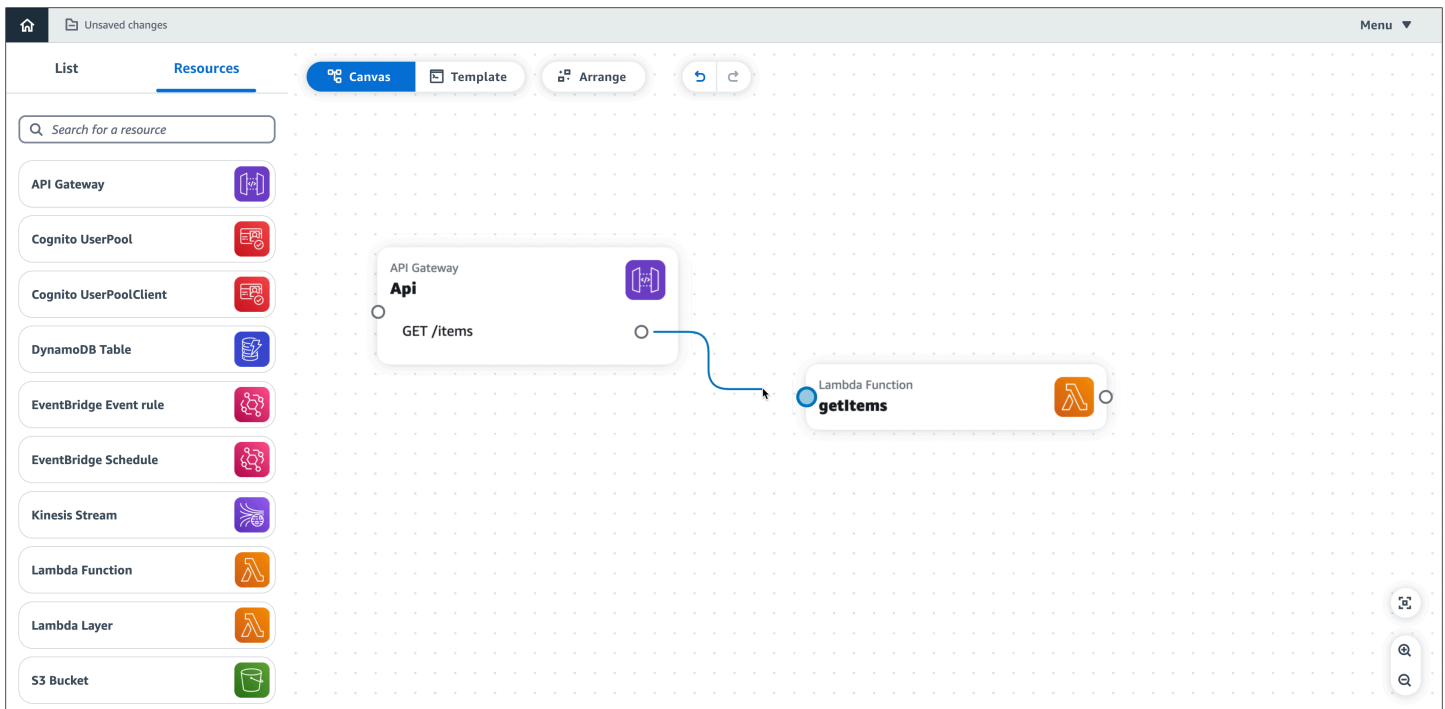


Untuk mempelajari lebih lanjut, lihat bagian di bawah ini.

## Koneksi antara kartu komponen yang disempurnakan

Di Infrastructure Composer, koneksi antara dua kartu komponen yang disempurnakan ditampilkan secara visual oleh garis yang solid. Baris-baris ini mewakili hubungan yang digerakkan oleh peristiwa dalam aplikasi Anda.

Untuk menghubungkan dua kartu, klik pada port dari satu kartu dan seret ke port pada kartu lain.



### Note

Kartu sumber daya IAC standar tidak memiliki port konektor. Untuk kartu ini, Anda harus menentukan hubungan yang digerakkan oleh peristiwa di template aplikasi Anda,



dan Infrastructure Composer akan secara otomatis mendeteksi koneksi mereka dan memvisualisasikannya dengan garis putus-putus di antara kartu Anda.

Untuk informasi selengkapnya, lihat [Connect kartu pada kanvas visual Infrastructure Composer](#).

## Apa yang ditingkatkan penyediaan kartu komponen

Koneksi antara dua kartu, yang ditunjukkan secara visual oleh garis, memberikan yang berikut jika perlu:

- AWS Identity and Access Management (IAM) kebijakan
- Variabel-variabel lingkungan
- Peristiwa

### Kebijakan IAM

Ketika sumber daya memerlukan izin untuk memanggil sumber daya lain, Infrastructure Composer menyediakan kebijakan berbasis sumber daya menggunakan templat kebijakan (). AWS Serverless Application Model AWS SAM

- Untuk mempelajari lebih lanjut tentang IAM izin dan kebijakan, lihat [Ringkasan manajemen akses: Izin dan kebijakan](#) di IAMPanduan Pengguna.
- Untuk mempelajari lebih lanjut tentang templat AWS SAM kebijakan, lihat [templat AWS SAM kebijakan](#) di Panduan AWS Serverless Application Model Pengembang.

### Variabel-variabel lingkungan

Variabel lingkungan adalah nilai sementara yang dapat diubah untuk mempengaruhi perilaku sumber daya Anda. Bila perlu, Infrastructure Composer mendefinisikan kode infrastruktur untuk memanfaatkan variabel lingkungan antar sumber daya.

### Peristiwa

Sumber daya dapat memanggil sumber daya lain melalui berbagai jenis acara. Bila perlu, Infrastructure Composer mendefinisikan kode infrastruktur yang diperlukan untuk sumber daya untuk berinteraksi melalui jenis acara.

## Koneksi ke dan dari kartu sumber daya IAC standar

Semua AWS CloudFormation sumber daya tersedia untuk digunakan sebagai kartu sumber daya IAC standar dari palet Sumber Daya. Saat Anda menyeret kartu sumber daya IAC standar ke kanvas, kartu sumber daya IAC standar menjadi kartu komponen standar, dan ini meminta Infrastructure Composer untuk membuat template awal untuk sumber daya Anda di aplikasi Anda.

Untuk informasi selengkapnya, lihat [Kartu standar di Infrastructure Composer](#).

# Memulai dengan konsol Infrastructure Composer

Gunakan topik di bagian ini untuk mengatur AWS Infrastructure Composer dan mempelajari cara mendesain aplikasi menggunakan kanvas visualnya. Tur dan tutorial di bagian ini ditampilkan di konsol Infrastructure Composer, yang merupakan pengalaman pengguna default. Topik di bagian ini menunjukkan kepada Anda cara menyelesaikan prasyarat untuk menggunakan Infrastructure Composer, menggunakan konsol Infrastructure Composer, memuat dan memodifikasi proyek, dan membangun aplikasi pertama Anda.

Infrastructure Composer juga tersedia dari AWS Toolkit for Visual Studio Code dan dalam mode CloudFormation konsol. Pengalaman antar alat umumnya sama tetapi ada beberapa perbedaan di antara masing-masing alat. Untuk detail tentang penggunaan Infrastructure Composer di masing-masing alat ini, lihat [Di mana Anda dapat menggunakan Infrastructure Composer](#).

## Topik

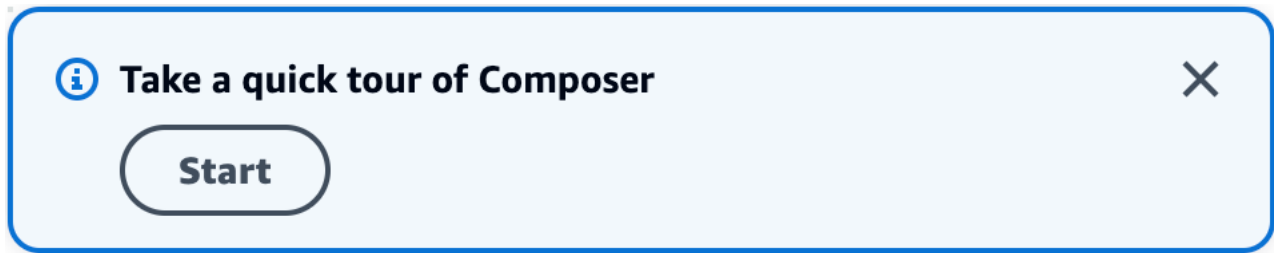
- [Ikuti tur di konsol Infrastructure Composer](#)
- [Memuat dan memodifikasi proyek demo Infrastructure Composer](#)
- [Bangun aplikasi pertama Anda dengan Infrastructure Composer](#)

## Ikuti tur di konsol Infrastructure Composer

Untuk mendapatkan gambaran umum tentang cara AWS Infrastructure Composer kerja, ikuti tur yang dibangun ke dalam konsol Infrastructure Composer. Untuk gambaran umum tentang konsol Infrastructure Composer, lihat [Ikuti tur di konsol Infrastructure Composer](#). Untuk panduan mendalam tentang penggunaan Infrastructure Composer, lihat. [Cara menulis di AWS Infrastructure Composer](#)

Untuk mengikuti tur Komposer Infrastruktur

1. Masuk ke [konsol Infrastructure Composer](#).
2. Pada halaman Beranda, pilih Buka demo.
3. Di sudut kanan atas, di jendela Ikuti tur singkat Komposer, pilih Mulai.



4. Di jendela tur Komposer, lakukan hal berikut:

- Untuk melanjutkan ke langkah berikutnya, pilih Berikutnya.
- Untuk kembali ke langkah sebelumnya, pilih Sebelumnya.
- Pada langkah terakhir, untuk menyelesaikan tur, pilih Akhiri.

Tur ini memberikan gambaran singkat tentang fungsi Komposer Infrastruktur dasar, seperti menggunakan, mengonfigurasi, dan menghubungkan kartu. Untuk informasi lebih lanjut, lihat [Cara menulis di AWS Infrastructure Composer](#).

## Langkah selanjutnya

Untuk memuat dan memodifikasi proyek di Infrastructure Composer, lihat [Memuat dan memodifikasi proyek demo Infrastructure Composer](#).

## Memuat dan memodifikasi proyek demo Infrastructure Composer

Gunakan tutorial ini untuk menjadi akrab dengan antarmuka pengguna Infrastructure Composer dan pelajari cara memuat, memodifikasi, dan menyimpan proyek demo Infrastructure Composer.

Tutorial ini dilakukan di konsol Infrastructure Composer. Setelah selesai, Anda akan siap untuk memulai [Bangun aplikasi pertama Anda dengan Infrastructure Composer](#).

### Topik

- [Langkah 1: Buka demo](#)
- [Langkah 2: Jelajahi kanvas visual Infrastructure Composer](#)
- [Langkah 3: Perluas arsitektur aplikasi Anda](#)
- [Langkah 4: Simpan aplikasi Anda](#)
- [Langkah selanjutnya](#)

## Langkah 1: Buka demo

Mulai gunakan Infrastructure Composer dengan membuat proyek demo.

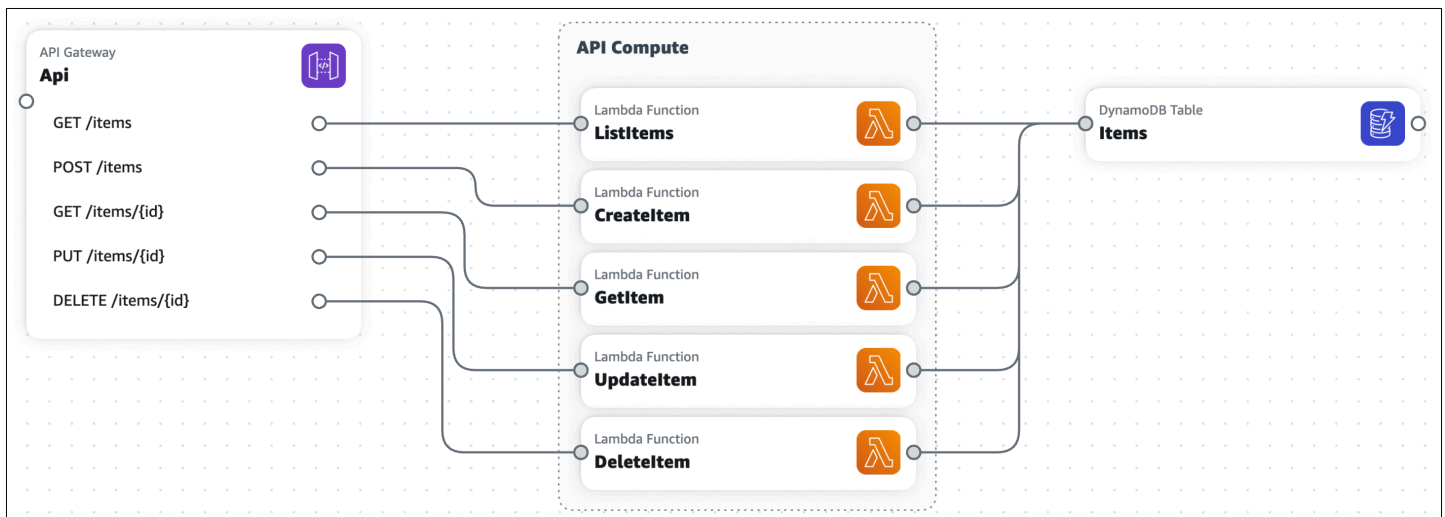
Untuk membuat proyek demo

1. Masuk ke [konsol Infrastructure Composer](#).
2. Pada halaman Beranda, pilih Buka demo.

Aplikasi demo adalah aplikasi dasar buat, baca, hapus, dan perbarui (CRUD) tanpa server yang mencakup:

- Sumber daya Amazon API Gateway dengan lima rute.
- Lima AWS Lambda fungsi.
- Tabel Amazon DynamoDB.

Gambar berikut adalah dari demo:

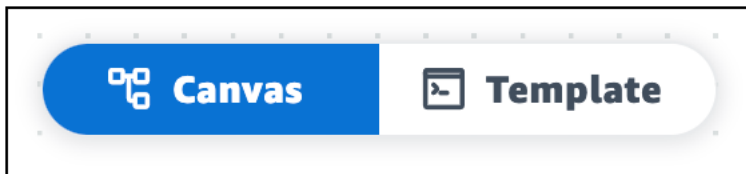


## Langkah 2: Jelajahi kanvas visual Infrastructure Composer

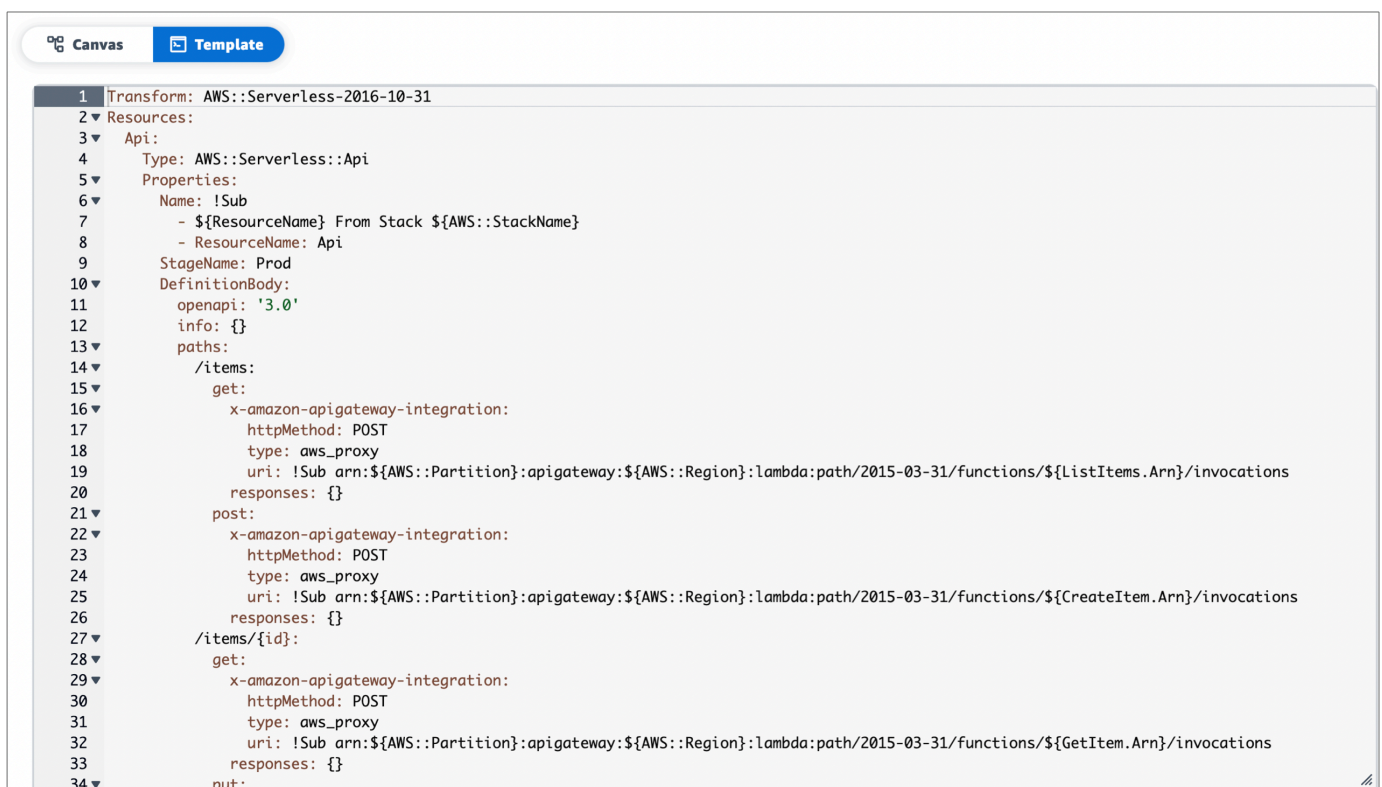
Pelajari fitur kanvas visual untuk membangun proyek demo Infrastructure Composer Anda. Untuk ikhtisar tata letak kanvas visual, lihat [Ikhtisar visual](#).

## Untuk mengeksplorasi fitur kanvas visual

1. Ketika Anda membuka proyek aplikasi baru atau yang sudah ada, Infrastructure Composer memuat tampilan kanvas, seperti yang ditunjukkan di atas area tampilan utama.

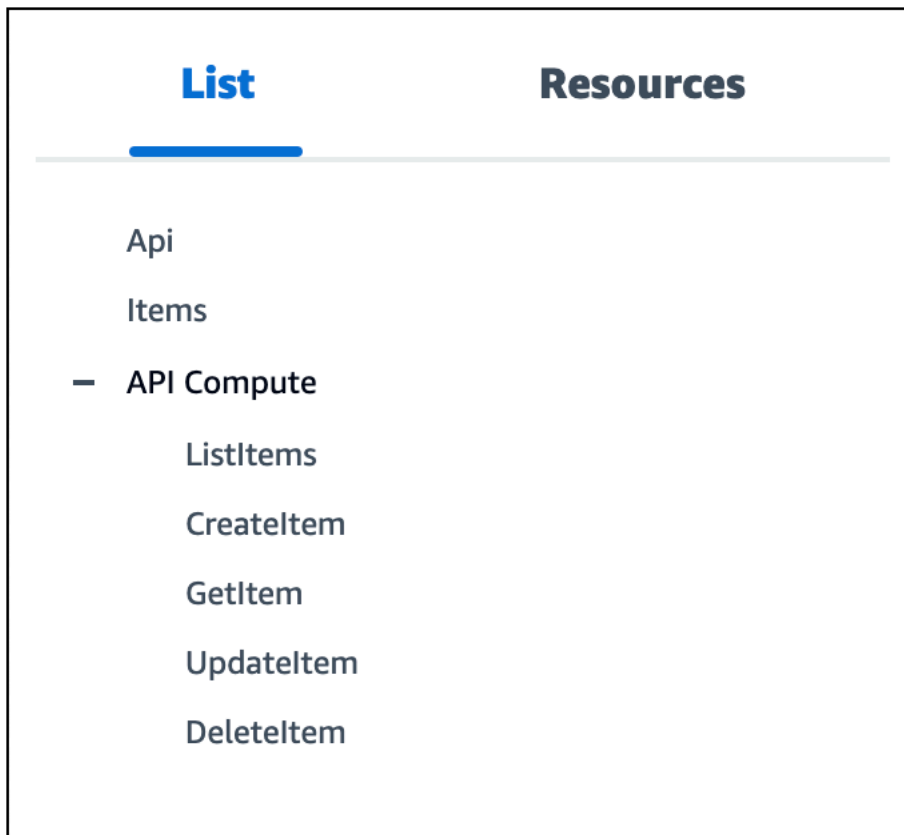


Untuk menampilkan kode infrastruktur aplikasi Anda di area tampilan utama, pilih Template. Misalnya, berikut adalah tampilan template AWS Serverless Application Model (AWS SAM) dari proyek demo Infrastructure Composer.

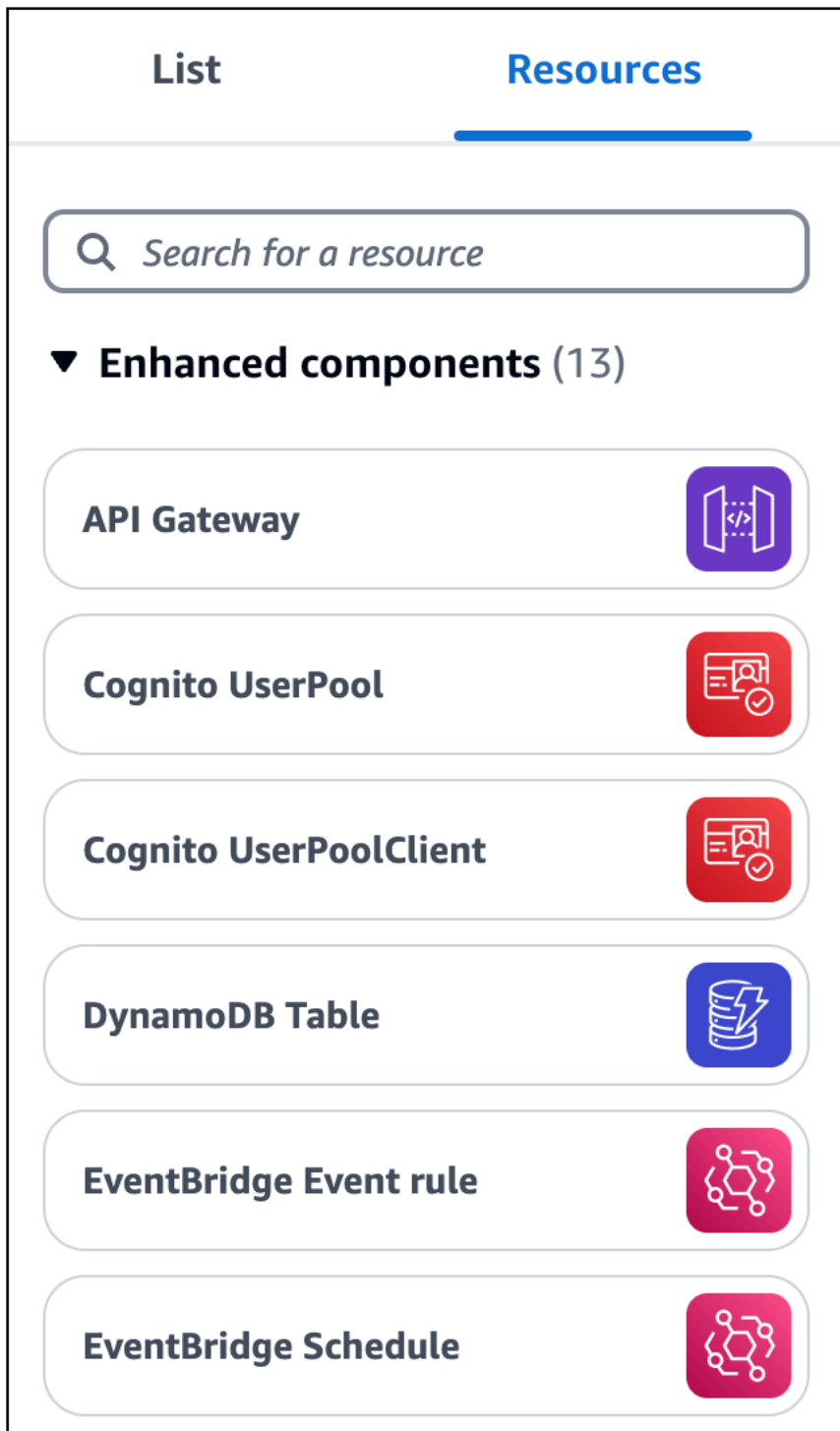
A screenshot of the Infrastructure Composer interface showing the 'Template' view. The interface has a top navigation bar with 'Canvas' and 'Template' buttons. The 'Template' button is selected. Below the navigation bar is a code editor displaying the following JSON code:

```
1 Transform: AWS::Serverless-2016-10-31
2 Resources:
3   Api:
4     Type: AWS::Serverless::Api
5     Properties:
6       Name: !Sub
7         - ${ResourceName} From Stack ${AWS::StackName}
8         - ResourceName: Api
9       StageName: Prod
10    DefinitionBody:
11      openapi: '3.0'
12      info: {}
13      paths:
14        /items:
15          get:
16            x-amazon-apigateway-integration:
17              httpMethod: POST
18              type: aws_proxy
19              uri: !Sub arn:${AWS::Partition}:apigateway:${AWS::Region}:lambda:path/2015-03-31/functions/${ListItems.Arn}/invocations
20              responses: {}
21          post:
22            x-amazon-apigateway-integration:
23              httpMethod: POST
24              type: aws_proxy
25              uri: !Sub arn:${AWS::Partition}:apigateway:${AWS::Region}:lambda:path/2015-03-31/functions/${CreateItem.Arn}/invocations
26              responses: {}
27        /items/{id}:
28          get:
29            x-amazon-apigateway-integration:
30              httpMethod: POST
31              type: aws_proxy
32              uri: !Sub arn:${AWS::Partition}:apigateway:${AWS::Region}:lambda:path/2015-03-31/functions/${GetItem.Arn}/invocations
33              responses: {}
34          put:
```

2. Untuk menampilkan tampilan kanvas aplikasi Anda lagi, pilih Canvas.
3. Untuk menampilkan sumber daya aplikasi Anda yang terorganisir dalam tampilan pohon, pilih Daftar.



4. Untuk menampilkan palet sumber daya, pilih Resources. Palet ini memiliki kartu yang dapat Anda gunakan untuk memperluas arsitektur aplikasi Anda. Anda dapat mencari kartu atau menggulir daftar.



5. Untuk bergerak di sekitar kanvas visual, gunakan gerakan dasar. Untuk informasi selengkapnya, lihat [Tempatkan kartu di kanvas](#).

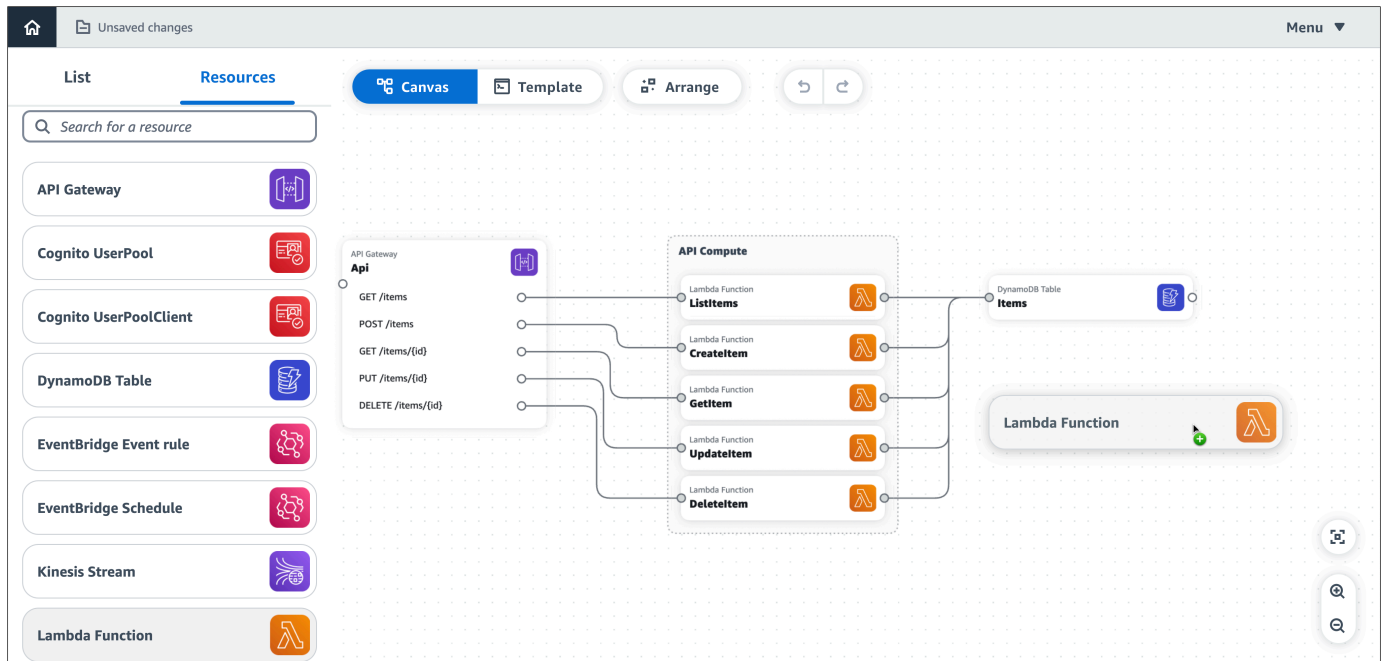


## Langkah 3: Perluas arsitektur aplikasi Anda

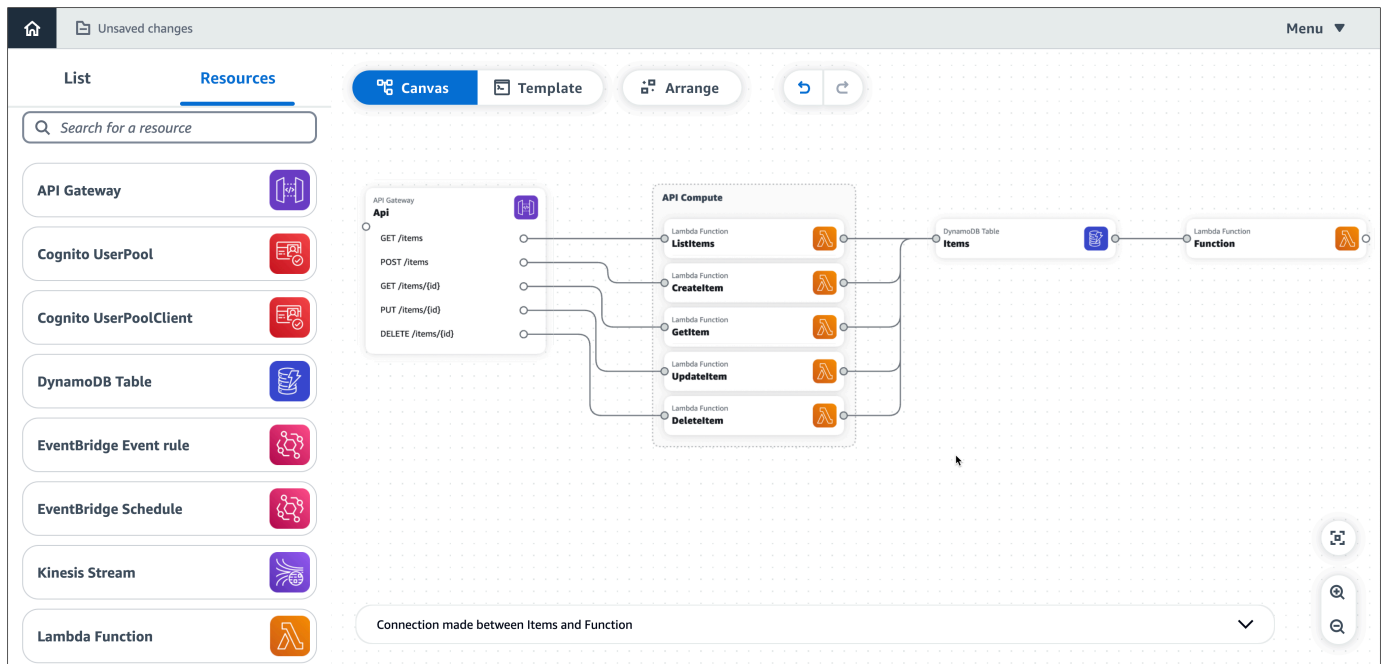
Pada langkah ini, Anda akan memperluas arsitektur aplikasi Anda dengan menambahkan fungsi Lambda ke tabel DynamoDB Anda.

Untuk menambahkan fungsi Lambda ke tabel DynamoDB Anda

1. Dari palet sumber daya (Resources), seret kartu komponen yang disempurnakan Fungsi Lambda ke kanvas, di sebelah kanan kartu DynamoDB Table.



2. Hubungkan tabel DynamoDB ke fungsi Lambda. Untuk menghubungkannya, klik port kanan kartu DynamoDB Table dan seret ke port kiri kartu Fungsi Lambda.
3. Pilih Atur untuk mengatur kartu dalam tampilan kanvas.



4. Konfigurasi fungsi Lambda Anda. Untuk mengkonfigurasinya, lakukan salah satu hal berikut:
- Dalam tampilan kanvas, ubah properti fungsi pada panel Resource properties. Untuk membuka panel, klik dua kali kartu Fungsi Lambda. Atau, pilih kartu, lalu pilih Detail. Untuk informasi selengkapnya tentang properti fungsi Lambda yang dapat dikonfigurasi yang tercantum di panel Properti sumber daya, lihat Panduan [AWS Lambda Pengembang](#).
  - Dalam tampilan template, ubah kode untuk fungsi Anda (`AWS::Serverless::Function`). Infrastructure Composer secara otomatis menyinkronkan perubahan Anda ke kanvas. Untuk informasi selengkapnya tentang sumber daya fungsi dalam AWS SAM template, lihat [AWS::Serverless::Function](#) dalam referensi AWS SAM sumber daya dan properti.

## Langkah 4: Simpan aplikasi Anda

Simpan aplikasi Anda dengan menyimpan template aplikasi Anda secara manual ke mesin lokal Anda atau dengan mengaktifkan sinkronisasi lokal.

Untuk menyimpan template aplikasi Anda secara manual

1. Dari menu, pilih Menyimpan > Simpan file templat.
2. Berikan nama untuk template Anda dan pilih lokasi di mesin lokal Anda untuk menyimpan template Anda. Tekan Simpan.

Untuk petunjuk tentang mengaktifkan sinkronisasi lokal, lihat [Sinkronkan dan simpan proyek Anda secara lokal di konsol Infrastructure Composer](#).

## Langkah selanjutnya

Untuk memulai membangun aplikasi pertama Anda, lihat [Bangun aplikasi pertama Anda dengan Infrastructure Composer](#).

## Bangun aplikasi pertama Anda dengan Infrastructure Composer

Dalam tutorial ini, Anda gunakan AWS Infrastructure Composer untuk membuat, membaca, memperbarui, dan menghapus (CRUD) aplikasi tanpa server yang mengelola pengguna dalam database.

Untuk tutorial ini, kita menggunakan Infrastructure Composer di AWS Management Console. Kami menyarankan Anda menggunakan Google Chrome atau Microsoft Edge, dan jendela browser layar penuh.

### Apakah Anda baru mengenal tanpa server?

Kami merekomendasikan pemahaman dasar tentang topik-topik berikut:

- [Arsitektur berbasis peristiwa](#)
- [Infrastruktur sebagai Kode \(IaC\)](#)
- [Teknologi tanpa server](#)

Untuk mempelajari selengkapnya, lihat [Konsep tanpa server untuk AWS Infrastructure Composer](#).

### Topik

- [Referensi properti sumber daya](#)
- [Langkah 1: Buat proyek Anda](#)
- [Langkah 2: Tambahkan kartu ke kanvas](#)
- [Langkah 3: Konfigurasi API Gateway Anda REST API](#)
- [Langkah 4: Konfigurasi fungsi Lambda Anda](#)

- [Langkah 5: Hubungkan kartu Anda](#)
- [Langkah 6: Atur kanvas](#)
- [Langkah 7: Tambahkan dan hubungkan tabel DynamoDB](#)
- [Langkah 8: Tinjau AWS CloudFormation template Anda](#)
- [Langkah 9: Integrasikan ke dalam alur kerja pengembangan Anda](#)
- [Langkah selanjutnya](#)

## Referensi properti sumber daya

Saat membuat aplikasi Anda, gunakan tabel ini sebagai referensi untuk mengonfigurasi properti Amazon API Gateway dan AWS Lambda sumber daya Anda.

| Metode | Jalur       | Nama fungsi |
|--------|-------------|-------------|
| GET    | /item       | getItem     |
| GET    | /item/ {id} | getItem     |
| PUT    | /item/ {id} | updateItem  |
| POST   | /barang     | addItem     |
| DELETE | /item/ {id} | deleteItem  |

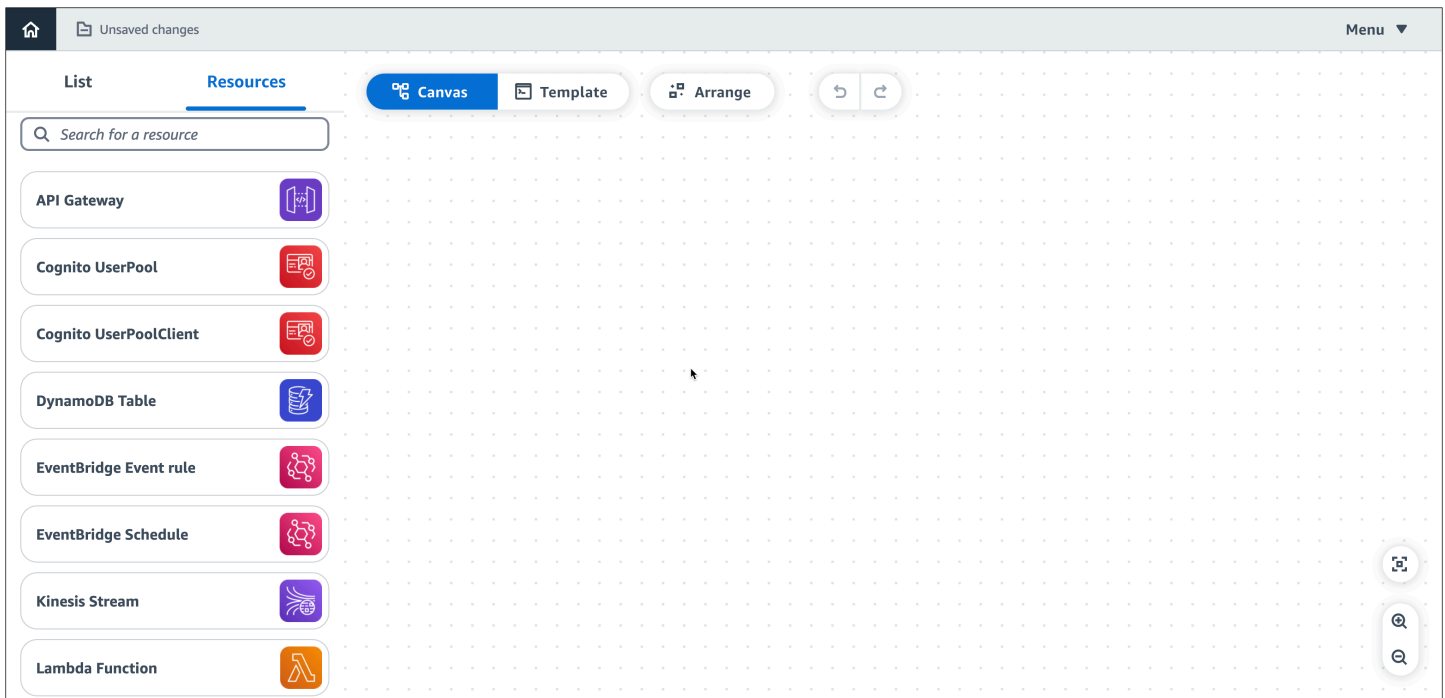
## Langkah 1: Buat proyek Anda

Untuk memulai aplikasi CRUD tanpa server Anda, buat proyek baru di Infrastructure Composer dan aktifkan sinkronisasi lokal.

Untuk membuat proyek kosong baru

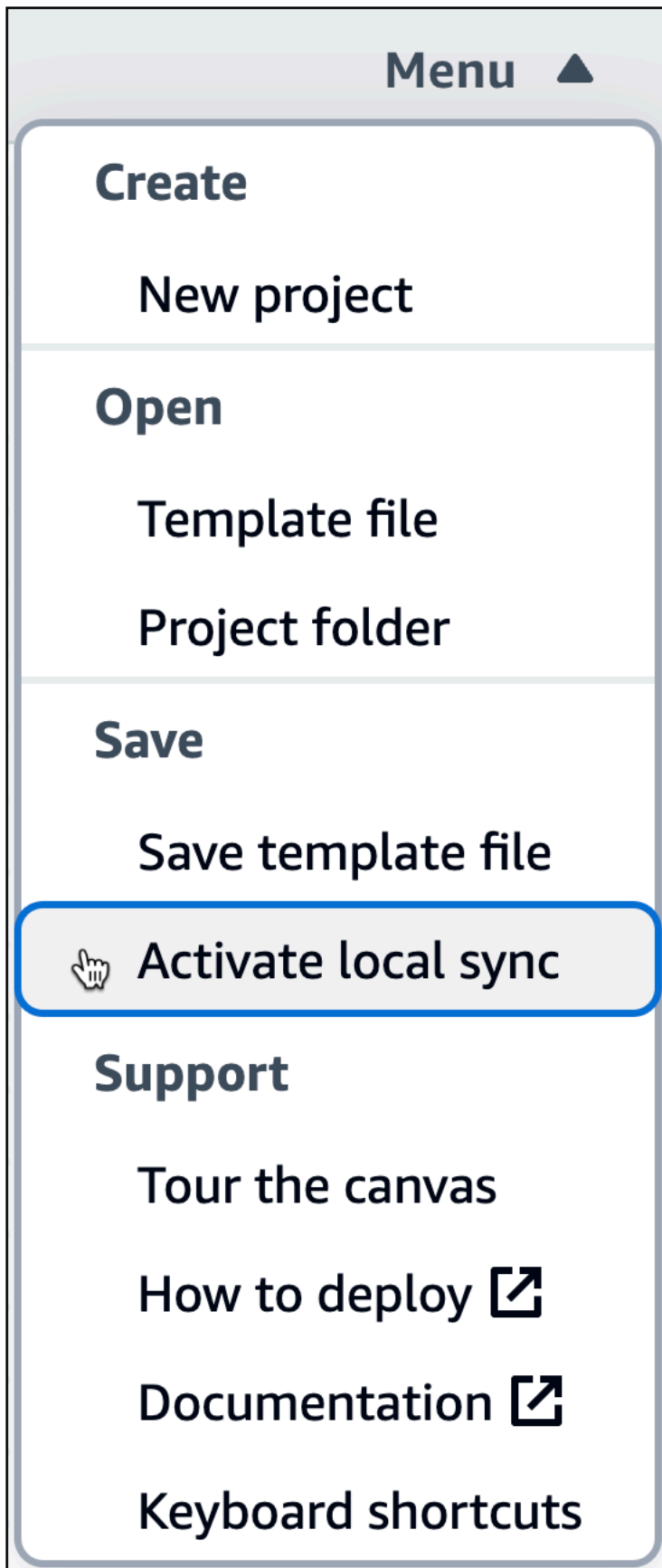
1. Masuk ke [konsol Infrastructure Composer](#).
2. Pada halaman Beranda, pilih Buat proyek.

Seperti yang ditunjukkan pada gambar berikut, Infrastructure Composer membuka kanvas visual dan memuat template aplikasi awal (kosong).




Untuk mengaktifkan sinkronisasi lokal

1. Dari menu Infrastructure Composer, pilih Simpan > Aktifkan sinkronisasi lokal.



2. Untuk lokasi Proyek, tekan Pilih folder dan pilih direktori. Di sinilah Infrastructure Composer akan menyimpan dan menyinkronkan file dan folder template Anda saat Anda mendesain.

Lokasi proyek tidak boleh berisi template aplikasi yang ada.

 Note

Sinkronisasi lokal memerlukan browser yang mendukung Akses Sistem FileAPI. Untuk informasi selengkapnya, lihat [Data Infrastructure Composer mendapatkan akses ke](#).

3. Saat diminta untuk mengizinkan akses, pilih Lihat file.
4. Tekan Aktifkan untuk mengaktifkan sinkronisasi lokal. Saat diminta untuk menyimpan perubahan, pilih Simpan perubahan.

Saat diaktifkan, indikator Autosave akan ditampilkan di area kiri atas kanvas Anda.

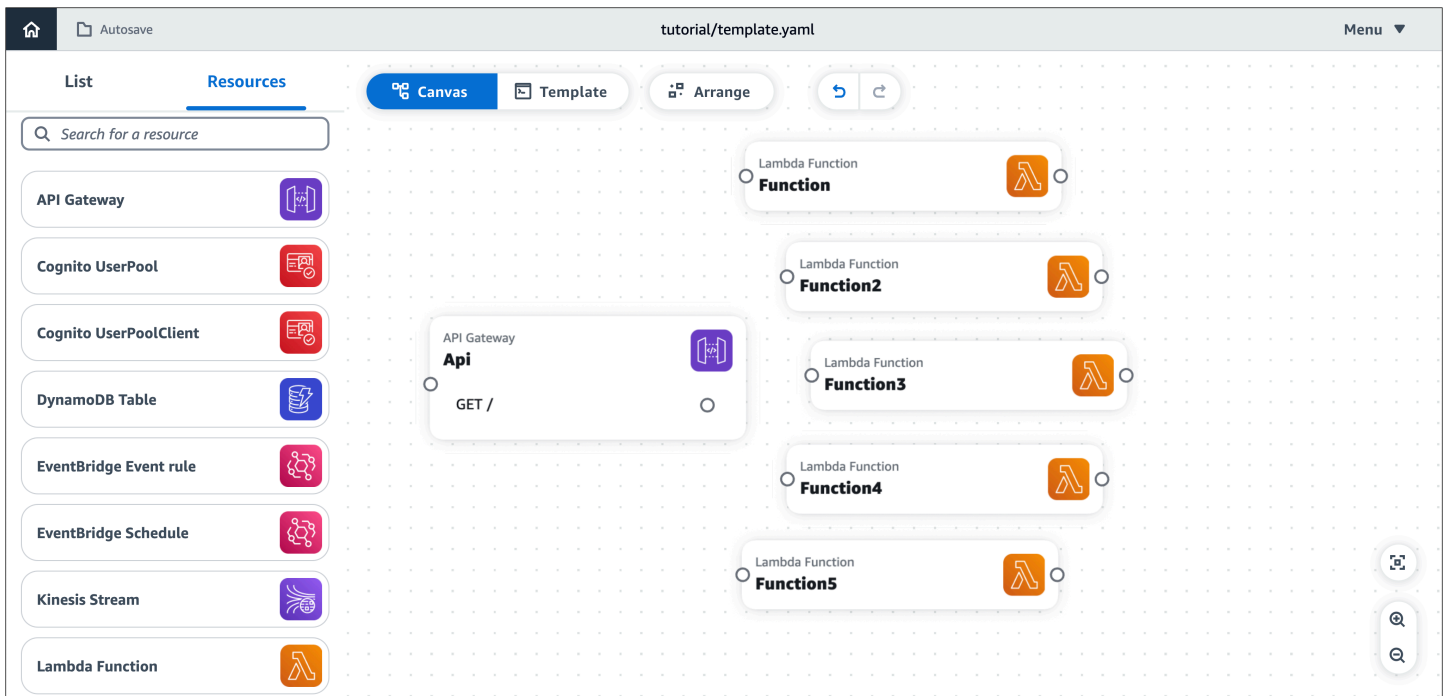
## Langkah 2: Tambahkan kartu ke kanvas

Mulai merancang arsitektur aplikasi Anda menggunakan kartu komponen yang disempurnakan, dimulai dengan API Gateway REST API dan lima fungsi Lambda.

Untuk menambahkan kartu API Gateway dan Lambda ke kanvas

Dari palet Resources, di bawah bagian Enhanced components, lakukan hal berikut:

1. Seret kartu APIGateway ke kanvas.
2. Seret kartu Fungsi Lambda ke kanvas. Ulangi sampai Anda menambahkan lima kartu Fungsi Lambda ke kanvas.



## Langkah 3: Konfigurasi API Gateway Anda REST API

Selanjutnya, tambahkan lima rute dalam kartu API Gateway Anda.

Untuk menambahkan rute ke kartu API Gateway

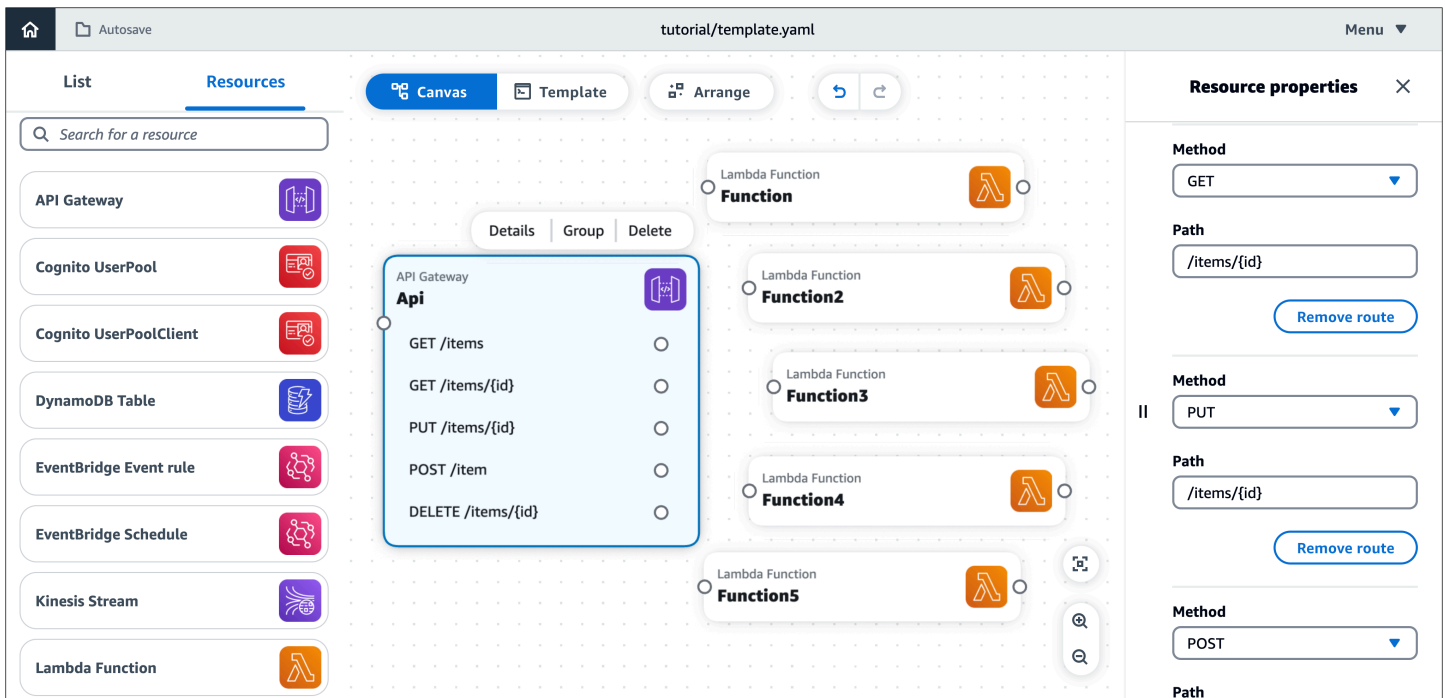
1. Buka panel Resource properties untuk kartu APIGateway. Untuk membuka panel, klik dua kali kartu. Atau, pilih kartu, lalu pilih Detail.
2. Di panel Resource properties, di bawah Routes, lakukan hal berikut:

### Note

Untuk masing-masing rute berikut, gunakan HTTP metode dan nilai jalur yang ditentukan dalam [tabel referensi properti sumber daya](#).

- a. Untuk Metode, pilih HTTP metode yang ditentukan. Misalnya, GET.
  - b. Untuk Path, masukkan jalur yang ditentukan. Misalnya, **/items**.
  - c. Pilih Tambahkan rute.
  - d. Ulangi langkah sebelumnya hingga Anda menambahkan kelima rute yang ditentukan.
3. Pilih Simpan.





## Langkah 4: Konfigurasi fungsi Lambda Anda

Beri nama masing-masing dari lima fungsi Lambda seperti yang ditentukan dalam tabel [referensi properti sumber daya](#).

Untuk memberi nama fungsi Lambda

1. Buka panel Resource properties dari kartu Fungsi Lambda. Untuk membuka panel, klik dua kali kartu. Atau, pilih kartu, lalu pilih Detail.
2. Di panel Resource properties, untuk Logical ID, masukkan nama fungsi tertentu. Misalnya, **getItems**.
3. Pilih Simpan.
4. Ulangi langkah sebelumnya sampai Anda menamai kelima fungsi.

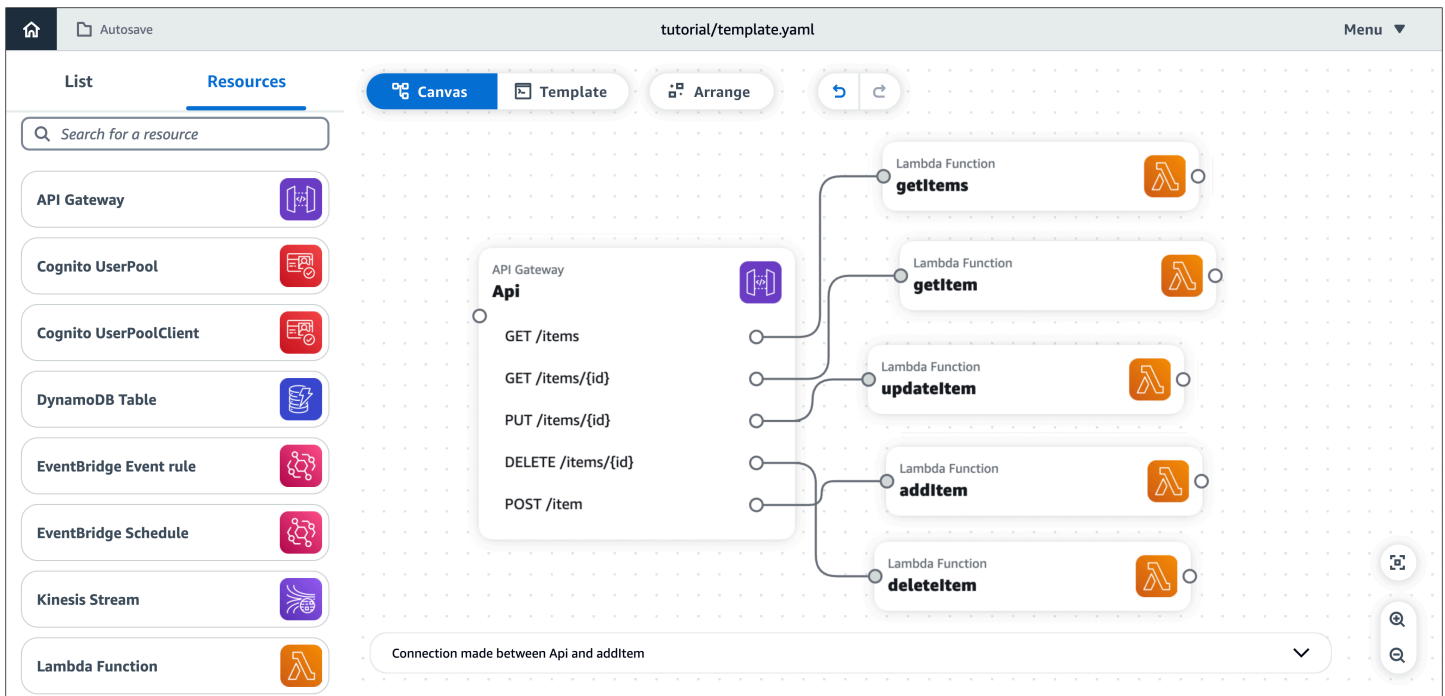
The screenshot displays the AWS Infrastructure Composer interface for a template named 'tutorial/template.yaml'. On the left, a 'Resources' panel lists various AWS services. The central canvas shows an 'API Gateway' resource named 'Api' with five routes. Each route is connected to a corresponding 'Lambda Function' resource: 'getItems' for GET /items, 'getItem' for GET /items/{id}, 'updateItem' for PUT /items/{id}, 'addItem' for DELETE /items/{id}, and 'deleteItem' for POST /item. The 'deleteItem' resource is highlighted. On the right, the 'Resource properties' panel for the selected Lambda Function shows its logical ID, package type (Zip), and source path (src/Function5).

## Langkah 5: Hubungkan kartu Anda

Hubungkan setiap rute pada kartu APIGateway Anda ke kartu Fungsi Lambda terkait, seperti yang ditentukan dalam tabel [referensi properti sumber daya](#).

Untuk menghubungkan kartu Anda

1. Klik port kanan pada kartu APIGateway dan seret ke port kiri kartu Fungsi Lambda yang ditentukan. Misalnya, klik port GET/items dan seret ke port kiri. getItems
2. Ulangi langkah sebelumnya hingga Anda menghubungkan kelima rute pada kartu APIGateway ke kartu Fungsi Lambda yang sesuai.



## Langkah 6: Atur kanvas

Atur kanvas visual dengan mengelompokkan fungsi Lambda Anda dan mengatur semua kartu.

Untuk mengelompokkan fungsi Anda

1. Tekan dan tahan Shift, lalu pilih setiap kartu Fungsi Lambda di kanvas.
2. Pilih Grup.

Untuk memberi nama grup Anda

1. Klik dua kali bagian atas grup, di dekat nama grup (Grup).

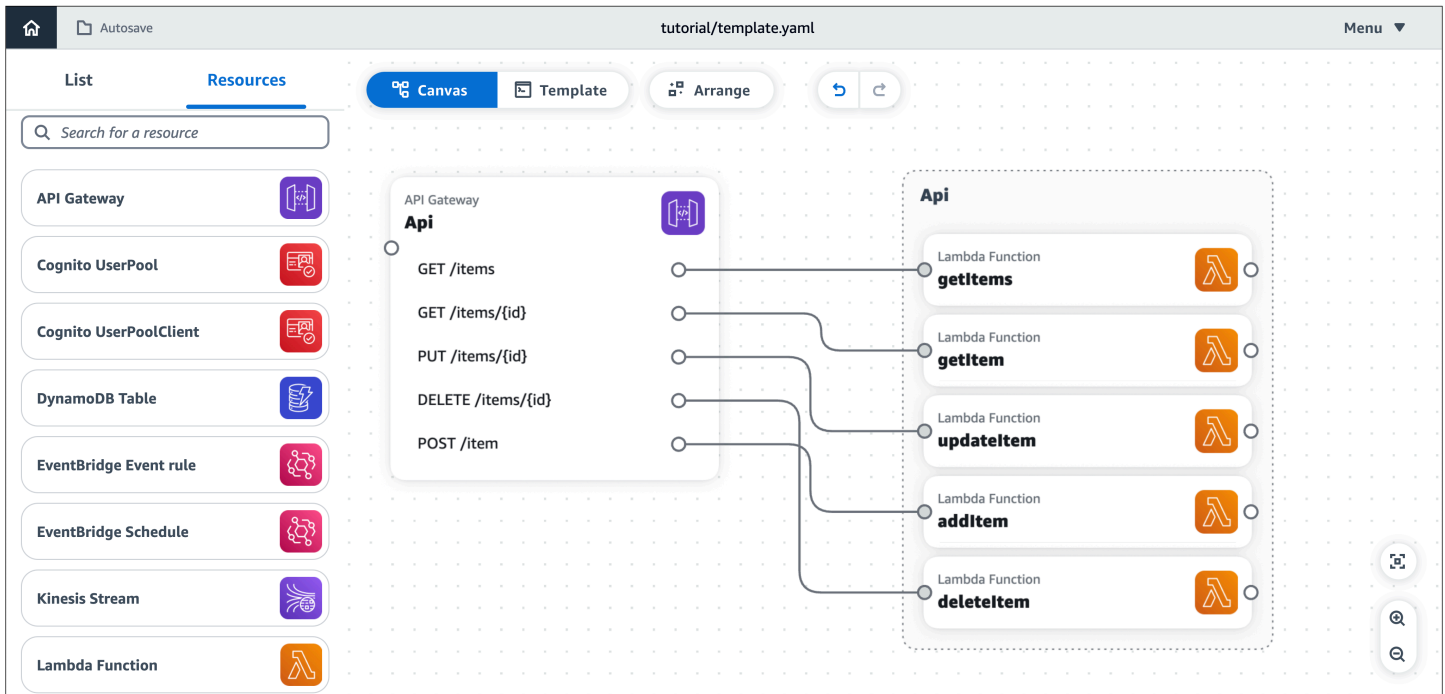
Panel properti Grup terbuka.

2. Pada panel properti Grup, untuk nama Grup, masukkan **API**.
3. Pilih Simpan.

Untuk mengatur kartu Anda

Di kanvas, di atas area tampilan utama, pilih Atur.

Infrastructure Composer mengatur dan menyelaraskan semua kartu pada kanvas visual, termasuk grup baru Anda (API), seperti yang ditunjukkan di sini:

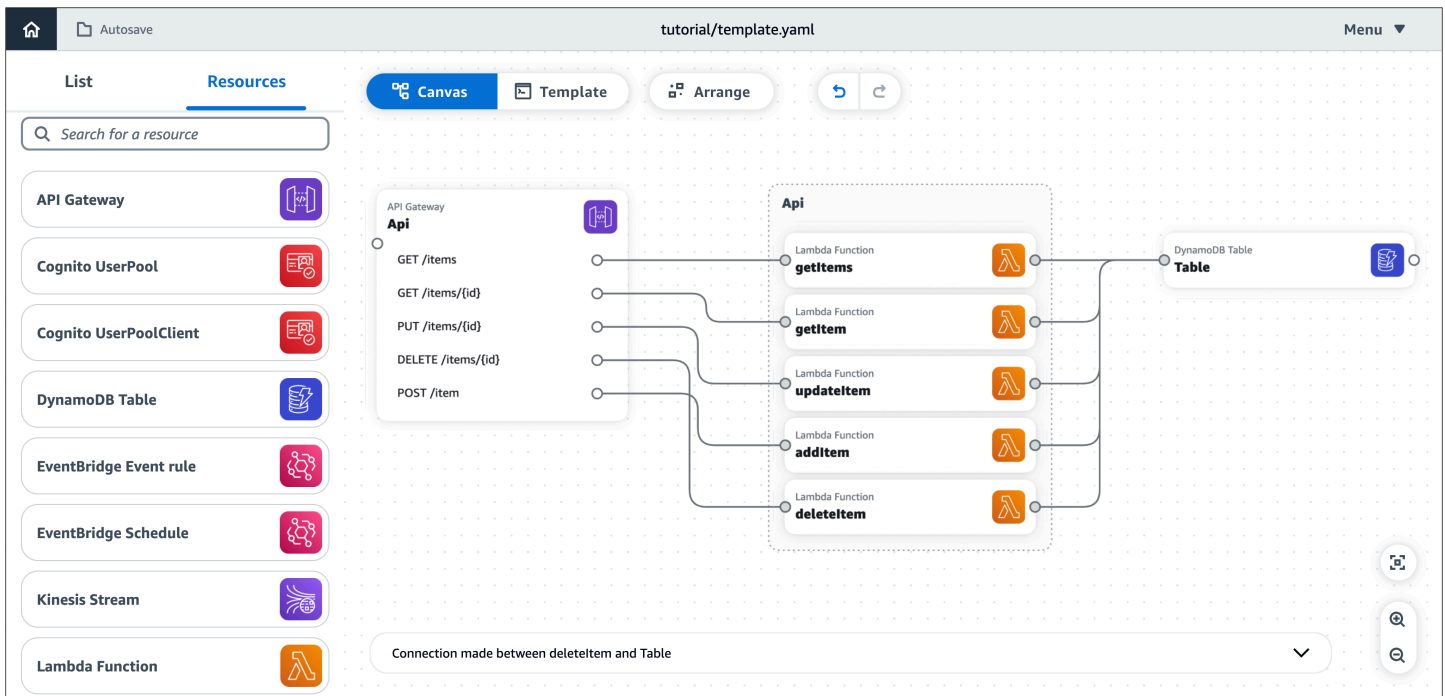


## Langkah 7: Tambahkan dan hubungkan tabel DynamoDB

Sekarang, tambahkan tabel DynamoDB ke arsitektur aplikasi Anda dan hubungkan ke fungsi Lambda Anda.

Untuk menambah dan menghubungkan tabel DynamoDB

1. Dari palet sumber daya (Resources), di bawah bagian Enhanced components, seret kartu DynamoDB Table ke kanvas.
2. Klik port kanan pada kartu Fungsi Lambda dan seret ke port kiri kartu DynamoDB Table.
3. Ulangi langkah sebelumnya sampai Anda menghubungkan kelima kartu Fungsi Lambda ke kartu DynamoDB Table.
4. (Opsional) Untuk mengatur ulang dan menyelaraskan kembali kartu di kanvas, pilih Atur.

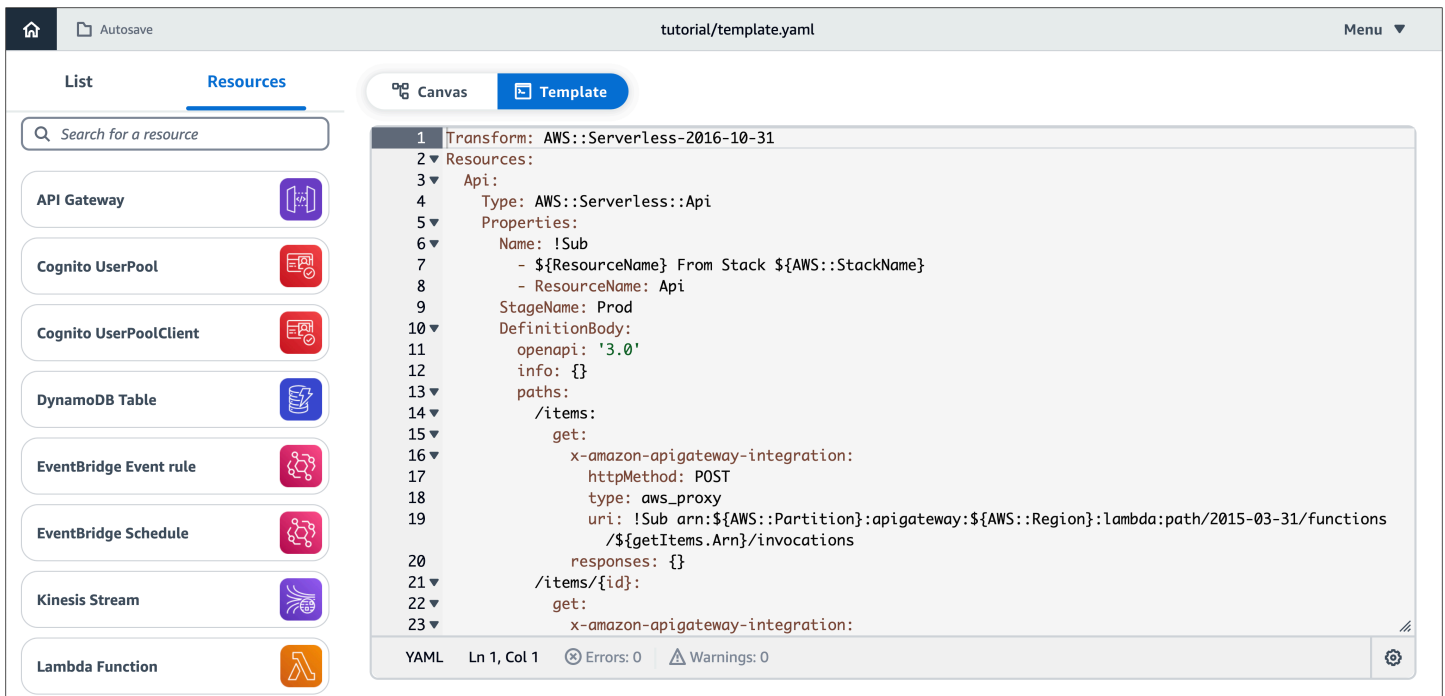


## Langkah 8: Tinjau AWS CloudFormation template Anda

Selamat! Anda telah berhasil merancang aplikasi tanpa server yang siap digunakan. Terakhir, pilih Template untuk meninjau AWS CloudFormation template yang secara otomatis dihasilkan oleh Infrastructure Composer untuk Anda.

Dalam template, Infrastructure Composer telah mendefinisikan yang berikut:

- **TransformDeklarasi**, yang menentukan template sebagai AWS Serverless Application Model (AWS SAM) template. Untuk informasi selengkapnya, lihat [anatomi AWS SAM templat](#) di Panduan AWS Serverless Application Model Pengembang.
- **AWS::Serverless::Api** sumber daya, yang menentukan API Gateway Anda REST API dengan lima rute.
- Lima **AWS::Serverless::Function** sumber daya, yang menentukan konfigurasi fungsi Lambda Anda, termasuk variabel lingkungan dan kebijakan izinnya.
- **AWS::DynamoDB::Table** sumber daya, yang menentukan tabel DynamoDB Anda dan propertinya.
- **MetadataBagian**, yang berisi informasi tentang grup sumber daya Anda (API). Untuk informasi selengkapnya tentang bagian ini, lihat [Metadata](#) di AWS CloudFormation Panduan Pengguna.



## Langkah 9: Integrasikan ke dalam alur kerja pengembangan Anda

Gunakan file template dan direktori proyek yang dibuat oleh Infrastructure Composer untuk pengujian dan penerapan lebih lanjut.

- Dengan sinkronisasi lokal, Anda dapat menghubungkan Infrastructure Composer ke mesin lokal Anda untuk mempercepat pengembangan. IDE Untuk mempelajari selengkapnya, lihat [Connect konsol Infrastructure Composer dengan lokal Anda IDE](#).
- Dengan sinkronisasi lokal, Anda dapat menggunakan AWS Serverless Application Model Command Line Interface (AWS SAM CLI) pada mesin lokal Anda untuk menguji dan menyebarkan aplikasi Anda. Untuk mempelajari selengkapnya, lihat [Menerapkan aplikasi tanpa server Infrastructure Composer Anda ke Cloud AWS](#).

## Langkah selanjutnya

Anda sekarang siap untuk membangun aplikasi Anda sendiri dengan Infrastructure Composer. Untuk detail mendalam tentang penggunaan Infrastructure Composer, lihat. [Cara menulis di AWS Infrastructure Composer](#) Ketika Anda siap untuk menerapkan aplikasi Anda, lihat. [Menerapkan aplikasi tanpa server Infrastructure Composer Anda ke Cloud AWS](#)

# Di mana Anda dapat menggunakan Infrastructure Composer

Anda dapat menggunakan Infrastructure Composer dari konsolnya, dari AWS Toolkit for Visual Studio Code, dan di Infrastructure Composer dalam mode CloudFormation konsol. Sementara masing-masing bervariasi untuk kasus penggunaan yang sedikit berbeda, secara keseluruhan mereka adalah pengalaman yang serupa. Bagian ini memberikan detail dari setiap pengalaman.

Topiknya [Menggunakan AWS Infrastructure Composer konsol](#) adalah ikhtisar komprehensif tentang pengalaman konsol default. Topik [CloudFormation modus konsol](#) ini memberikan rincian tentang versi Infrastructure Composer yang terintegrasi dengan alur kerja AWS CloudFormation stack. [AWS Toolkit for Visual Studio Code](#) memberikan informasi tentang mengakses dan menggunakan Infrastructure Composer di VS Code.

Topik

- [Menggunakan AWS Infrastructure Composer konsol](#)
- [Menggunakan Infrastructure Composer dalam mode CloudFormation konsol](#)
- [Menggunakan Infrastructure Composer dari AWS Toolkit for Visual Studio Code](#)

## Menggunakan AWS Infrastructure Composer konsol

Bagian ini memberikan rincian tentang mengakses dan menggunakan AWS Infrastructure Composer dari konsol Infrastructure Composer. Ini adalah pengalaman default untuk Infrastructure Composer dan merupakan cara yang baik untuk menjadi akrab dengan Infrastructure Composer. Anda juga dapat mengintegrasikan konsol Infrastructure Composer dengan lokal IDE Anda. Untuk detailnya, lihat [Connect konsol Infrastructure Composer dengan lokal Anda IDE](#).

Anda juga dapat [mengakses Infrastructure Composer dari AWS Toolkit di VS Code](#), dan Anda dapat menggunakan [mode Infrastructure Composer yang dirancang khusus untuk digunakan](#). AWS CloudFormation

Untuk dokumentasi umum tentang penggunaan Infrastructure Composer, lihat [Cara menulis](#).

Topik

- [AWS Infrastructure Composer ikhtisar visual konsol](#)
- [Kelola proyek Anda dari konsol Infrastructure Composer](#)

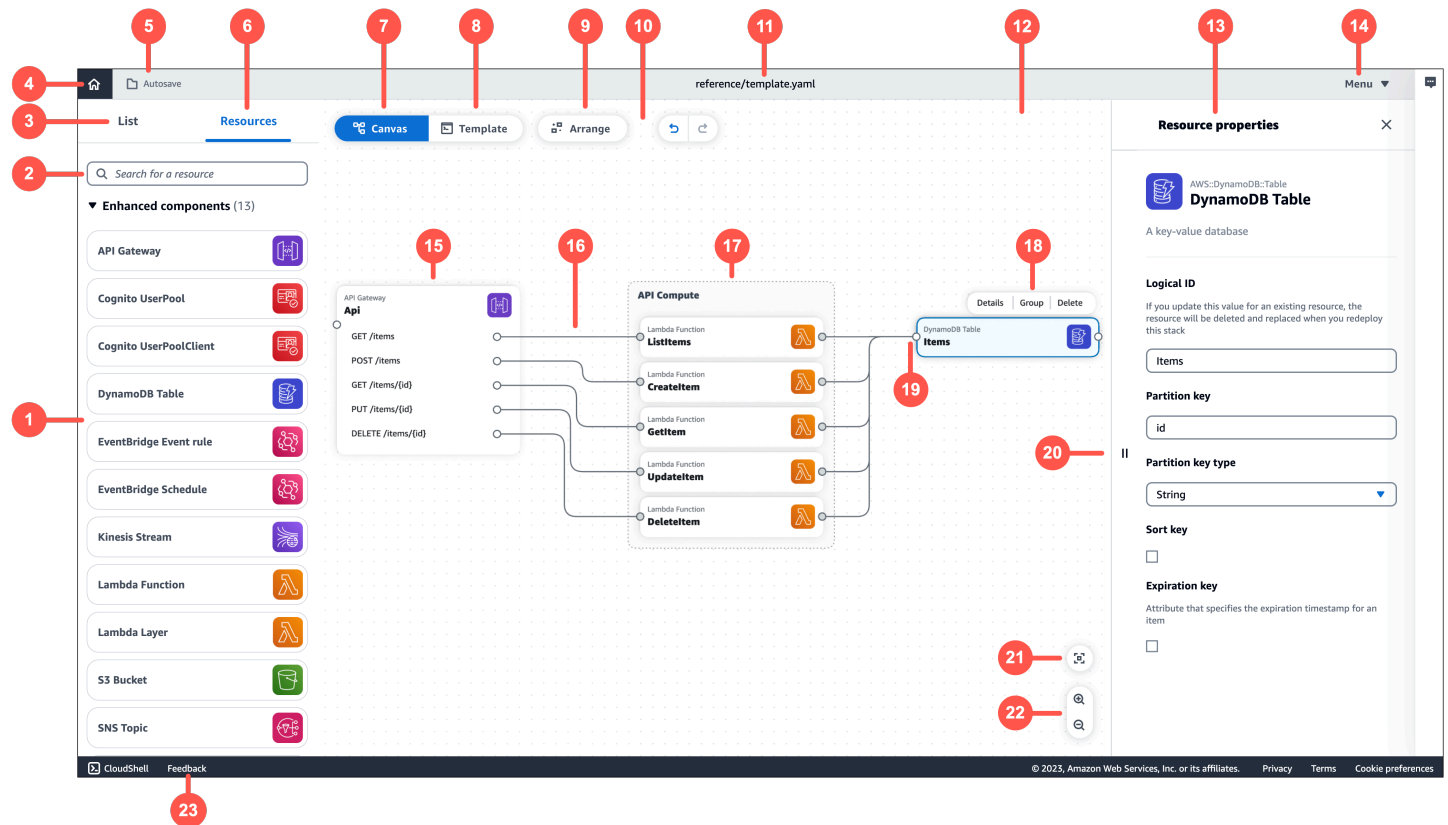




4. Buat proyek — Buat atau muat proyek.
5. Mulai membangun — Tautan cepat untuk mulai membangun aplikasi.
6. Umpan balik - Buka di sini untuk mengirimkan umpan balik.

## Desainer visual dan kanvas visual

Gambar berikut adalah desainer visual dan kanvas visual Infrastructure Composer:



1. Palet sumber daya - Menampilkan kartu yang dapat Anda desain dengan.
2. Bilah pencarian sumber daya — Cari kartu yang dapat Anda tambahkan ke kanvas.
3. Daftar - Menampilkan tampilan pohon dari sumber daya aplikasi Anda.
4. Home - Pilih di sini untuk pergi ke homepage Infrastructure Composer.
5. Simpan status - Menunjukkan apakah perubahan Infrastructure Composer disimpan ke mesin lokal Anda. Negara termasuk:
  - Simpan Otomatis - Sinkronisasi lokal diaktifkan dan proyek Anda disinkronkan dan disimpan secara otomatis.
  - Perubahan disimpan - Template aplikasi Anda disimpan ke mesin lokal Anda.

- Perubahan yang belum disimpan — Template aplikasi Anda memiliki perubahan yang tidak disimpan ke mesin lokal Anda.
6. Sumber Daya - Menampilkan palet sumber daya.
  7. Canvas - Menampilkan tampilan kanvas aplikasi Anda di area tampilan utama.
  8. Template - Menampilkan tampilan template aplikasi Anda di area tampilan utama.
  9. Atur - Mengatur arsitektur aplikasi Anda di kanvas.
  10. Undo dan redo — Lakukan tindakan undo dan redo saat didukung.
  11. Nama templat - Menunjukkan nama template yang Anda rancang.
  12. Area tampilan utama - Menampilkan kanvas atau template berdasarkan pilihan Anda.
  13. Panel properti sumber daya - Menampilkan properti yang relevan untuk kartu yang telah dipilih di kanvas. Panel ini dinamis. Properti yang ditampilkan akan berubah saat Anda mengkonfigurasi kartu Anda.
  14. Menu - Menyediakan opsi umum seperti berikut ini:
    - Buat proyek
    - Buka file template atau proyek
    - Menyimpan file template
    - [Aktifkan sinkronisasi lokal](#)
    - [Ekspor kanvas](#)
    - Dapatkan dukungan
    - Pintasan keyboard
  15. Kartu — Menampilkan tampilan kartu Anda di kanvas.
  16. Line — Merupakan koneksi antar kartu.
  17. Grup — Kelompokkan kartu yang dipilih bersama untuk organisasi visual.
  18. Tindakan kartu — Menyediakan tindakan yang dapat Anda ambil pada kartu Anda.
    - a. Detail - Memunculkan panel properti sumber daya.
    - b. Grup — Kelompokkan kartu yang dipilih bersama.
    - c. Hapus — Menghapus kartu dari kanvas Anda.
  19. Port — Koneksi menunjuk ke kartu lain.
  20. Bidang properti sumber daya — Satu set bidang properti yang dikuratori untuk dikonfigurasi untuk kartu Anda.
  21. Re-center — Pusatkan kembali diagram aplikasi Anda pada kanvas visual.

22. Zoom - Zoom in dan out pada kanvas Anda.

23. Umpan balik - Buka di sini untuk mengirimkan umpan balik.

## Kelola proyek Anda dari konsol Infrastructure Composer

Topik ini memberikan panduan tentang tugas-tugas dasar yang Anda lakukan untuk mengelola proyek Anda dari konsol Infrastructure Composer. Ini termasuk tugas-tugas umum seperti membuat proyek baru, menyimpan proyek, dan mengimpor proyek atau template. Anda juga dapat memuat proyek yang ada jika Anda mengaktifkan [mode sinkronisasi lokal](#). Setelah mengaktifkan mode sinkronisasi lokal, Anda dapat melakukan hal berikut:

- Buat proyek baru yang terdiri dari template awal dan struktur folder.
- Muat proyek yang ada dengan memilih folder induk yang berisi templat dan file proyek Anda.
- Gunakan Infrastructure Composer untuk mengelola template dan folder

Dengan mode sinkronisasi lokal, Infrastructure Composer secara otomatis menyimpan perubahan template dan folder proyek Anda ke mesin lokal Anda. Jika browser Anda tidak mendukung mode sinkronisasi lokal, atau jika Anda lebih suka menggunakan Infrastructure Composer tanpa mode sinkronisasi lokal diaktifkan, Anda dapat membuat template baru atau memuat template yang sudah ada. Untuk menyimpan perubahan, Anda harus mengeksport template ke mesin lokal Anda.

### Note

Infrastructure Composer mendukung aplikasi yang terdiri dari berikut ini:

- Sebuah AWS CloudFormation atau AWS Serverless Application Model template yang mendefinisikan kode infrastruktur Anda.
- Struktur folder yang mengatur file proyek Anda, seperti kode fungsi Lambda, file konfigurasi, dan folder build.

### Topik

- [Buat proyek baru di konsol Infrastructure Composer](#)
- [Impor folder proyek yang ada di konsol Infrastructure Composer](#)
- [Impor template proyek yang ada di konsol Infrastructure Composer](#)

- [Simpan template proyek yang ada di konsol Infrastructure Composer](#)

## Buat proyek baru di konsol Infrastructure Composer

Saat Anda membuat proyek baru, Infrastructure Composer menghasilkan template awal. Saat Anda mendesain aplikasi Anda di kanvas, template Anda dimodifikasi. Untuk menyimpan pekerjaan Anda, Anda harus mengekspor template Anda atau mengaktifkan mode sinkronisasi lokal.

Untuk membuat proyek baru

1. Masuk ke [konsol Infrastructure Composer](#).
2. Pada halaman Beranda, pilih Buat proyek.

### Note

Anda juga dapat memuat yang sudah ada di Infrastructure Composer, tetapi Anda harus terlebih dahulu [mengaktifkan mode sinkronisasi lokal](#). Setelah diaktifkan, lihat [Muat proyek Infrastructure Composer yang ada dengan sinkronisasi lokal diaktifkan](#) untuk memuat proyek yang ada.

## Impor folder proyek yang ada di konsol Infrastructure Composer

Menggunakan mode sinkronisasi lokal, Anda dapat mengimpor folder induk dari proyek yang ada. Jika proyek Anda berisi beberapa template, Anda dapat memilih template yang akan dimuat.

Untuk mengimpor proyek yang ada dari halaman Beranda

1. Masuk ke [konsol Infrastructure Composer](#).
2. Pada halaman Beranda, pilih Muat CloudFormation templat.
3. Untuk lokasi Proyek, pilih Pilih folder. Pilih folder induk proyek Anda dan pilih Pilih.


### Note

Jika Anda tidak menerima prompt ini, browser Anda mungkin tidak mendukung Akses Sistem FileAPI, yang diperlukan untuk mode sinkronisasi lokal. Untuk informasi selengkapnya, lihat [Izinkan akses halaman web ke file lokal di Infrastructure Composer](#).

4. Saat diminta oleh browser Anda, pilih Lihat file.
5. Untuk file Template, pilih template Anda dari daftar dropdown. Jika proyek Anda berisi satu template, Infrastructure Composer secara otomatis memilihnya untuk Anda.
6. Pilih Buat.

Untuk mengimpor proyek yang ada dari kanvas

1. Dari kanvas, pilih Menu untuk membuka menu.
2. Di bagian Buka, pilih folder Proyek.

 Note

Jika opsi folder Proyek tidak tersedia, browser Anda mungkin tidak mendukung Akses Sistem FileAPI, yang diperlukan untuk mode sinkronisasi lokal. Untuk informasi selengkapnya, lihat [Izinkan akses halaman web ke file lokal di Infrastructure Composer](#).

3. Untuk lokasi Proyek, pilih Pilih folder. Pilih folder induk proyek Anda dan pilih Pilih.
4. Saat diminta oleh browser Anda, pilih Lihat file.
5. Untuk file Template, pilih template Anda dari daftar dropdown. Jika proyek Anda berisi satu template, Infrastructure Composer secara otomatis memilihnya untuk Anda.
6. Pilih Buat.

Saat Anda mengimpor folder proyek yang ada, Infrastructure Composer mengaktifkan mode sinkronisasi lokal. Perubahan yang dibuat pada template atau file proyek Anda secara otomatis disimpan ke mesin lokal Anda.

## Impor template proyek yang ada di konsol Infrastructure Composer

Saat Anda mengimpor AWS SAM template AWS CloudFormation atau yang sudah ada, Infrastructure Composer secara otomatis menghasilkan visualisasi arsitektur aplikasi Anda di kanvas.

Anda dapat mengimpor template proyek dari mesin lokal Anda.

Untuk mengimpor template proyek yang ada

1. Masuk ke [konsol Infrastructure Composer](#).
2. Pilih Buat proyek untuk membuka kanvas kosong.

3. Pilih Menu untuk membuka menu.
4. Di bagian Buka, pilih file Template.
5. Pilih template Anda dan pilih Buka.

Untuk menyimpan perubahan pada template Anda, Anda harus mengekspor template Anda atau mengaktifkan mode sinkronisasi lokal.

## Simpan template proyek yang ada di konsol Infrastructure Composer

Jika Anda tidak menggunakan mode sinkronisasi lokal, Anda harus mengekspor template Anda untuk menyimpan perubahan Anda. Jika Anda mengaktifkan mode sinkronisasi lokal, menyimpan templat Anda secara manual tidak diperlukan. Perubahan secara otomatis disimpan ke mesin lokal Anda.

Untuk menyimpan template proyek yang ada

1. Dari kanvas Infrastructure Composer, pilih Menu untuk membuka menu.
2. Di bagian Simpan, pilih Simpan file templat.
3. Berikan nama untuk template Anda.
4. Pilih lokasi untuk menyimpan template Anda.
5. Pilih Simpan.

## Connect konsol Infrastructure Composer dengan lokal Anda IDE

Untuk menghubungkan konsol Infrastructure Composer dengan lingkungan pengembangan terintegrasi lokal Anda (IDE), gunakan mode sinkronisasi lokal. Mode ini secara otomatis menyinkronkan dan menyimpan data ke mesin lokal Anda. Untuk informasi selengkapnya tentang mode sinkronisasi lokal, lihat [Sinkronkan dan simpan proyek Anda secara lokal di konsol Infrastructure Composer](#). Untuk petunjuk tentang penggunaan mode sinkronisasi lokal, lihat [Sinkronkan dan simpan proyek Anda secara lokal di konsol Infrastructure Composer](#).

### Note

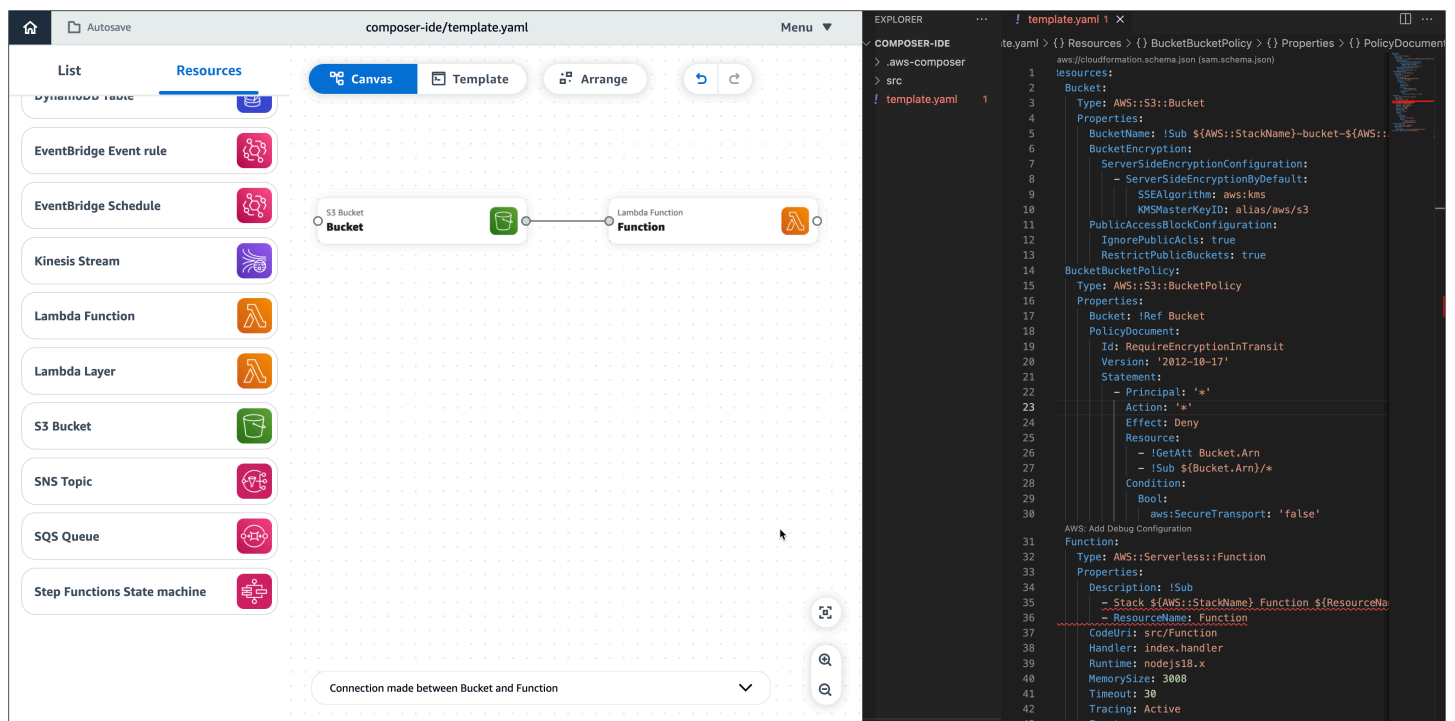
Opsi Aktifkan sinkronisasi lokal tidak tersedia di setiap browser. Ini tersedia di Google Chrome dan Microsoft Edge.

## Manfaat menggunakan Infrastructure Composer dengan lokal Anda IDE

Saat Anda mendesain di Infrastructure Composer, templat lokal dan direktori proyek Anda secara otomatis disinkronkan dan disimpan.

Anda dapat menggunakan lokal Anda IDE untuk melihat perubahan dan memodifikasi template Anda. Perubahan yang Anda buat secara lokal secara otomatis disinkronkan ke Infrastructure Composer.

Anda dapat menggunakan alat lokal seperti AWS Serverless Application Model Command Line Interface (AWS SAM CLI) untuk membangun, menguji, menyebarkan aplikasi Anda, dan banyak lagi. Contoh berikut menunjukkan bagaimana Anda dapat menarik dan melepas sumber daya ke kanvas visual Infrastructure Composer yang, pada gilirannya, membuat markup di AWS SAM template Anda di lokal Anda. IDE



## Integrasikan Infrastructure Composer dengan lokal Anda IDE

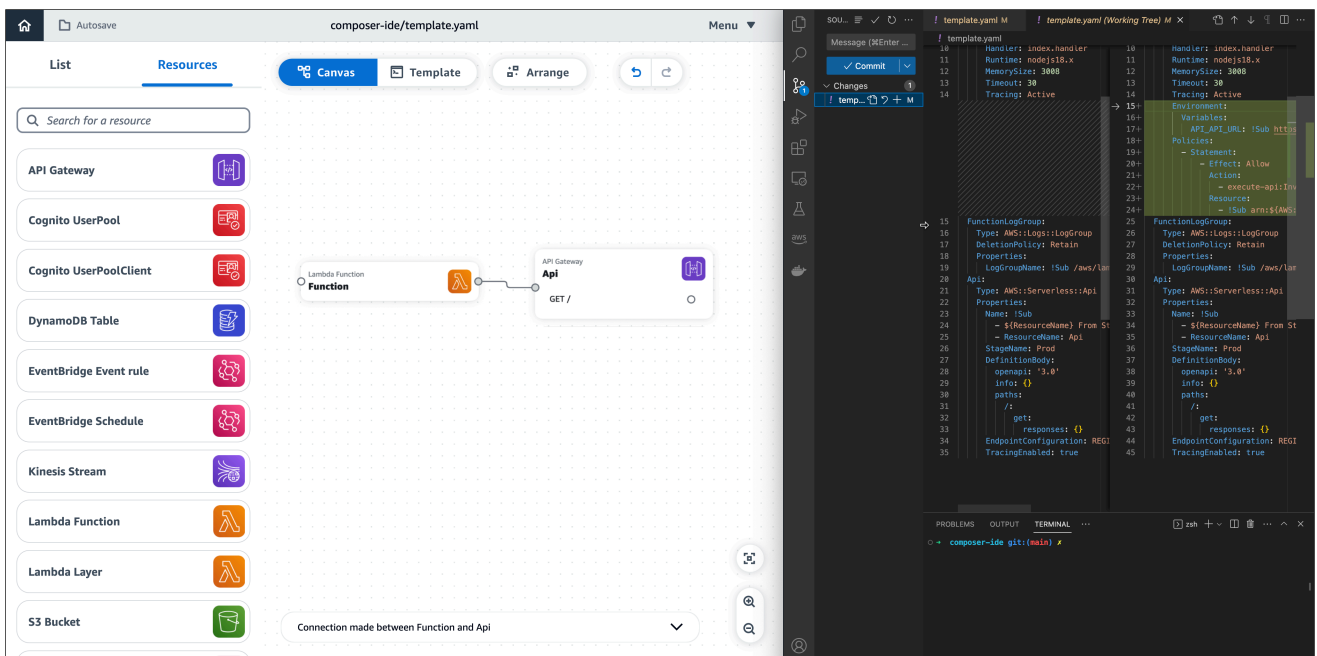
Untuk mengintegrasikan Infrastructure Composer dengan lokal Anda IDE

1. Di Infrastructure Composer, buat atau muat proyek, dan aktifkan sinkronisasi lokal dengan memilih tombol Menu di sisi kanan atas layar dan memilih Aktifkan sinkronisasi lokal.

### Note

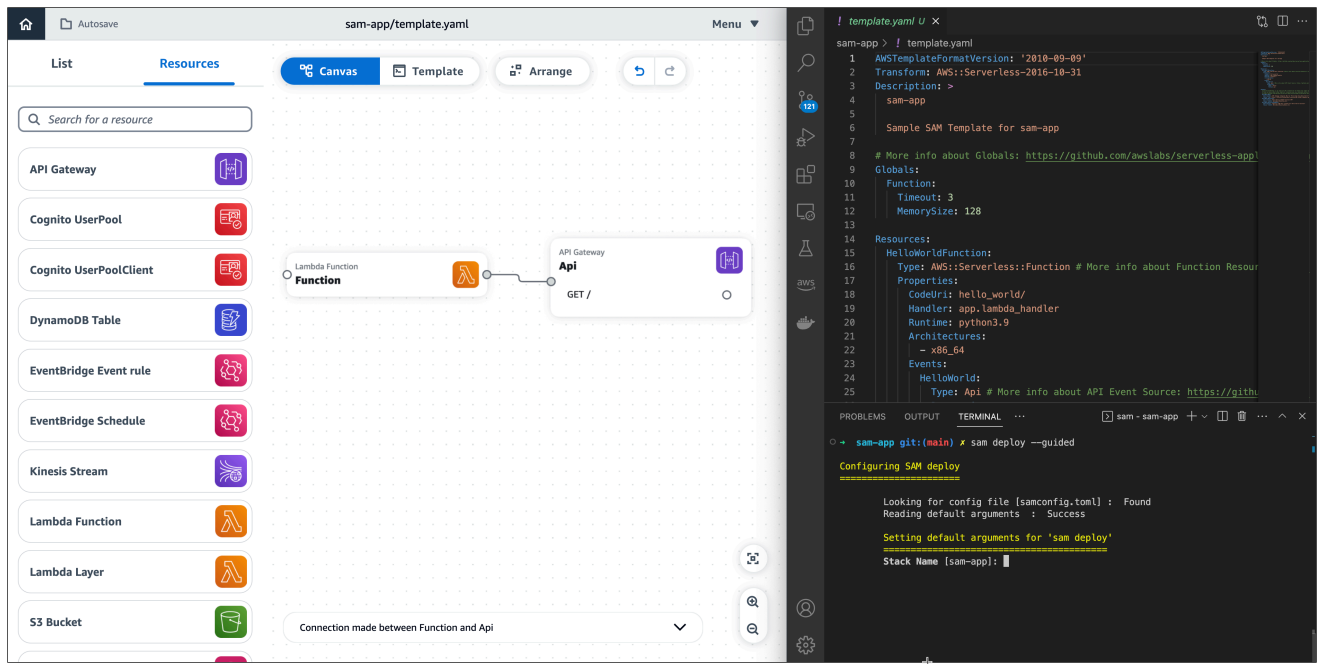
Opsi Aktifkan sinkronisasi lokal tidak tersedia di setiap browser. Ini tersedia di Google Chrome dan Microsoft Edge.

2. Di lokal Anda IDE, buka folder proyek yang sama dengan Infrastructure Composer.
3. Gunakan Infrastructure Composer dengan lokal IDE Anda. Pembaruan yang dibuat di Infrastructure Composer akan secara otomatis disinkronkan dengan mesin lokal Anda. Berikut adalah beberapa contoh dari apa yang dapat Anda lakukan:
  - a. Gunakan sistem kontrol versi pilihan Anda untuk melacak pembaruan yang dilakukan oleh Infrastructure Composer.



- b. Gunakan AWS SAM CLI secara lokal untuk membangun, menguji, menyebarkan aplikasi Anda, dan banyak lagi. Untuk mempelajari selengkapnya, lihat [Menerapkan aplikasi tanpa server Infrastructure Composer Anda ke Cloud AWS](#).





## Izinkan akses halaman web ke file lokal di Infrastructure Composer

Konsol Infrastructure Composer mendukung [mode sinkronisasi lokal](#) dan [Mengimpor fungsi dari konsol Lambda](#). Untuk menggunakan fitur-fitur ini, diperlukan browser web yang mendukung Akses Sistem File. Versi terbaru Google Chrome dan Microsoft Edge mendukung semua kemampuan Akses Sistem File API dan dapat digunakan dengan mode sinkronisasi lokal di Infrastructure Composer.

Akses Sistem File API memungkinkan halaman web mendapatkan akses ke sistem file lokal Anda untuk membaca, menulis, atau menyimpan file. Fitur ini tidak aktif secara default dan memerlukan izin Anda melalui prompt visual untuk mengizinkannya. Setelah diberikan, akses ini tetap selama sesi browser halaman web Anda.

Untuk mempelajari lebih lanjut tentang Akses Sistem File API, lihat:

- [Akses Sistem File API](#) di dokumen web mdn.
- [Akses Sistem File API: menyederhanakan akses ke file lokal](#) di situs web.dev.

## modus sinkronisasi lokal

Mode sinkronisasi lokal memungkinkan Anda secara otomatis menyinkronkan dan menyimpan file template dan folder proyek Anda secara lokal saat Anda mendesain di Infrastructure Composer. Untuk menggunakan fitur ini, diperlukan browser web yang mendukung Akses API Sistem File.

### Data Infrastructure Composer mendapatkan akses ke

Infrastructure Composer mendapatkan akses baca dan tulis ke folder proyek yang Anda izinkan, bersama dengan folder anak dari folder proyek tersebut. Akses ini digunakan untuk membuat, memperbarui, dan menyimpan file template, folder proyek, dan direktori cadangan yang dihasilkan saat Anda mendesain. Data yang diakses oleh Infrastructure Composer tidak digunakan untuk tujuan lain dan tidak disimpan di mana pun di luar sistem file lokal Anda.

### Akses ke data sensitif

Akses Sistem File API mengecualikan atau membatasi akses ke direktori tertentu yang mungkin berisi data sensitif. Kesalahan akan terjadi jika Anda memilih salah satu direktori ini untuk digunakan dengan mode sinkronisasi lokal Infrastructure Composer. Anda dapat memilih direktori lokal lain untuk terhubung atau menggunakan Infrastructure Composer dalam mode default dengan sinkronisasi lokal dinonaktifkan.

Untuk informasi selengkapnya, termasuk contoh direktori sensitif, lihat [Pengguna yang memberikan akses ke file yang lebih, atau lebih sensitif daripada yang dimaksudkan dalam Laporan](#) Grup Komunitas Draf W3C Akses Sistem File.

Jika Anda menggunakan Windows Subsystem for Linux (WSL), Akses Sistem File API tidak termasuk akses ke seluruh Linux direktori karena lokasinya di dalam Windows sistem. Anda dapat menggunakan Infrastructure Composer dengan sinkronisasi lokal dinonaktifkan atau mengonfigurasi solusi untuk menyinkronkan file proyek dari WSL direktori ke direktori kerja di Windows. Kemudian, gunakan mode sinkronisasi lokal Infrastructure Composer dengan Windows direktori.

## Sinkronkan dan simpan proyek Anda secara lokal di konsol Infrastructure Composer

Bagian ini memberikan informasi tentang penggunaan mode sinkronisasi lokal Infrastructure Composer untuk secara otomatis menyinkronkan dan menyimpan proyek Anda ke mesin lokal Anda.

Kami menyarankan Anda menggunakan sinkronisasi lokal untuk alasan berikut:

Anda dapat mengaktifkan sinkronisasi lokal untuk proyek baru, atau memuat proyek yang ada dengan sinkronisasi lokal diaktifkan.

- Secara default, Anda perlu menyimpan template aplikasi secara manual saat Anda mendesain. Gunakan sinkronisasi lokal untuk secara otomatis menyimpan template aplikasi Anda ke mesin lokal Anda saat Anda membuat perubahan.
- Sinkronisasi lokal mengelola dan secara otomatis menyinkronkan folder proyek, folder cadangan, dan [file eksternal yang didukung](#) ke mesin lokal Anda.
- Saat menggunakan sinkronisasi lokal, Anda dapat menghubungkan Infrastructure Composer dengan lokal Anda IDE untuk mempercepat pengembangan. Untuk mempelajari selengkapnya, lihat [Connect konsol Infrastructure Composer dengan lokal Anda IDE](#).

## Mode sinkronisasi lokal apa yang disimpan

Mode sinkronisasi lokal secara otomatis menyinkronkan dan menyimpan yang berikut ini ke mesin lokal Anda:

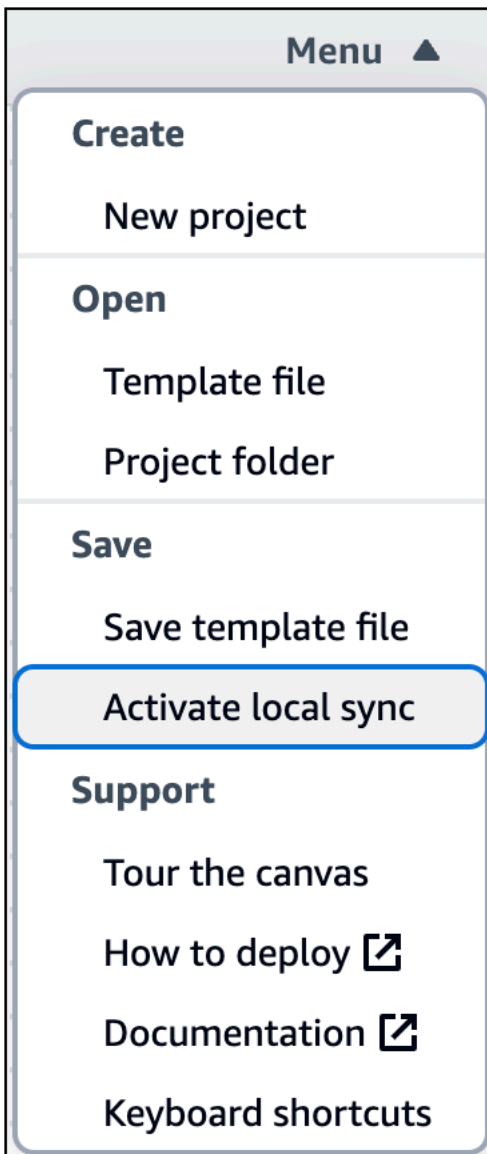
- File template aplikasi — Template AWS CloudFormation or AWS Serverless Application Model (AWS SAM) yang berisi infrastruktur Anda sebagai kode (IaC).
- Folder proyek — Struktur direktori umum yang mengatur AWS Lambda fungsi Anda.
- Direktori Backup — Direktori cadangan bernama `.aws-composer`, dibuat di root lokasi proyek Anda. Direktori ini berisi salinan cadangan file template aplikasi dan folder proyek Anda.
- File eksternal - File eksternal yang didukung yang dapat Anda gunakan dalam Infrastructure Composer. Untuk mempelajari selengkapnya, lihat [Referensi file eksternal di Infrastructure Composer](#).

## Persyaratan browser

Mode sinkronisasi lokal memerlukan browser yang mendukung Akses Sistem FileAPI. Untuk informasi selengkapnya, lihat [Izinkan akses halaman web ke file lokal di Infrastructure Composer](#).

## Mengaktifkan mode sinkronisasi lokal

Mode sinkronisasi lokal dinonaktifkan secara default. Anda dapat mengaktifkan mode sinkronisasi lokal melalui menu Infrastructure Composer.



Untuk petunjuk tentang mengaktifkan sinkronisasi lokal dan proyek pemuatan yang ada, lihat topik berikut:

- [Aktifkan sinkronisasi lokal di Infrastructure Composer](#)
- [Muat proyek Infrastructure Composer yang ada dengan sinkronisasi lokal diaktifkan](#)

## Aktifkan sinkronisasi lokal di Infrastructure Composer

Untuk mengaktifkan sinkronisasi lokal, selesaikan langkah-langkah berikut:

1. Dari [halaman](#) beranda Infrastructure Composer, pilih Create project.

2. Dari menu Infrastructure Composer, pilih Activate local sync.
3. Untuk lokasi Proyek, tekan Pilih folder dan pilih direktori. Di sinilah Infrastructure Composer akan menyimpan dan menyinkronkan file dan folder template Anda saat Anda mendesain.

 Note

Lokasi proyek tidak boleh berisi template aplikasi yang ada.

4. Saat diminta untuk mengizinkan akses, pilih Lihat file.
5. Tekan Aktifkan. Saat diminta untuk menyimpan perubahan, pilih Simpan perubahan.

Saat diaktifkan, indikator Autosave akan ditampilkan di area kiri atas kanvas Anda.

## Muat proyek Infrastructure Composer yang ada dengan sinkronisasi lokal diaktifkan

Untuk memuat proyek yang ada dengan sinkronisasi lokal diaktifkan, selesaikan langkah-langkah berikut:

1. Dari [halaman](#) beranda Infrastructure Composer, pilih Load a AWS CloudFormation template.
2. Dari menu Infrastructure Composer, pilih Open > Project folder.
3. Untuk lokasi Proyek, tekan Pilih folder dan pilih folder root proyek Anda.
4. Saat diminta untuk mengizinkan akses, pilih Lihat file.
5. Untuk file Template, pilih template aplikasi Anda dan tekan Create.
6. Saat diminta untuk menyimpan perubahan, pilih Simpan perubahan.

Saat diaktifkan, indikator Autosave akan ditampilkan di area kiri atas kanvas Anda.

## Impor fungsi ke Infrastructure Composer dari konsol Lambda

Infrastructure Composer menyediakan integrasi dengan AWS Lambda konsol. Anda dapat mengimpor fungsi Lambda dari konsol Lambda ke konsol Infrastructure Composer. Kemudian, gunakan kanvas Infrastructure Composer untuk mendesain arsitektur aplikasi Anda lebih lanjut.

- Integrasi ini membutuhkan browser yang mendukung Akses Sistem FileAPI. Untuk informasi selengkapnya, lihat [Izinkan akses halaman web ke file lokal di Infrastructure Composer](#).

- Saat Anda mengimpor fungsi Lambda Anda ke Infrastructure Composer, Anda harus mengaktifkan mode sinkronisasi lokal untuk menyimpan perubahan apa pun. Untuk informasi selengkapnya, lihat [Sinkronkan dan simpan proyek Anda secara lokal di konsol Infrastructure Composer](#).

Untuk memulai menggunakan integrasi ini, lihat [Menggunakan AWS Lambda dengan AWS Infrastructure Composer](#) di Panduan AWS Lambda Pengembang.

## Ekspor gambar kanvas visual Infrastructure Composer

Topik ini menjelaskan fitur kanvas ekspor AWS Infrastructure Composer konsol.

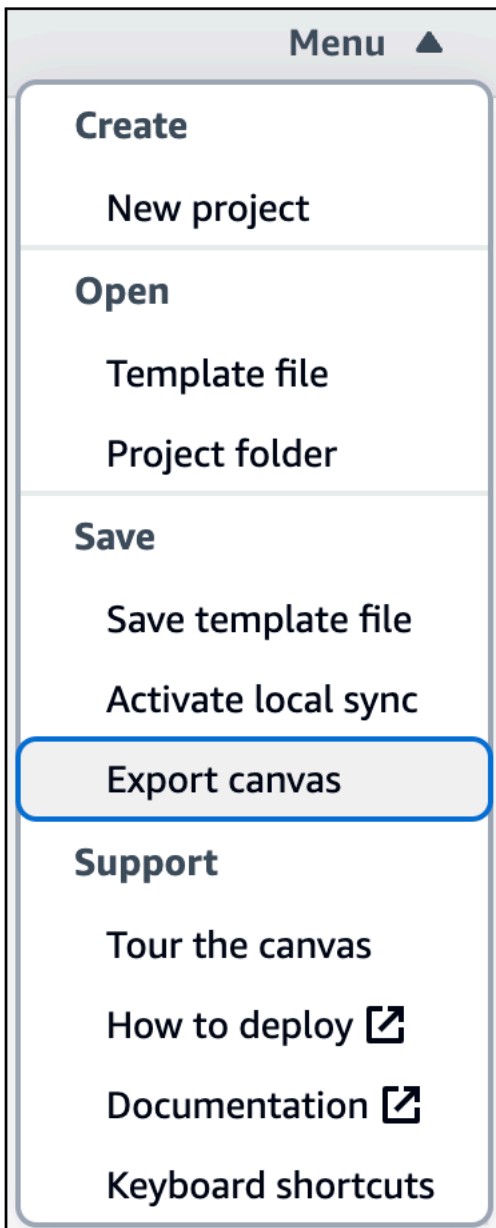
Untuk gambaran visual dari semua fitur Infrastructure Composer, lihat [AWS Infrastructure Composer ikhtisar visual konsol](#).

### Tentang kanvas ekspor

Fitur kanvas ekspor mengeksport kanvas aplikasi Anda sebagai gambar ke mesin lokal Anda.

- Infrastructure Composer menghapus elemen UI desainer visual dan hanya mengeksport diagram aplikasi Anda.
- Format file gambar default adalah png.
- File diekspor ke lokasi unduhan default mesin lokal Anda.

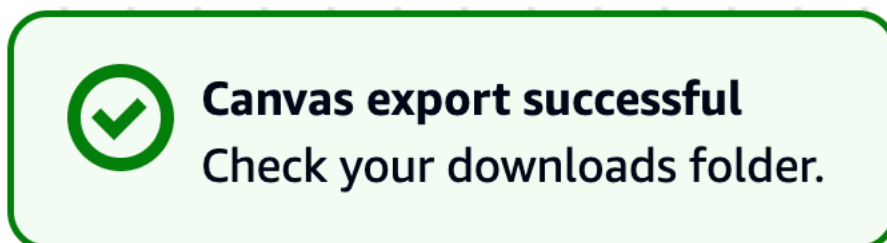
Anda dapat mengakses fitur kanvas ekspor dari Menu.



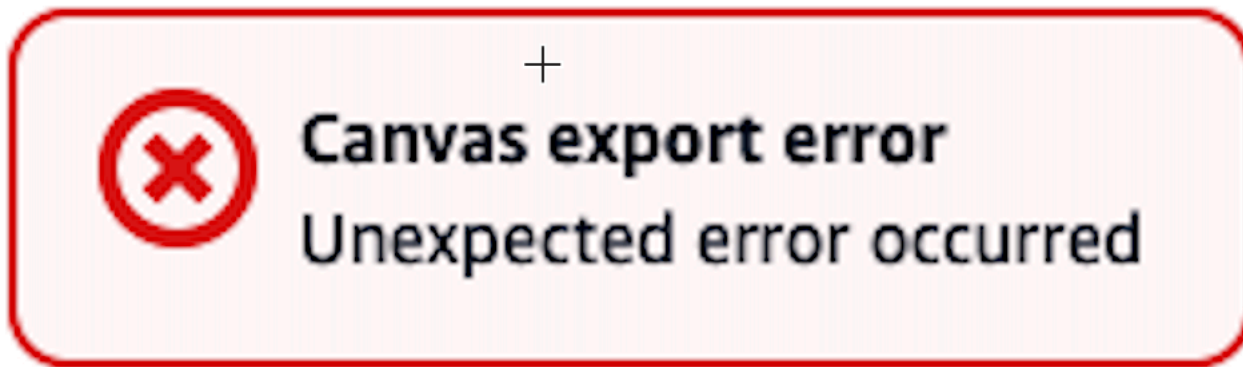
## Mengekspor kanvas

Saat Anda mengekspor kanvas Anda, Infrastructure Composer menampilkan pesan status.

Jika ekspor berhasil, Anda akan melihat pesan berikut:



Jika ekspor tidak berhasil, Anda akan melihat pesan kesalahan. Jika Anda menerima kesalahan, coba ekspor lagi.



## Menggunakan Infrastructure Composer dalam mode CloudFormation konsol

Infrastructure Composer dalam mode CloudFormation konsol adalah alat yang disarankan untuk memvisualisasikan template Anda AWS CloudFormation . Anda juga dapat menggunakan alat ini untuk membuat dan mengedit AWS CloudFormation template.

### Bagaimana mode ini berbeda dari konsol Infrastructure Composer?

Infrastructure Composer dalam mode CloudFormation konsol umumnya memiliki fungsi yang sama dengan [konsol Infrastructure Composer default](#), tetapi ada beberapa perbedaan yang perlu diperhatikan.

- Mode ini terintegrasi dengan alur kerja tumpukan di AWS CloudFormation konsol. Ini memungkinkan Anda untuk menggunakan Infrastructure Composer langsung di AWS CloudFormation.
- [Sinkronkan dan simpan proyek Anda secara lokal di konsol Infrastructure Composer](#), fitur yang secara otomatis menyinkronkan dan menyimpan data ke mesin lokal Anda, tidak didukung.
- Kartu terkait Lambda (Fungsi Lambda dan Lapisan Lambda) memerlukan pembuatan kode dan solusi pengemasan yang tidak tersedia dalam mode ini.



**Note**

Kartu ini dan sinkronisasi lokal dapat digunakan di [Infrastructure Composer Console](#) atau AWS Toolkit for Visual Studio Code

Saat Anda membuka Infrastructure Composer dari AWS CloudFormation konsol, Infrastructure Composer terbuka dalam mode CloudFormation konsol. Dalam mode ini, Anda dapat menggunakan Infrastructure Composer untuk memvisualisasikan, membuat, dan memperbarui template Anda.

## Cara mengakses Infrastructure Composer dalam mode CloudFormation konsol

Infrastructure Composer dalam mode CloudFormation konsol adalah upgrade dari AWS CloudFormation Designer. Sebaiknya gunakan Infrastructure Composer untuk memvisualisasikan template Anda AWS CloudFormation . Anda juga dapat menggunakan alat ini untuk membuat dan mengedit AWS CloudFormation template.

1. Buka [konsol Cloudformation](#) dan masuk.
2. Pilih Infrastructure Composer dari menu navigasi sisi kiri. Ini akan membawa Anda ke Infrastructure Composer dalam mode CloudFormation konsol.

**Note**

Untuk informasi tentang penggunaan Infrastructure Composer dalam mode CloudFormation konsol, lihat [Menggunakan Infrastructure Composer dalam mode CloudFormation konsol](#).

## Visualisasikan penerapan di Infrastructure Composer dalam mode konsol CloudFormation

Ikuti petunjuk dalam topik ini untuk memvisualisasikan template AWS CloudFormation stack/ Infrastructure Composer yang digunakan.

1. Buka [AWS CloudFormation konsol](#) dan masuk.
2. Pilih tumpukan yang ingin Anda edit.

3. Pilih tab Template.
4. Pilih Komposer Infrastruktur.

Infrastructure Composer akan memvisualisasikan tumpukan/template Anda. Perubahan juga bisa dilakukan di sini.

## Buat template baru di Infrastructure Composer dalam mode CloudFormation konsol

Ikuti petunjuk dalam topik ini untuk membuat template baru.

1. Buka [AWS CloudFormation konsol](#) dan masuk.
2. Pilih Infrastructure Composer dari menu navigasi sisi kiri. Ini akan membuka Infrastructure Composer dalam mode CloudFormation konsol.
3. Seret, lepas, konfigurasi, dan hubungkan sumber daya ([kartu](#)) yang Anda butuhkan dari palet Resources.

### Note

Lihat [Cara menulis](#) detail tentang penggunaan Infrastructure Composer, dan perhatikan bahwa kartu terkait Lambda (Fungsi Lambda dan Lapisan Lambda) memerlukan pembuatan kode dan solusi pengemasan yang tidak tersedia di Infrastructure Composer dalam mode konsol. CloudFormation Kartu-kartu ini dapat digunakan di [konsol Infrastructure Composer](#) atau AWS Toolkit for Visual Studio Code Untuk informasi tentang penggunaan alat ini, lihat [Di mana Anda dapat menggunakan Infrastructure Composer](#).

4. Klik dua kali kartu untuk menggunakan panel Resource properties untuk menentukan bagaimana kartu dikonfigurasi.
5. [Hubungkan kartu Anda](#) untuk menentukan alur kerja berbasis peristiwa aplikasi Anda.
6. Pilih Template untuk melihat dan mengedit kode infrastruktur Anda. Perubahan secara otomatis disinkronkan dengan tampilan kanvas Anda.
7. Setelah template Anda siap untuk diekspor ke tumpukan, pilih Buat template.
8. Pilih CloudFormation tombol Konfirmasi dan ekspor ke. Ini akan membawa Anda kembali ke alur kerja create stack dengan pesan yang mengonfirmasi template Anda berhasil diimpor.

**Note**

Hanya template dengan sumber daya di dalamnya yang dapat diekspor.

9. Dalam alur kerja Buat tumpukan, pilih Berikutnya.
10. Berikan nama tumpukan, tinjau parameter apa pun yang tercantum, dan pilih Berikutnya.

**Note**

Nama tumpukan harus dimulai dengan huruf dan hanya berisi huruf, angka, tanda hubung.

11. Pilih Berikutnya setelah memberikan informasi berikut:
  - Tag yang terkait dengan tumpukan
  - Izin tumpukan
  - Opsi kegagalan tumpukan

**Note**

Untuk panduan mengelola tumpukan, lihat [praktik AWS CloudFormation terbaik](#) di Panduan AWS CloudFormation Pengguna.

12. Konfirmasikan detail tumpukan Anda benar, periksa ucapan terima kasih di bagian bawah halaman, dan pilih tombol Kirim.

AWS CloudFormation akan mulai membuat tumpukan berdasarkan data di template Anda.


## Perbarui tumpukan yang ada di Infrastructure Composer dalam mode CloudFormation konsol

Ikuti petunjuk dalam topik ini untuk memperbarui AWS CloudFormation tumpukan yang ada.

**Note**


Jika file Anda disimpan secara lokal, sebaiknya gunakan [AWS Toolkit for Visual Studio Code](#).

1. Buka [AWS CloudFormation konsol](#) dan masuk.
2. Pilih tumpukan yang ingin Anda edit.
3. Pilih tombol Perbarui. Melakukan hal ini akan membawa Anda ke update stack wizard.
4. Di sebelah kanan, pilih Edit di Infrastructure Composer.
5. Pilih tombol di bawah ini yang berlabel Edit di Infrastructure Composer. Ini akan membawa Anda ke Infrastructure Composer dalam mode CloudFormation konsol.
6. Di sini, Anda dapat menyeret, melepas, mengonfigurasi, dan menghubungkan sumber daya ([kartu](#)) dari palet Sumber Daya.

 Note

Lihat [Cara menulis](#) detail tentang penggunaan Infrastructure Composer, dan perhatikan bahwa kartu terkait Lambda (Fungsi Lambda dan Lapisan Lambda) memerlukan pembuatan kode dan solusi pengemasan yang tidak tersedia di Infrastructure Composer dalam mode konsol. CloudFormation Kartu-kartu ini dapat digunakan di [konsol Infrastructure Composer](#) atau. AWS Toolkit for Visual Studio Code Untuk informasi tentang penggunaan alat ini, lihat [Di mana Anda dapat menggunakan Infrastructure Composer](#).

7. Ketika Anda siap untuk mengekspor perubahan ke AWS CloudFormation, pilih Perbarui template.
8. Pilih Konfirmasi dan lanjutkan ke CloudFormation. Ini akan membawa Anda kembali ke alur kerja tumpukan Pembaruan dengan pesan yang mengonfirmasi bahwa template Anda berhasil diimpor.

 Note

Hanya template dengan sumber daya di dalamnya yang dapat diekspor.

9. Dalam alur kerja Update stack, pilih Next.
10. Tinjau parameter yang tercantum dan pilih Berikutnya.
11. Pilih Berikutnya setelah memberikan informasi berikut:
  - Tag yang terkait dengan tumpukan
  - Izin tumpukan
  - Opsi kegagalan tumpukan

**Note**

Untuk panduan mengelola tumpukan, lihat [praktik AWS CloudFormation terbaik](#) di Panduan AWS CloudFormation Pengguna.

12. Konfirmasikan detail tumpukan Anda benar, periksa ucapan terima kasih di bagian bawah halaman, dan pilih tombol Kirim.

AWS CloudFormation akan mulai memperbarui tumpukan berdasarkan pembaruan yang Anda buat di template Anda.

## Menggunakan Infrastructure Composer dari AWS Toolkit for Visual Studio Code

Bagian ini menjelaskan bagaimana Anda dapat menggunakan AWS Infrastructure Composer dari [AWS Toolkit for Visual Studio Code](#). Ini termasuk gambaran visual Komposer Infrastruktur dari AWS Toolkit for Visual Studio Code. Ini juga mencakup instruksi yang menunjukkan bagaimana Anda dapat mengakses pengalaman ini dan menyinkronkan proyek Anda dari VS Code ke AWS cloud. Untuk menyinkronkan, Anda menggunakan `aws sync` perintah dari AWS SAM CLI. Bagian ini juga memberikan panduan tentang penggunaan Amazon Q sementara di Infrastructure Composer dari AWS Toolkit for Visual Studio Code

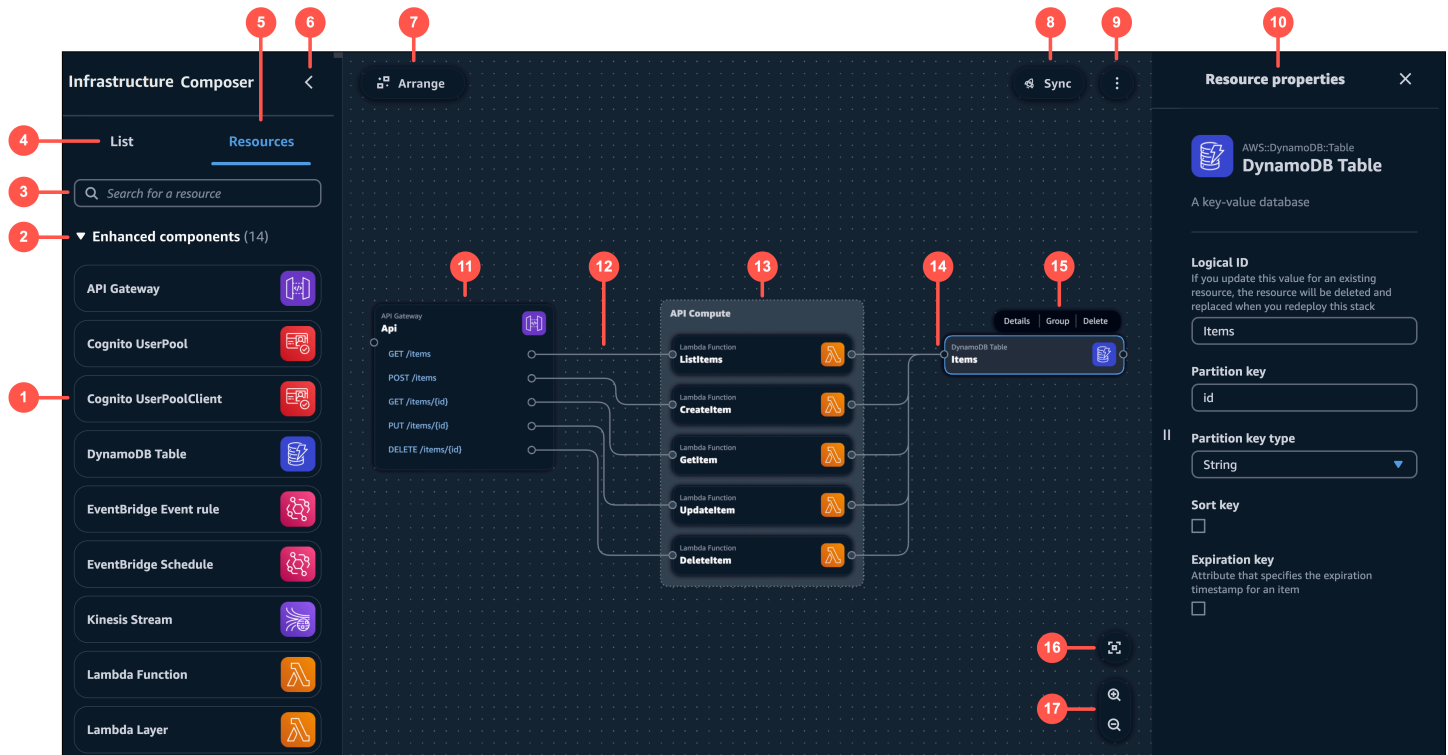
Untuk panduan tambahan tentang penggunaan Infrastructure Composer dari AWS Toolkit for Visual Studio Code, lihat [Cara menulis](#) Konten di bagian ini berlaku untuk pengalaman ini, serta pengalaman konsol Infrastructure Composer.

### Topik

- [Gambaran visual Komposer Infrastruktur dari AWS Toolkit for Visual Studio Code](#)
- [Akses Komposer Infrastruktur dari AWS Toolkit for Visual Studio Code](#)
- [Sync Infrastructure Composer untuk menyebarkan ke AWS Cloud](#)
- [Menggunakan AWS Infrastructure Composer dengan Amazon Q Developer](#)

# Gambaran visual Komposer Infrastruktur dari AWS Toolkit for Visual Studio Code

Desainer visual Infrastructure Composer AWS Toolkit for Visual Studio Code termasuk kanvas visual, yang mencakup komponen yang diberi nomor dalam gambar berikut dan tercantum di bawah ini.



1. Palet sumber daya - Menampilkan kartu yang dapat Anda desain dengan.
2. Kategori kartu - Kartu diatur berdasarkan kategori yang unik untuk Komposer Infrastruktur.
3. Bilah pencarian sumber daya — Cari kartu yang dapat Anda tambahkan ke kanvas.
4. Daftar - Menampilkan tampilan pohon dari sumber daya aplikasi Anda.
5. Sumber Daya - Menampilkan palet sumber daya.
6. Toggle panel kiri — Sembunyikan atau tampilkan panel kiri.
7. Atur - Mengatur arsitektur aplikasi Anda di kanvas.
8. Sinkronisasi - Memulai AWS Serverless Application Model (SAM) CLI `sam sync` perintah untuk menyebarkan aplikasi Anda.
9. Menu - Menyediakan opsi umum seperti berikut ini:
  - Ekspor kanvas
  - Tur kanvas

- Link ke Dokumentasi
- Pintasan keyboard

10Panel properti sumber daya - Menampilkan properti yang relevan untuk kartu yang telah dipilih di kanvas. Panel ini dinamis. Properti yang ditampilkan akan berubah saat Anda mengkonfigurasi kartu Anda.

11Kartu — Menampilkan tampilan kartu Anda di kanvas.

12Line — Merupakan koneksi antar kartu.

13Grup — Sekelompok kartu. Anda dapat mengelompokkan kartu untuk organisasi visual.

14Port — Koneksi menunjuk ke kartu lain.

15.Tindakan kartu — Menyediakan tindakan yang dapat Anda ambil pada kartu Anda.

- Detail - Memunculkan panel properti Resource.
- Grup — Kelompokkan kartu yang dipilih bersama.
- Hapus — Menghapus kartu dari kanvas dan template Anda.

16Re-center — Pusatkan kembali diagram aplikasi Anda pada kanvas visual.

17Zoom - Zoom in dan out pada kanvas Anda.

## Akses Komposer Infrastruktur dari AWS Toolkit for Visual Studio Code

Ikuti petunjuk dalam topik ini untuk mengakses Infrastructure Composer dari AWS Toolkit for Visual Studio Code

### Note

Sebelum Anda dapat mengakses Infrastructure Composer dari AWS Toolkit for Visual Studio Code, Anda harus terlebih dahulu mengunduh dan menginstal Toolkit for VS Code. Untuk petunjuk, lihat [Mengunduh Toolkit for VS Code](#).

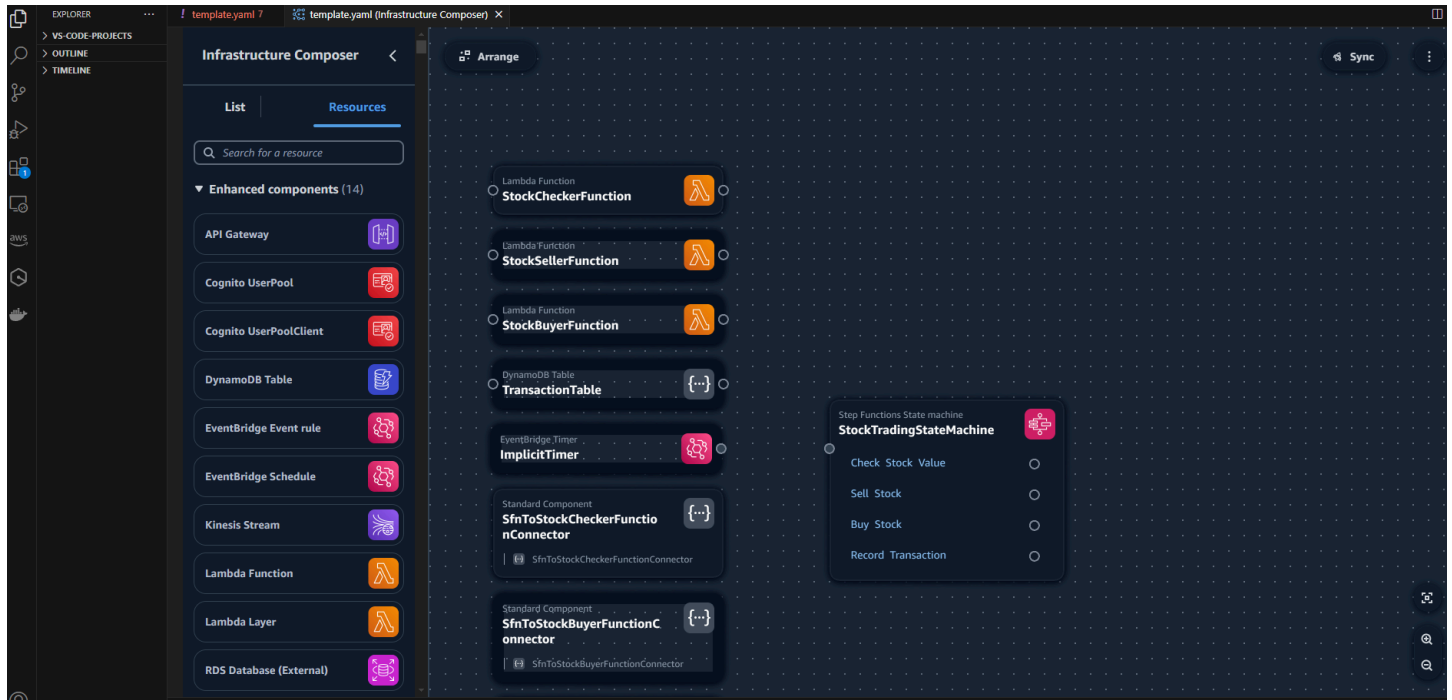
Untuk mengakses Infrastructure Composer dari Toolkit for VS Code

Anda dapat mengakses Infrastructure Composer dengan salah satu cara berikut:

1. Dengan memilih tombol Infrastructure Composer dari salah satu AWS CloudFormation atau AWS SAM template.

2. Melalui menu konteks dengan mengklik kanan pada AWS SAM template AWS CloudFormation atau Anda.
3. Dari Palet Perintah Kode VS.

Berikut ini adalah contoh mengakses Infrastructure Composer dari tombol Infrastructure Composer:



Untuk informasi selengkapnya tentang mengakses Infrastructure Composer, lihat [Mengakses AWS Infrastructure Composer dari Toolkit](#).

## Sync Infrastructure Composer untuk menyebarkan ke AWS Cloud

Gunakan tombol sinkronisasi AWS Infrastructure Composer dari AWS Toolkit for Visual Studio Code untuk menyebarkan aplikasi Anda ke file. AWS Cloud

Tombol sinkronisasi memulai `sam sync` perintah dari Antarmuka Baris AWS SAM Perintah (CLI).

`sam sync` Perintah dapat menyebarkan aplikasi baru atau dengan cepat menyinkronkan perubahan yang Anda buat secara lokal ke file. AWS Cloud Berlari `sam sync` mungkin termasuk yang berikut:

- Membangun aplikasi Anda `sam build` untuk mempersiapkan file aplikasi lokal Anda untuk penyebaran dengan membuat atau memperbarui `.aws-sam` direktori lokal.



- Untuk sumber daya yang mendukung AWS layanan APIs, AWS SAM CLI akan menggunakan APIs untuk menyebarkan perubahan Anda. The AWS SAM CLI melakukan ini untuk memperbarui sumber daya Anda dengan cepat di cloud.
- Jika perlu, AWS SAM CLI melakukan AWS CloudFormation penerapan untuk memperbarui seluruh tumpukan Anda melalui set perubahan.

`sam sync` Perintah ini paling cocok untuk lingkungan pengembangan yang cepat saat memperbarui sumber daya cloud Anda dengan cepat dapat bermanfaat bagi pengembangan dan pengujian alur kerja Anda.

Untuk mempelajari selengkapnya `sam sync`, lihat [Menggunakan sinkronisasi sam](#) di Panduan AWS Serverless Application Model Pengembang.

## Penyiapan

Untuk menggunakan fitur sinkronisasi di Infrastructure Composer, Anda harus memiliki AWS SAM CLI diinstal pada mesin lokal Anda. Untuk petunjuk, lihat [Memasang AWS SAM CLI](#) di Panduan Developer AWS Serverless Application Model .

Saat Anda menggunakan fitur sinkronisasi di Infrastructure Composer, AWS SAM CLI mereferensikan file konfigurasi Anda untuk informasi yang dibutuhkan untuk menyinkronkan aplikasi Anda ke file AWS Cloud. Untuk petunjuk cara membuat, memodifikasi, dan menggunakan file konfigurasi, lihat [Mengkonfigurasi pengaturan proyek](#) di Panduan AWS Serverless Application Model Pengembang.

## Sinkronkan dan terapkan aplikasi Anda

Untuk menyinkronkan aplikasi Anda ke AWS Cloud

1. Pilih tombol sinkronisasi pada kanvas Infrastructure Composer.
2. Anda mungkin menerima prompt untuk mengonfirmasi bahwa Anda bekerja dengan tumpukan pengembangan. Pilih OK untuk melanjutkan.
3. Infrastructure Composer dapat meminta Anda untuk mengkonfigurasi opsi berikut:
  - Wilayah AWS— Wilayah untuk menyinkronkan aplikasi Anda ke.
  - AWS CloudFormation nama tumpukan — Nama AWS CloudFormation tumpukan Anda. Anda dapat memilih nama tumpukan yang ada atau membuat yang baru.

- Bucket Amazon Simple Storage Service (Amazon S3) — Nama bucket Amazon S3 Anda. The AWS SAM CLI akan mengemas dan menyimpan file aplikasi dan kode fungsi Anda di sini. Anda dapat memilih bucket yang sudah ada atau membuat yang baru.

Infrastructure Composer akan memulai AWS SAM CLI `sam sync` perintah dan buka jendela terminal di Anda IDE untuk menampilkan kemajuannya.

## Menggunakan AWS Infrastructure Composer dengan Amazon Q Developer

AWS Infrastructure Composer dari AWS Toolkit for Visual Studio Code menyediakan integrasi dengan Amazon Q. Anda dapat menggunakan Amazon Q dalam Infrastructure Composer untuk menghasilkan kode infrastruktur untuk AWS sumber daya Anda saat Anda mendesain aplikasi Anda.

Amazon Q adalah tujuan umum, generator kode bertenaga pembelajaran mesin. Untuk mempelajari lebih lanjut, lihat [Apa itu Amazon Q?](#) di Amazon Q Developer Panduan Pengguna.

Untuk sumber daya standar dan kartu komponen standar, Anda dapat menggunakan Amazon Q untuk menghasilkan saran kode infrastruktur untuk sumber daya Anda.

The screenshot displays the AWS Infrastructure Composer interface. On the left, a grid of standard components is shown, with a 'VPC' card highlighted. The card is labeled 'Standard Component VPC' and includes a 'VPC' icon. Above the grid, there are 'Sync' and 'Details | Group | Delete' buttons. On the right, the 'Resource properties' panel is open for the 'AWS::EC2::VPC' resource. It shows the resource type as 'CFN Resource' and the editing mode as 'VPC'. The 'Logical ID' is set to 'VPC', with a note that updating it will generate a new resource. The 'Resource configuration' section is empty, with a note that updating it will change the resource's properties. A modal dialog is open, displaying a loading spinner and the text 'Generating can take up to 30 seconds.' with a 'Stop generating' button. A 'Resource reference' button is also visible at the bottom right of the properties panel.

Sumber daya standar dan kartu komponen standar dapat mewakili AWS CloudFormation sumber daya atau kumpulan sumber AWS CloudFormation daya. Untuk mempelajari selengkapnya, lihat [Mengkonfigurasi dan memodifikasi kartu di Infrastructure Composer](#).

## Pengaturan

Untuk menggunakan Amazon Q di Infrastructure Composer, Anda harus mengautentikasi dengan Amazon Q di Toolkit. Untuk petunjuk, lihat [Memulai Amazon Q di VS Code dan JetBrains](#) di Amazon Q Developer Panduan Pengguna.

## Penggunaan Amazon Q Developer di Infrastructure Composer

Anda dapat menggunakan Amazon Q Developer dari panel Resource properties dari setiap sumber daya standar atau kartu komponen standar.

Untuk menggunakan Amazon Q di Infrastructure Composer

1. Dari sumber daya standar atau kartu komponen standar, buka panel Resource properties.
2. Temukan bidang konfigurasi Sumber Daya. Bidang ini berisi kode infrastruktur untuk kartu.
3. Pilih tombol Hasilkan saran. Amazon Q akan menghasilkan saran.

### Note

Kode yang dihasilkan pada tahap ini tidak akan menimpa kode infrastruktur yang ada dari template Anda.

4. Untuk menghasilkan lebih banyak saran, pilih Regenerate. Anda dapat beralih melalui sampel untuk membandingkan hasil.
5. Untuk memilih opsi, pilih Pilih. Anda dapat memodifikasi kode di sini sebelum menyimpannya ke aplikasi Anda. Untuk keluar tanpa menyimpan, pilih ikon keluar (X).
6. Untuk menyimpan kode ke template aplikasi Anda, pilih Simpan dari panel Resource properties.

## Pelajari selengkapnya

Untuk mempelajari lebih lanjut tentang Amazon Q, lihat [Apa itu Amazon Q?](#) di Amazon Q Developer Panduan Pengguna.

# Cara menulis di AWS Infrastructure Composer

Bagian ini mencakup dasar-dasar penggunaan Infrastructure Composer dari [Konsol Komposer InfrastrukturCloudFormation modus konsol](#), dan [AWS Toolkit for Visual Studio Code](#). Lebih khusus lagi, topik di bagian ini memberikan detail kunci tentang cara membuat aplikasi dengan Infrastructure Composer, dan mencakup detail tentang fitur dan pintasan tambahan. Ada beberapa variasi fungsionalitas antara pengalaman konsol dan VS Code, dan topik di bagian ini mengidentifikasi dan menjelaskan variasi ini di mana mereka terjadi.

Setelah menyusun aplikasi Anda, Anda akan siap [Menerapkan aplikasi tanpa server Infrastructure Composer Anda ke Cloud AWS](#) untuk meninjau informasi tentang penerapan aplikasi Anda.

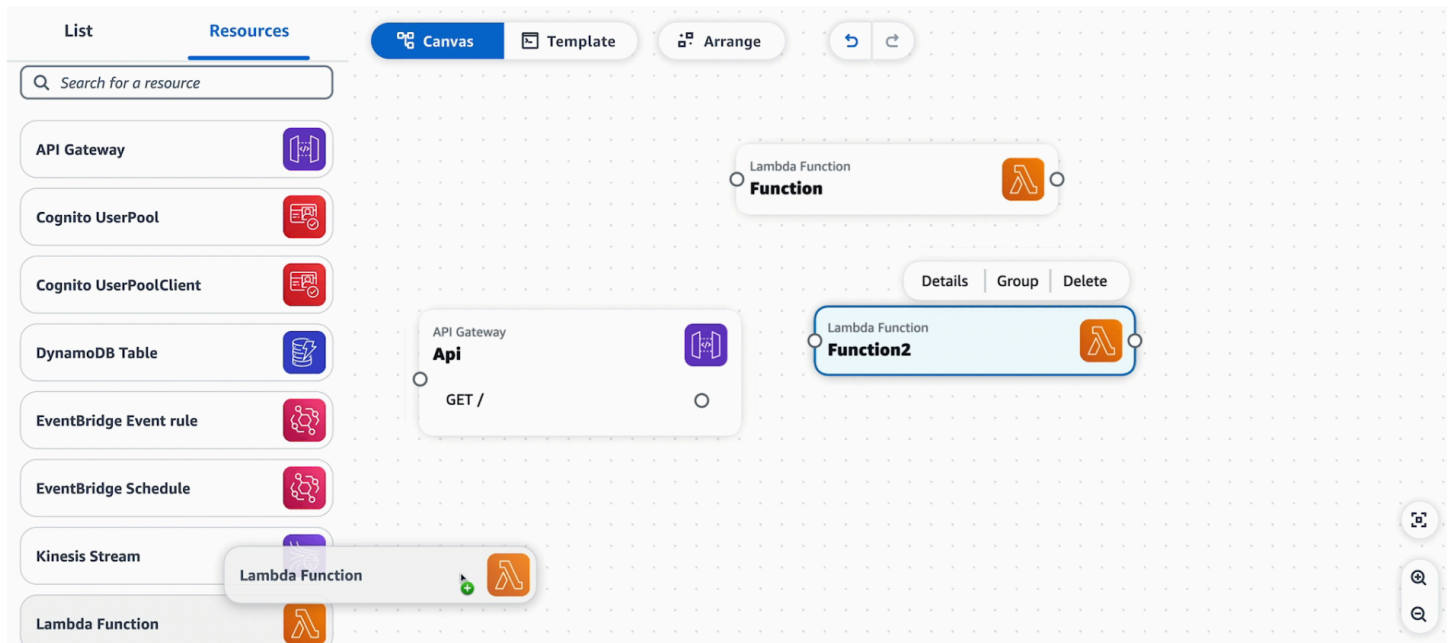
## Topik

- [Tempatkan kartu di kanvas visual Infrastructure Composer](#)
- [Kelompokkan kartu bersama di kanvas visual Infrastructure Composer](#)
- [Connect kartu pada kanvas visual Infrastructure Composer](#)
- [Putuskan sambungan kartu di Infrastructure Composer](#)
- [Atur kartu pada kanvas visual Infrastructure Composer](#)
- [Mengkonfigurasi dan memodifikasi kartu di Infrastructure Composer](#)
- [Hapus kartu di Infrastructure Composer](#)
- [Lihat pembaruan kode dengan Change Inspector di Infrastructure Composer](#)
- [Referensi file eksternal di Infrastructure Composer](#)
- [Integrasikan Komposer Infrastruktur dengan Amazon Virtual Private Cloud \(AmazonVPC\)](#)

## Tempatkan kartu di kanvas visual Infrastructure Composer

Bagian ini menjelaskan bagaimana Anda memilih dan menyeret [kartu](#) Infrastructure Composer di kanvas visualnya. Sebelum memulai, identifikasi sumber daya apa yang dibutuhkan aplikasi Anda dan bagaimana mereka perlu berinteraksi. Untuk tips melakukan ini, lihat [Bangun aplikasi pertama Anda dengan Infrastructure Composer](#).

Untuk menambahkan kartu ke aplikasi Anda, seret dari palet sumber daya dan jatuhkan ke kanvas visual.



Anda dapat memilih dari dua jenis kartu: Kartu [komponen yang disempurnakan](#) dan kartu [sumber daya Standar IAC](#).

Setelah menempatkan kartu Anda di kanvas visual, Anda akan siap untuk mengelompokkan, menghubungkan, mengatur, dan mengkonfigurasi kartu Anda. Lihat topik berikut untuk informasi tentang melakukan ini:

- [Kelompokkan kartu bersama di kanvas visual Infrastructure Composer](#)
- [Connect kartu pada kanvas visual Infrastructure Composer](#)
- [Atur kartu pada kanvas visual Infrastructure Composer](#)
- [Mengkonfigurasi dan memodifikasi kartu di Infrastructure Composer](#)

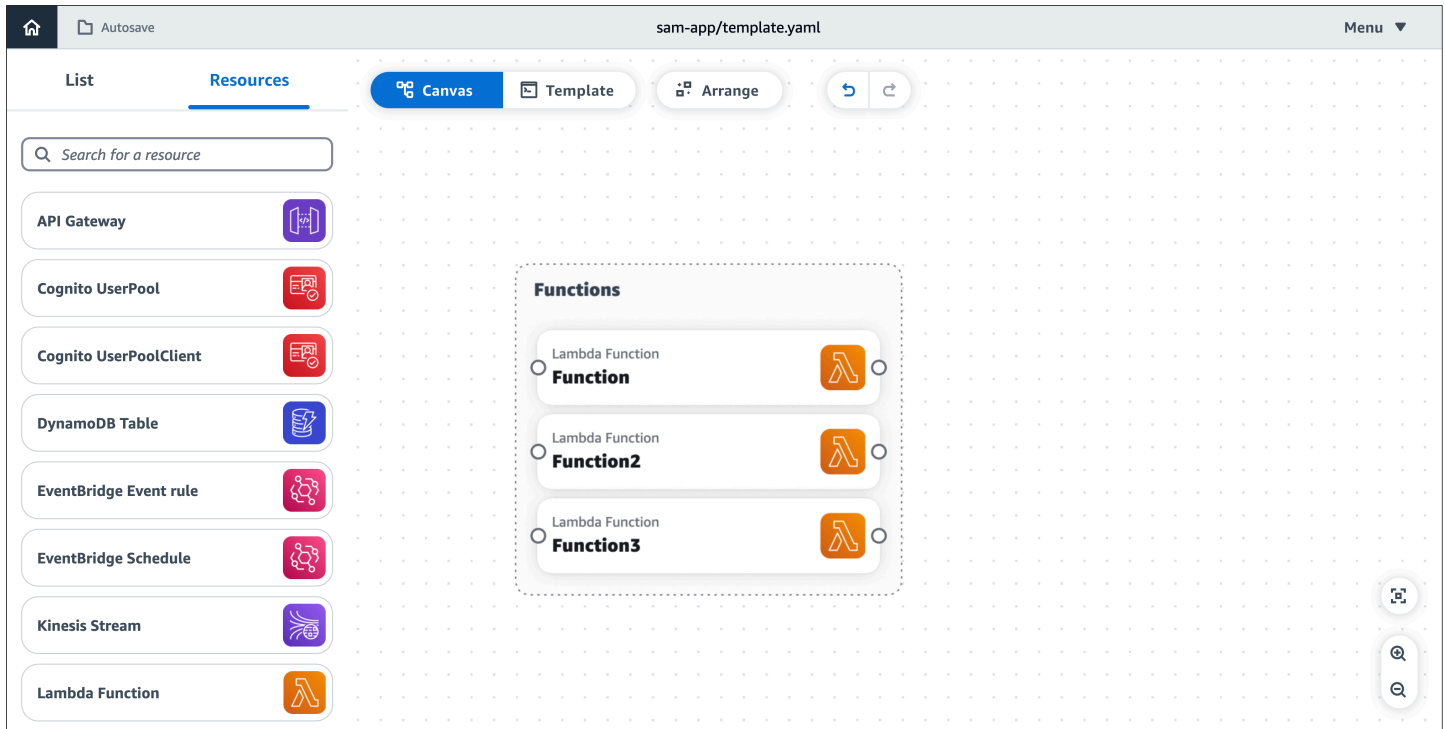
## Kelompokkan kartu bersama di kanvas visual Infrastructure Composer

Topik ini berisi detail tentang pengelompokan kartu komponen yang disempurnakan dan kartu komponen standar. Kartu pengelompokan membantu Anda mengkategorikan dan mengatur sumber daya Anda tanpa perlu memikirkan kode atau markup yang perlu Anda tulis.

### Mengelompokkan kartu komponen yang disempurnakan

Ada dua cara untuk mengelompokkan kartu komponen yang disempurnakan bersama-sama:

- Sambil menekan Shift, pilih kartu untuk dikelompokkan. Kemudian, pilih Grup dari menu tindakan sumber daya.
- pilih kartu yang Anda inginkan dalam grup. Dari menu yang muncul, pilih Grup. Ini akan membuat grup yang dapat Anda seret dan jatuhkan kartu lain.



## Mengelompokkan kartu komponen standar ke kartu lain

Contoh berikut menunjukkan satu cara kartu komponen standar dapat dikelompokkan ke kartu lain dari panel properti Resource:

The screenshot displays the AWS Infrastructure Composer interface. On the left is a canvas with a grid background. A 'Standard Component' card for 'Function' is placed on the canvas, containing sub-components 'Role' and 'Function'. On the right is the 'Resource properties' panel for an 'AWS::Lambda::Function' resource. The panel includes an 'Editing' dropdown set to 'Function', a 'Logical ID' dropdown set to 'Function', and a 'Resource configuration' section with a code editor containing 'Code: {}' and 'Role: !Ref Role'. A 'Resource reference' button is at the bottom right of the panel.

Di bidang konfigurasi Resource pada panel Resource properties, Role telah direferensikan dalam fungsi Lambda. Ini menghasilkan kartu Peran dikelompokkan ke dalam kartu Fungsi di kanvas.

## Connect kartu pada kanvas visual Infrastructure Composer

Gunakan topik ini untuk memahami cara menghubungkan kartu di Infrastructure Composer. Bagian ini mencakup detail tentang menghubungkan kartu komponen yang disempurnakan dan kartu



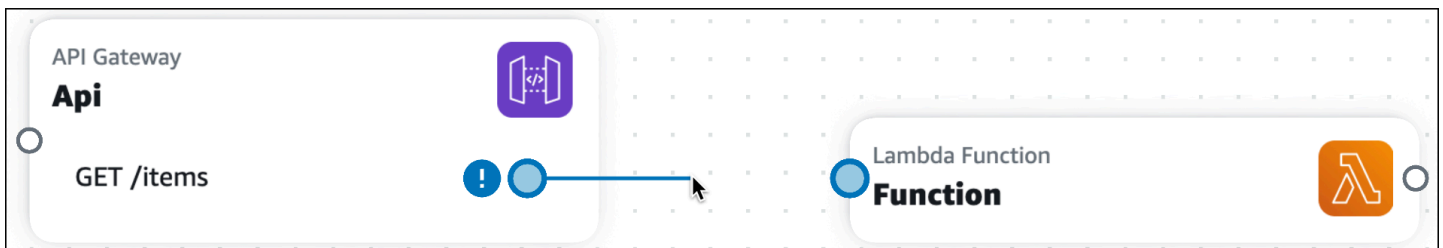
komponen standar. Ini juga memberikan beberapa contoh yang menggambarkan berbagai cara kartu dapat dihubungkan.

## Menghubungkan kartu komponen yang disempurnakan

Pada kartu komponen yang disempurnakan, port secara visual mengidentifikasi di mana koneksi dapat dibuat.

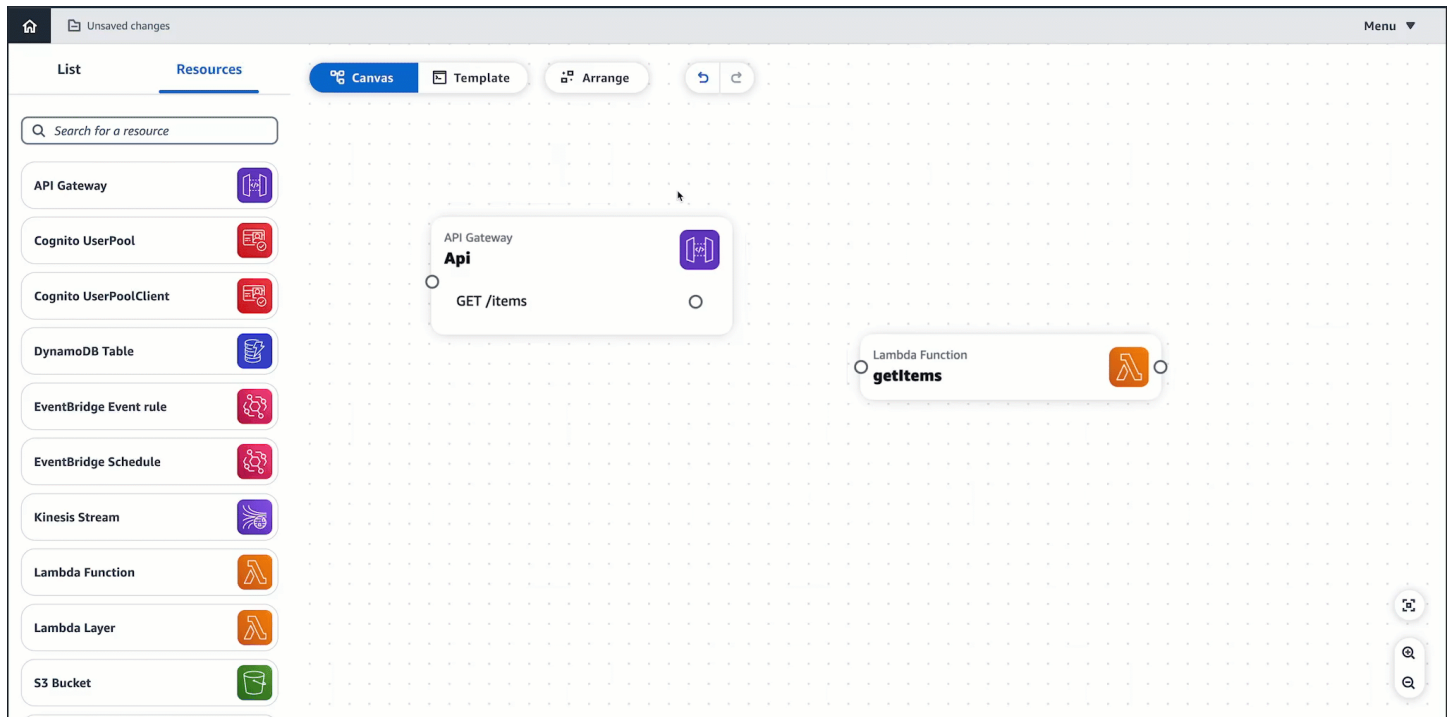
- Sebuah port di sisi kanan kartu menunjukkan kesempatan bagi kartu untuk memanggil kartu lain.
- Sebuah port di sisi kiri kartu menunjukkan kesempatan bagi kartu untuk dipanggil oleh kartu lain.

Connect kartu bersama-sama dengan mengklik pada port kanan dari satu kartu dan menyeretnya ke port kiri pada kartu lain.



Saat Anda membuat koneksi, sebuah pesan akan ditampilkan, memberi tahu Anda jika koneksi berhasil dibuat. Pilih pesan untuk melihat apa yang diubah Infrastructure Composer untuk menyediakan koneksi. Jika koneksi tidak berhasil, Anda dapat memilih tampilan Template untuk memperbarui kode infrastruktur secara manual untuk menyediakan koneksi.

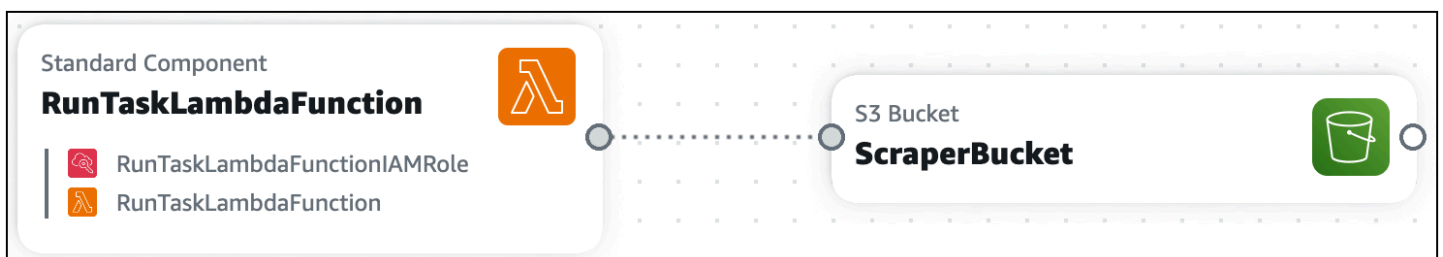
- Jika berhasil, klik pesan untuk melihat inspektur Ubah. Di sini, Anda dapat melihat Infrastructure Composer apa yang dimodifikasi untuk menyediakan koneksi Anda.
- Ketika tidak berhasil, sebuah pesan akan ditampilkan. Anda dapat memilih tampilan Template dan memperbarui kode infrastruktur secara manual untuk menyediakan koneksi.



Saat Anda menghubungkan kartu komponen yang disempurnakan bersama-sama, Infrastructure Composer secara otomatis membuat kode infrastruktur di template Anda untuk menyediakan hubungan berbasis peristiwa antara sumber daya Anda.

## Menghubungkan kartu komponen standar (kartu sumber daya standar IAC)

Kartu sumber daya IAC standar tidak termasuk port untuk membuat koneksi dengan sumber daya lain. Selama [konfigurasi kartu](#), Anda menentukan hubungan berbasis peristiwa dalam template aplikasi Anda, Infrastructure Composer akan secara otomatis mendeteksi koneksi ini dan memvisualisasikannya dengan garis putus-putus di antara kartu Anda. Berikut ini adalah contoh koneksi antara kartu komponen standar dan kartu komponen yang disempurnakan:



Contoh berikut menunjukkan bagaimana fungsi Lambda dapat dihubungkan dengan rest Amazon API Gateway: API

```
AWSTemplateFormatVersion: '2010-09-09'
```

```
Resources:
  MyApi:
    Type: 'AWS::ApiGateway::RestApi'
    Properties:
      Name: MyApi

  ApiGatewayMethod:
    Type: 'AWS::ApiGateway::Method'
    Properties:
      HttpMethod: POST # Specify the HTTP method you want to use (e.g., GET, POST,
PUT, DELETE)
      ResourceId: !GetAtt MyApi.RootResourceId
      RestApiId: !Ref MyApi
      AuthorizationType: NONE
      Integration:
        Type: AWS_PROXY
        IntegrationHttpMethod: POST
        Uri: !Sub
          - arn:aws:apigateway:${AWS::Region}:lambda:path/2015-03-31/functions/
${LambdaFunctionArn}/invocations
          - { LambdaFunctionArn: !GetAtt MyLambdaFunction.Arn }
      MethodResponses:
        - StatusCode: 200

  MyLambdaFunction:
    Type: 'AWS::Lambda::Function'
    Properties:
      Handler: index.handler
      Role: !GetAtt LambdaExecutionRole.Arn
      Runtime: nodejs14.x
      Code:
        S3Bucket: your-bucket-name
        S3Key: your-lambda-zip-file.zip

  LambdaExecutionRole:
    Type: 'AWS::IAM::Role'
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: lambda.amazonaws.com
            Action: 'sts:AssumeRole'
```

```

Policies:
  - PolicyName: LambdaExecutionPolicy
    PolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Effect: Allow
          Action:
            - 'logs:CreateLogGroup'
            - 'logs:CreateLogStream'
            - 'logs:PutLogEvents'
          Resource: 'arn:aws:logs:*:*:*'
        - Effect: Allow
          Action:
            - 'lambda:InvokeFunction'
          Resource: !GetAtt MyLambdaFunction.Arn

```

Dalam contoh di atas, cuplikan kode yang tercantum di `ApiGatewayMethod:` bawah `Integration:` menentukan hubungan event-driven yang menghubungkan dua kartu.

## Contoh untuk menghubungkan kartu di Infrastructure Composer

Gunakan contoh di bagian ini untuk memahami bagaimana kartu dapat dihubungkan di Infrastructure Composer.

### Memanggil AWS Lambda fungsi saat item ditempatkan di bucket Amazon Simple Storage Service (Amazon S3)

Dalam contoh ini, kartu bucket Amazon S3 terhubung ke kartu fungsi Lambda. Saat item ditempatkan di bucket Amazon S3, fungsi tersebut dipanggil. Fungsi tersebut kemudian dapat digunakan untuk memproses item atau memicu peristiwa lain dalam aplikasi Anda.



Interaksi ini mengharuskan suatu peristiwa didefinisikan untuk fungsi tersebut. Inilah ketentuan Infrastructure Composer:

```

Transform: AWS::Serverless-2016-10-31
...
Resources:

```

```

MyBucket:
  Type: AWS::S3::Bucket
  ...
MyBucketBucketPolicy:
  Type: AWS::S3::BucketPolicy
  ...
MyFunction:
  Type: AWS::Serverless::Function
  Properties:
    ...
  Events:
    MyBucket:
      Type: S3
      Properties:
        Bucket: !Ref MyBucket
      Events:
        - s3:ObjectCreated:* # Event that triggers invocation of function
        - s3:ObjectRemoved:* # Event that triggers invocation of function

```

## Memanggil bucket Amazon S3 dari fungsi Lambda

Dalam contoh ini, kartu fungsi Lambda memanggil kartu bucket Amazon S3. Fungsi Lambda dapat digunakan untuk melakukan CRUD operasi pada item di bucket Amazon S3.



Interaksi ini membutuhkan hal-hal berikut, yang disediakan oleh Infrastructure Composer:

- IAM kebijakan yang memungkinkan fungsi Lambda berinteraksi dengan bucket Amazon S3.
- Variabel lingkungan yang mempengaruhi perilaku fungsi Lambda.

```

Transform: AWS::Serverless-2016-10-31
...
Resources:
  MyBucket:
    Type: AWS::S3::Bucket
    ...
  MyBucketBucketPolicy:
    Type: AWS::S3::BucketPolicy

```

```

...
MyFunction:
  Type: AWS::Serverless::Function
  Properties:
    ...
    Environment:
      Variables:
        BUCKET_NAME: !Ref MyBucket
        BUCKET_ARN: !GetAtt MyBucket.Arn
    Policies:
      - S3CrudPolicy:
        BucketName: !Ref MyBucket

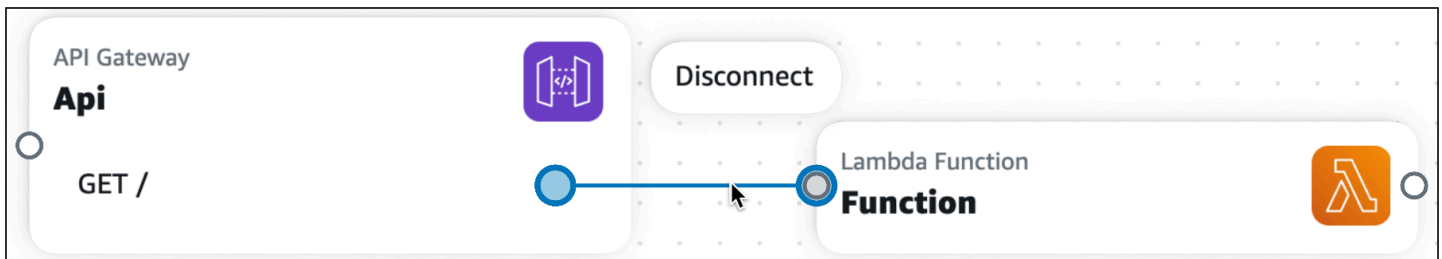
```

## Putuskan sambungan kartu di Infrastructure Composer

Di Infrastructure Composer, Anda menghubungkan dan memutuskan AWS sumber daya menggunakan kartu komponen yang disempurnakan dan kartu komponen standar. Bagian ini menjelaskan cara memutuskan kedua jenis kartu.

### Kartu komponen yang disempurnakan

Untuk memutuskan sambungan kartu komponen yang disempurnakan, pilih garis dan pilih Putuskan sambungan.



Infrastructure Composer akan secara otomatis memodifikasi template Anda untuk menghapus hubungan berbasis peristiwa dari aplikasi Anda.

### Kartu komponen standar

Kartu komponen standar tidak termasuk port untuk membuat koneksi dengan sumber daya lain. Selama [konfigurasi kartu](#), Anda menentukan hubungan berbasis peristiwa dalam template aplikasi Anda, Infrastructure Composer akan secara otomatis mendeteksi koneksi ini dan memvisualisasikannya dengan garis putus-putus di antara kartu Anda. Untuk memutuskan

sambungan kartu komponen standar, hapus hubungan yang digerakkan oleh peristiwa dalam templat aplikasi Anda.

Contoh berikut menunjukkan fungsi Lambda yang terhubung dengan sisa Amazon API Gateway: API

```

AWSTemplateFormatVersion: '2010-09-09'
Resources:
  MyApi:
    Type: 'AWS::ApiGateway::RestApi'
    Properties:
      Name: MyApi

  ApiGatewayMethod:
    Type: 'AWS::ApiGateway::Method'
    Properties:
      HttpMethod: POST # Specify the HTTP method you want to use (e.g., GET, POST,
PUT, DELETE)
      ResourceId: !GetAtt MyApi.RootResourceId
      RestApiId: !Ref MyApi
      AuthorizationType: NONE
      Integration:
        Type: AWS_PROXY
        IntegrationHttpMethod: POST
        Uri: !Sub
          - arn:aws:apigateway:${AWS::Region}:lambda:path/2015-03-31/functions/
${LambdaFunctionArn}/invocations
          - { LambdaFunctionArn: !GetAtt MyLambdaFunction.Arn }
      MethodResponses:
        - StatusCode: 200

  MyLambdaFunction:
    Type: 'AWS::Lambda::Function'
    Properties:
      Handler: index.handler
      Role: !GetAtt LambdaExecutionRole.Arn
      Runtime: nodejs14.x
      Code:
        S3Bucket: your-bucket-name
        S3Key: your-lambda-zip-file.zip

  LambdaExecutionRole:
    Type: 'AWS::IAM::Role'
    Properties:
      AssumeRolePolicyDocument:

```

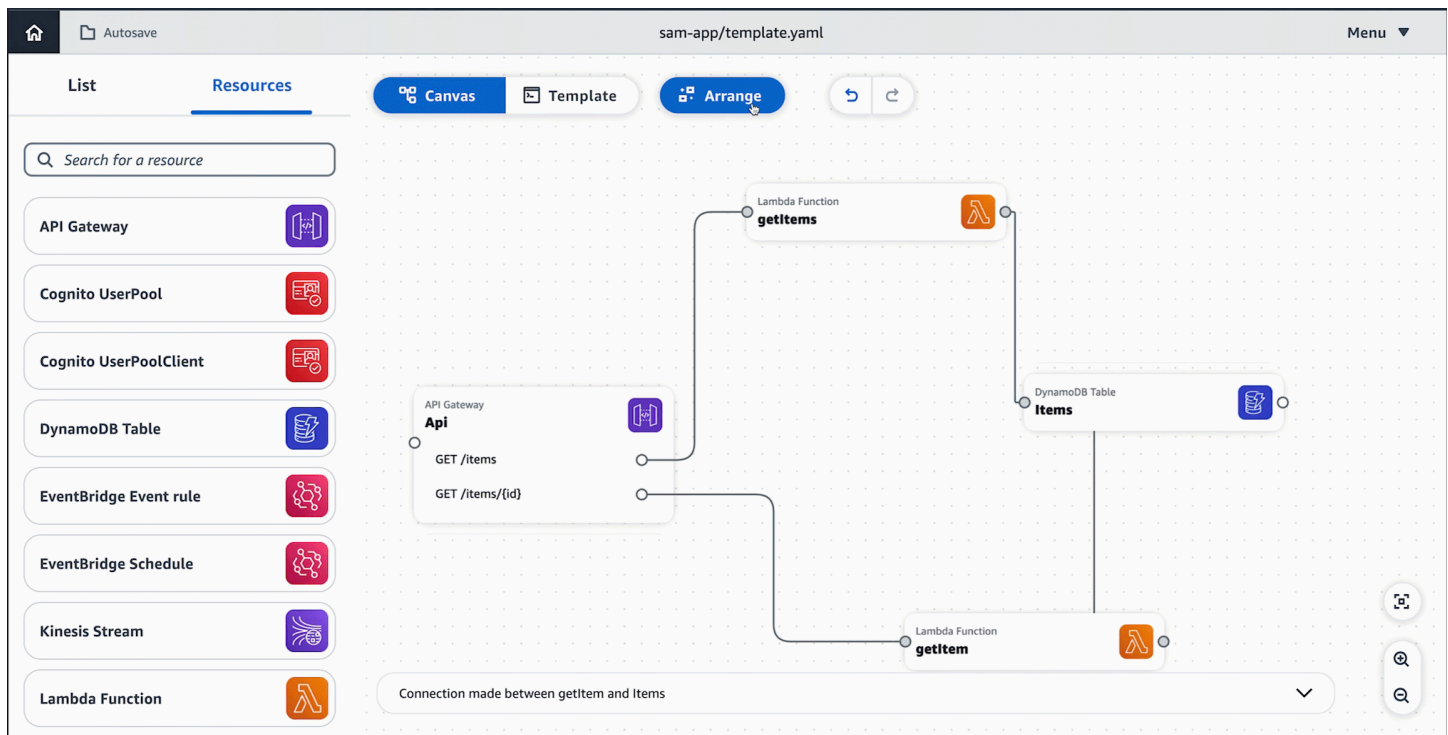
```
Version: '2012-10-17'
Statement:
  - Effect: Allow
    Principal:
      Service: lambda.amazonaws.com
    Action: 'sts:AssumeRole'
Policies:
  - PolicyName: LambdaExecutionPolicy
    PolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Effect: Allow
          Action:
            - 'logs:CreateLogGroup'
            - 'logs:CreateLogStream'
            - 'logs:PutLogEvents'
          Resource: 'arn:aws:logs:*:*:*'
        - Effect: Allow
          Action:
            - 'lambda:InvokeFunction'
          Resource: !GetAtt MyLambdaFunction.Arn
```

Untuk menghapus koneksi antara dua kartu, hapus referensi yang `MyLambdaFunction` tercantum di `ApiGatewayMethod`: bawah. `Integration`

## Atur kartu pada kanvas visual Infrastructure Composer

Pilih `Atur` untuk mengatur dan mengatur kartu secara visual di kanvas. Menggunakan tombol `Atur` sangat berguna ketika ada banyak kartu dan koneksi di kanvas.





## Mengkonfigurasi dan memodifikasi kartu di Infrastructure Composer

Di Infrastructure Composer, kartu mewakili sumber daya yang Anda gunakan untuk mendesain arsitektur aplikasi Anda. Saat Anda mengonfigurasi kartu di Infrastructure Composer, Anda menentukan detail sumber daya dalam aplikasi Anda. Ini termasuk rincian seperti ID Logis kartu dan kunci partisi. Cara informasi ini didefinisikan bervariasi antara kartu komponen yang disempurnakan dan kartu Standar.

Kartu komponen yang disempurnakan adalah Kumpulan AWS CloudFormation sumber daya yang telah digabungkan menjadi satu kartu kurasi yang meningkatkan kemudahan penggunaan, fungsionalitas, dan dirancang untuk berbagai kasus penggunaan. Kartu sumber daya Standar IAC mewakili satu AWS CloudFormation sumber daya. Setiap kartu sumber daya IAC standar, setelah diseret ke kanvas, diberi label komponen Standar.

Topik ini memberikan detail tentang mengonfigurasi kartu komponen yang Ditingkatkan dan kartu komponen Standar.

### Note

Topik ini berlaku untuk menggunakan kartu dari Infrastructure Composer Console, AWS Toolkit for Visual Studio Code ekstensi, dan saat berada di Infrastructure Composer dalam

mode CloudFormation konsol. Kartu terkait Lambda (Fungsi Lambda dan Lapisan Lambda) memerlukan pembuatan kode dan solusi pengemasan yang tidak tersedia di Infrastructure Composer dalam mode konsol. CloudFormation Untuk informasi selengkapnya, lihat [Menggunakan Infrastructure Composer dalam mode CloudFormation konsol](#).

## Topik

- [Kartu komponen yang disempurnakan di Infrastructure Composer](#)
- [Kartu standar di Infrastructure Composer](#)

## Kartu komponen yang disempurnakan di Infrastructure Composer

Untuk mengkonfigurasi kartu komponen yang disempurnakan, Infrastructure Composer menyediakan formulir di panel properti Resource. Formulir ini dikuratori secara unik untuk memandu Anda mengonfigurasi setiap kartu komponen yang disempurnakan. Saat Anda mengisi formulir, Infrastructure Composer memodifikasi kode infrastruktur Anda.

Beberapa kartu komponen yang disempurnakan memang memiliki fitur tambahan. Bagian ini mengulas dasar-dasar penggunaan kartu komponen yang disempurnakan dan menawarkan detail tentang kartu dengan fitur tambahan.

Untuk informasi lebih lanjut tentang kartu komponen yang disempurnakan, lihat [Kartu komponen yang disempurnakan di Infrastructure Composer](#) dan [Kartu komponen yang disempurnakan di Infrastructure Composer](#)

## Prosedur

Panel properti Resource merampingkan konfigurasi dan menambahkan guiderail yang menyederhanakan konfigurasi kartu. Untuk menggunakan panel ini, lakukan langkah-langkah berikut:

1. Klik dua kali kartu untuk memunculkan panel Resource properties.
2. Klik pada kartu dan pilih Detail untuk membuka panel properti sumber daya.
3. Untuk Infrastructure Composer dari AWS Management Console, pilih Template untuk menampilkan kode aplikasi Anda. Konfigurasi langsung dari sini.

Gambar berikut menunjukkan bagaimana hal ini dapat dilakukan:

## Menggunakan Infrastructure Composer dengan Amazon Relational Database Service (Amazon) RDS

AWS Infrastructure Composer fitur integrasi dengan Amazon Relational Database Service (RDSAmazon). Menggunakan kartu komponen yang ditingkatkan RDSDatabase (Eksternal) di Infrastructure Composer, Anda dapat menghubungkan aplikasi Anda ke Amazon RDS DB cluster, instance, dan proxy yang didefinisikan pada template lain AWS CloudFormation atau AWS Serverless Application Model (AWS SAM)

Kartu komponen yang ditingkatkan RDSDatabase (Eksternal) mewakili RDS sumber daya Amazon yang ditentukan pada templat lain. Hal ini mencakup:

- Amazon RDS DB cluster atau instance yang didefinisikan pada template lain
- Amazon RDS DB proxy

Kartu komponen yang ditingkatkan RDSDatabase (Eksternal) tersedia dari palet Sumber Daya.



Untuk menggunakan kartu ini, seret ke kanvas Infrastructure Composer, konfigurasi, dan hubungkan ke sumber daya lain.

Anda dapat menghubungkan aplikasi Anda ke Amazon eksternal RDS DB cluster atau instance melalui fungsi Lambda.

## Persyaratan

Untuk menggunakan fitur ini, Anda harus memenuhi persyaratan berikut:

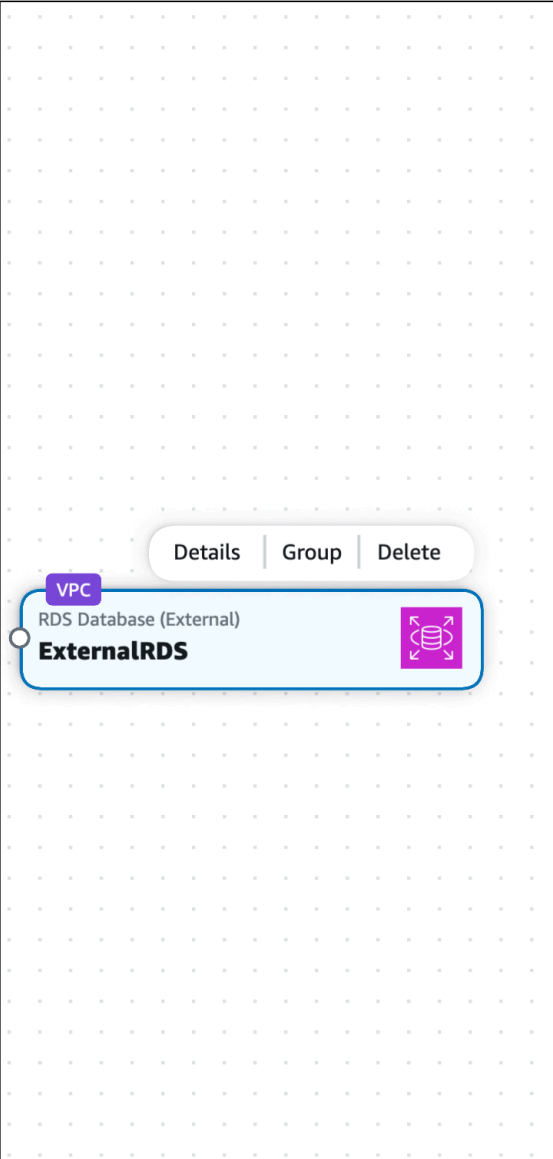
1. Amazon eksternal Anda RDS DB cluster, instance, atau proxy harus digunakan AWS Secrets Manager untuk mengelola kata sandi pengguna. Untuk mempelajari selengkapnya, lihat [Manajemen kata sandi dengan Amazon RDS dan AWS Secrets Manager](#) di Panduan RDS Pengguna Amazon.
2. Aplikasi Anda di Infrastructure Composer harus merupakan proyek baru atau harus awalnya dibuat di Infrastructure Composer.

## Prosedur

### Langkah 1: Konfigurasi kartu RDS Database eksternal

Dari palet Resources, seret kartu komponen yang ditingkatkan RDSDatabase (eksternal) ke kanvas.

Pilih kartu dan pilih Detail atau klik dua kali pada kartu untuk membuka panel Resource properties. Panel properti sumber daya kartu akan muncul:



## RDS Database (External)

RDS database cluster or instance defined outside of the template. This card will create 3 stack parameters by default. Specify values in this form or at deployment time. You can use “!ImportValue” or SSM with dynamic reference if value is stored elsewhere.

---

**Logical ID**

A unique name for your RDS database. This value will be used for environment variables and parameters in your template.

ExternalRDS

**Database Secret**

Secrets Manager secret to fetch database credentials. This field creates a stack parameter with name {Logical ID + SecretArn}.

**Database Hostname**

Hostname to connect to the RDS DB cluster or instance. For RDS Proxy, use the Proxy endpoint. This field creates a stack parameter with name {Logical ID + Hostname}.

**Database Port**

Port to connect to the RDS DB cluster or instance. This field creates a stack parameter with name {Logical ID + Port}.

Anda dapat mengonfigurasi yang berikut ini di sini:

- Logical ID - Nama unik untuk Amazon eksternal Anda RDS DB cluster, instance, atau proxy. ID ini tidak harus cocok dengan nilai ID logis Amazon eksternal Anda RDS DB sumber daya.
- Rahasia basis data — Pengidentifikasi AWS Secrets Manager rahasia yang terkait dengan Amazon Anda RDS DB cluster, instance, atau proxy. Bidang ini menerima nilai-nilai berikut:
  - Nilai statis — Pengidentifikasi unik dari rahasia database, seperti rahasiaARN. Berikut ini adalah contohnya: `arn:aws:secretsmanager:us-west-2:123456789012:secret:my-path/my-secret-name-1a2b3c`. Untuk informasi selengkapnya, lihat [AWS Secrets Manager konsep](#) di Panduan AWS Secrets Manager Pengguna.

- Nilai keluaran — Ketika rahasia Secrets Manager dikerahkan ke AWS CloudFormation, nilai output dibuat. Anda dapat menentukan nilai output di sini menggunakan fungsi [Fn::ImportValue](#) intrinsik. Misalnya, `!ImportValue MySecret`.
- Nilai dari SSM Parameter Store — Anda dapat menyimpan rahasia Anda di SSM Parameter Store dan menentukan nilainya menggunakan referensi dinamis. Misalnya, `{{resolve:ssm:MySecret}}`. Untuk informasi selengkapnya, lihat [SSMparameter](#) di Panduan AWS CloudFormation Pengguna.
- Nama host database - Nama host yang dapat digunakan untuk terhubung ke Amazon Anda RDS DB cluster, instance, atau proxy. Nilai ini ditentukan dalam templat eksternal yang mendefinisikan RDS sumber daya Amazon Anda. Nilai-nilai berikut diterima:
  - Nilai statis — Pengidentifikasi unik dari nama host database, seperti alamat titik akhir. Berikut ini adalah contohnya: `mystack-mydb-1apw1j4phy1rk.cg034hpkmjt.us-east-2.rds.amazonaws.com`.
  - Nilai keluaran - Nilai output dari Amazon yang dikerahkan RDS DB cluster, instance, atau proxy. Anda dapat menentukan nilai output menggunakan fungsi [Fn::ImportValue](#) intrinsik. Misalnya, `!ImportValue myStack-myDatabase-abcd1234`.
  - Nilai dari SSM Parameter Store - Anda dapat menyimpan nama host database di SSM Parameter Store dan menentukan nilainya menggunakan referensi dinamis. Misalnya, `{{resolve:ssm:MyDatabase}}`.
- Database port — Nomor port yang dapat digunakan untuk terhubung ke Amazon Anda RDS DB cluster, instance, atau proxy. Nilai ini ditentukan dalam templat eksternal yang mendefinisikan RDS sumber daya Amazon Anda. Nilai-nilai berikut diterima:
  - Nilai statis — Port database. Misalnya, `3306`.
  - Nilai keluaran - Nilai output dari Amazon yang dikerahkan RDS DB cluster, instance, atau proxy. Misalnya, `!ImportValue myStack-MyRDSInstancePort`.
  - Nilai dari SSM Parameter Store - Anda dapat menyimpan nama host database di SSM Parameter Store dan menentukan nilainya menggunakan referensi dinamis. Misalnya, `{{resolve:ssm:MyRDSInstancePort}}`.

#### Note

Hanya nilai ID logis yang harus dikonfigurasi di sini. Anda dapat mengonfigurasi properti lain pada waktu penerapan jika Anda mau.

## Langkah 2: Hubungkan kartu Fungsi Lambda

Dari palet Resources, seret kartu komponen yang disempurnakan Fungsi Lambda ke kanvas.

Hubungkan port kiri kartu Fungsi Lambda ke port kanan kartu RDSDatabase (eksternal).



Infrastructure Composer akan menyediakan template Anda untuk memfasilitasi koneksi ini.

Apa yang dilakukan Infrastructure Composer untuk membuat koneksi Anda

Ketika Anda menyelesaikan prosedur yang tercantum di atas, Infrastructure Composer melakukan tindakan spesifik untuk menghubungkan fungsi Lambda Anda ke database Anda.

Saat menentukan Amazon eksternal RDS DB cluster, contoh, atau proxy

Saat Anda menyeret kartu RDSDatabase (eksternal) ke kanvas, Infrastructure Composer memperbarui Metadata dan Parameters bagian template Anda sesuai kebutuhan. Berikut adalah contohnya:

```
Metadata:
  AWS::Composer::ExternalResources:
    ExternalRDS:
      Type: externalRDS
      Settings:
        Port: !Ref ExternalRDSPort
        Hostname: !Ref ExternalRDSHostname
        SecretArn: !Ref ExternalRDSSecretArn
Parameters:
  ExternalRDSPort:
    Type: Number
  ExternalRDSHostname:
    Type: String
  ExternalRDSSecretArn:
    Type: String
```

[Metadata](#) adalah bagian AWS CloudFormation template yang digunakan untuk menyimpan detail tentang template Anda. Metadata yang khusus untuk Infrastructure Composer disimpan di bawah kunci metadata. `AWS::Composer::ExternalResources` Di sini, Infrastructure Composer menyimpan nilai yang Anda tentukan untuk Amazon Anda RDS DB cluster, instance, atau proxy.

Bagian [Parameter](#) AWS CloudFormation template digunakan untuk menyimpan nilai kustom yang dapat disisipkan di seluruh template Anda saat penerapan. Bergantung pada jenis nilai yang Anda berikan, Infrastructure Composer dapat menyimpan nilai di sini untuk Amazon RDS DB cluster, instance, atau proxy dan tentukan di seluruh template Anda.

Nilai string di Parameters bagian Metadata dan menggunakan nilai ID logis yang Anda tentukan pada kartu RDSDatabase (eksternal) Anda. Jika Anda memperbarui ID logis, nilai string akan berubah.

Saat menghubungkan fungsi Lambda ke database Anda

Saat Anda menghubungkan kartu Fungsi Lambda ke kartu RDSDatabase (eksternal), Infrastructure Composer menyediakan variabel lingkungan dan AWS Identity and Access Management (IAM) kebijakan. Berikut adalah contohnya:

```
Resources:
  Function:
    Type: AWS::Serverless::Function
    Properties:
      ...
    Environment:
      Variables:
        EXTERNALRDS_PORT: !Ref ExternalRDSPort
        EXTERNALRDS_HOSTNAME: !Ref ExternalRDSHostname
        EXTERNALRDS_SECRETARN: !Ref ExternalRDSecretArn
    Policies:
      - AWSSecretsManagerGetSecretValuePolicy:
        SecretArn: !Ref ExternalRDSecretArn
```

Variabel [lingkungan](#) adalah variabel yang dapat digunakan oleh fungsi Anda saat runtime. Untuk mempelajari selengkapnya, lihat [Menggunakan variabel lingkungan Lambda](#) di Panduan AWS Lambda Pengembang.

[Kebijakan](#) menyediakan izin untuk fungsi Anda. Di sini, Infrastructure Composer membuat kebijakan untuk mengizinkan akses baca dari fungsi Anda ke Secrets Manager untuk mendapatkan kata sandi Anda untuk akses ke Amazon RDS DB cluster, instance, atau proxy.

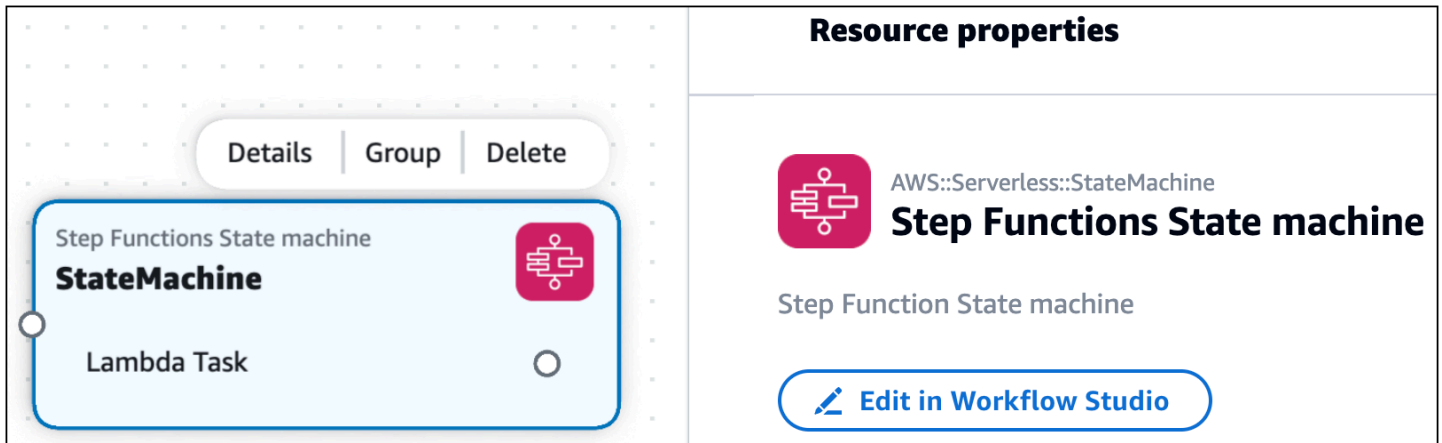
## Menggunakan AWS Infrastructure Composer dengan AWS Step Functions

AWS Infrastructure Composer fitur integrasi dengan [AWS Step Functions Workflow Studio](#). Gunakan Infrastructure Composer untuk melakukan hal berikut:



- Luncurkan Step Functions Workflow Studio langsung di dalam Komposer Infrastruktur.
- Buat dan kelola alur kerja baru atau impor alur kerja yang ada ke Infrastructure Composer.
- Integrasikan alur kerja Anda dengan AWS sumber daya lain menggunakan kanvas Infrastructure Composer.

Gambar berikut adalah kartu mesin Step Functions State



Dengan Step Functions Workflow Studio di Infrastructure Composer, Anda dapat menggunakan manfaat dari dua desainer visual yang kuat di satu tempat. Saat Anda mendesain alur kerja dan aplikasi Anda, Infrastructure Composer membuat infrastruktur Anda sebagai kode (IaC) untuk memandu Anda menuju penerapan.

## Topik

- [Kebijakan IAM](#)
- [Memulai dengan Step Functions Workflow Studio di Infrastructure Composer](#)
- [Menggunakan Step Functions Workflow Studio di Infrastructure Composer](#)
- [Pelajari selengkapnya](#)

## Kebijakan IAM

Saat Anda menghubungkan tugas dari alur kerja ke sumber daya, Infrastructure Composer secara otomatis membuat kebijakan AWS Identity and Access Management (IAM) yang diperlukan untuk mengotorisasi interaksi antar sumber daya Anda. Berikut adalah contohnya:

```
Transform: AWS::Serverless-2016-10-31
Resources:
  StockTradingStateMachine:
```

```

Type: AWS::Serverless::StateMachine
Properties:
  ...
Policies:
  - LambdaInvokePolicy:
      FunctionName: !Ref CheckStockValue
  ...
CheckStockValue:
  Type: AWS::Serverless::Function
  ...

```

Jika perlu, Anda dapat menambahkan lebih banyak IAM kebijakan ke template Anda.

Memulai dengan Step Functions Workflow Studio di Infrastructure Composer

Untuk memulai, Anda dapat membuat alur kerja baru atau mengimpor alur kerja yang ada.

Untuk membuat alur kerja baru

1. Dari palet Resources, seret kartu komponen yang disempurnakan mesin Step Functions State ke kanvas.



Saat Anda menyeret kartu mesin Step Functions State ke kanvas, Infrastructure Composer akan membuat yang berikut:

- [AWS::Serverless::StateMachine](#) Sumber daya yang mendefinisikan mesin negara Anda. Secara default, Infrastructure Composer menciptakan alur kerja standar. Untuk membuat alur kerja ekspres, ubah Type nilai dalam template Anda dari STANDARD keEXPRESS.
  - [AWS::Logs::LogGroup](#) Sumber daya yang mendefinisikan grup CloudWatch log Amazon untuk mesin status Anda.
2. Buka panel Resource properties kartu dan pilih Edit di Workflow Studio untuk membuka Workflow Studio dalam Komposer Infrastruktur.

Step Functions Workflow Studio terbuka dalam mode Desain. Untuk mempelajari lebih lanjut, lihat [Mode desain](#) di Panduan AWS Step Functions Pengembang.

**Note**

Anda dapat memodifikasi Infrastructure Composer untuk menyimpan definisi mesin status Anda dalam file eksternal. Untuk mempelajari selengkapnya, lihat [Bekerja dengan file eksternal](#).

3. Buat alur kerja Anda dan pilih Simpan. Untuk keluar Workflow Studio, pilih Kembali ke Komposer Infrastruktur.

Infrastructure Composer mendefinisikan alur kerja Anda menggunakan Definition properti sumber daya `AWS::Serverless::StateMachine`

4. Anda dapat mengubah alur kerja Anda dengan melakukan salah satu hal berikut:
  - Buka Workflow Studio lagi dan memodifikasi alur kerja Anda.
  - Untuk Infrastructure Composer dari konsol, Anda dapat membuka tampilan Template aplikasi Anda dan memodifikasi template Anda. Jika menggunakan sinkronisasi lokal, Anda dapat mengubah alur kerja di lokal IDE Anda. Infrastructure Composer akan mendeteksi perubahan Anda dan memperbarui alur kerja Anda di Infrastructure Composer.
  - Untuk Infrastructure Composer dari Toolkit for VS Code, Anda dapat langsung memodifikasi template Anda. Infrastructure Composer akan mendeteksi perubahan Anda dan memperbarui alur kerja Anda di Infrastructure Composer.

Untuk mengimpor alur kerja yang ada

Anda dapat mengimpor alur kerja dari aplikasi yang didefinisikan menggunakan AWS Serverless Application Model (AWS SAM) template. Gunakan mesin status apa pun yang ditentukan dengan jenis `AWS::Serverless::StateMachine` sumber daya, dan itu akan memvisualisasikan sebagai kartu komponen yang disempurnakan mesin Step Functions State yang dapat Anda gunakan untuk meluncurkan Workflow Studio.

Sumber `AWS::Serverless::StateMachine` daya dapat menentukan alur kerja menggunakan salah satu properti berikut:

- [Definition](#)— Alur kerja didefinisikan dalam AWS SAM template sebagai objek.
- [DefinitionUri](#)— Alur kerja didefinisikan pada file eksternal menggunakan [Amazon States Language](#). Jalur lokal file kemudian ditentukan dengan properti ini.

## Definisi properti

### Komposer Infrastruktur dari konsol

Untuk alur kerja yang ditentukan menggunakan `Definition` properti, Anda dapat mengimpor satu templat atau seluruh proyek.

- **Template** — Untuk petunjuk tentang mengimpor template, lihat [Impor template proyek yang ada di konsol Infrastructure Composer](#). Untuk menyimpan perubahan yang Anda buat dalam Infrastructure Composer, Anda harus mengekspor template Anda.
- **Proyek** — Saat Anda mengimpor proyek, Anda harus mengaktifkan sinkronisasi lokal. Perubahan yang Anda buat secara otomatis disimpan ke mesin lokal Anda. Untuk petunjuk tentang mengimpor proyek, lihat [Impor folder proyek yang ada di konsol Infrastructure Composer](#).

### Komposer Infrastruktur dari Toolkit for VS Code

Untuk alur kerja yang ditentukan menggunakan `Definition` properti, Anda dapat membuka Infrastructure Composer dari template Anda. Untuk petunjuk, silakan lihat [Akses Komposer Infrastruktur dari AWS Toolkit for Visual Studio Code](#).

## DefinitionUri properti

### Komposer Infrastruktur dari konsol

Untuk alur kerja yang ditentukan menggunakan `DefinitionUri` properti, Anda harus mengimpor proyek dan mengaktifkan sinkronisasi lokal. Untuk petunjuk tentang mengimpor proyek, lihat [Impor folder proyek yang ada di konsol Infrastructure Composer](#).

### Komposer Infrastruktur dari Toolkit for VS Code

Untuk alur kerja yang ditentukan menggunakan `DefinitionUri` properti, Anda dapat membuka Infrastructure Composer dari template Anda. Untuk petunjuk, silakan lihat [Akses Komposer Infrastruktur dari AWS Toolkit for Visual Studio Code](#).

## Menggunakan Step Functions Workflow Studio di Infrastructure Composer

### Membangun alur kerja

Infrastructure Composer menggunakan substitusi definisi untuk memetakan tugas alur kerja ke sumber daya dalam aplikasi Anda. Untuk mempelajari lebih lanjut tentang substitusi definisi, lihat [DefinitionSubstitutions](#) di Panduan AWS Serverless Application Model Pengembang.

Saat Anda membuat tugas di Workflow Studio, tentukan substitusi definisi untuk setiap tugas. Anda kemudian dapat menghubungkan tugas ke sumber daya di kanvas Infrastructure Composer.

Untuk menentukan substitusi definisi di Workflow Studio

1. Buka tab Konfigurasi tugas dan cari bidang APIParameter.

The screenshot displays the AWS Step Functions Workflow Studio interface. On the left, a workflow diagram is visible, starting with a 'Start' node, followed by a 'Lambda: Invoke Check Stock Value' task, a 'Choice state Choice' state with a condition '\$.stock\_price <= 50', two 'Lambda: Invoke Buy Stock' and 'Lambda: Invoke Sell Stock' tasks, and a 'DynamoDB: PutItem Record Transaction' task leading to an 'End' node. On the right, the configuration panel for the 'Check Stock Value' task is shown, with tabs for Configuration, Input, Output, and Error handling. The Configuration tab is active, showing fields for State name (Check Stock Value), API (Lambda: Invoke), Integration type (Optimized), and API Parameters (Function name: Enter a CloudFormation substitution, with a dropdown menu showing `${LambdaFunction1}`).

2. Jika bidang APIParameter memiliki opsi drop-down, pilih Masukkan AWS CloudFormation substitusi. Kemudian, berikan nama yang unik.

Untuk tugas yang terhubung ke sumber daya yang sama, tentukan substitusi definisi yang sama untuk setiap tugas. Untuk menggunakan substitusi definisi yang ada, pilih Pilih AWS CloudFormation substitusi dan pilih substitusi yang akan digunakan.

3. Jika bidang APIParameter berisi JSON objek, ubah entri yang menentukan nama sumber daya untuk menggunakan substitusi definisi. Dalam contoh berikut, kita ubah "MyDynamoDBTable" menjadi "\${RecordTransaction}".

The screenshot displays the AWS Step Functions console for a state machine named "Record Transaction". On the left, a state machine diagram shows the following flow:

- Start** (yellow circle) leads to **Lambda: Invoke Check Stock Value** (orange box).
- Lambda: Invoke Check Stock Value** leads to **Choice state Choice** (white box with a question mark icon).
- The **Choice state Choice** has two paths:
  - Condition:** `$.stock_price <= 50` leads to **Lambda: Invoke Buy Stock** (orange box).
  - Default:** leads to **Lambda: Invoke Sell Stock** (orange box).
- Both **Lambda: Invoke Buy Stock** and **Lambda: Invoke Sell Stock** lead to **DynamoDB: PutItem Record Transaction** (blue box).
- DynamoDB: PutItem Record Transaction** leads to **End** (yellow circle).

On the right, the configuration panel for the "Record Transaction" state is shown:

- State name:** Record Transaction
- API:** DynamoDB: PutItem
- Integration type:** Optimized
- API Parameters:**

```

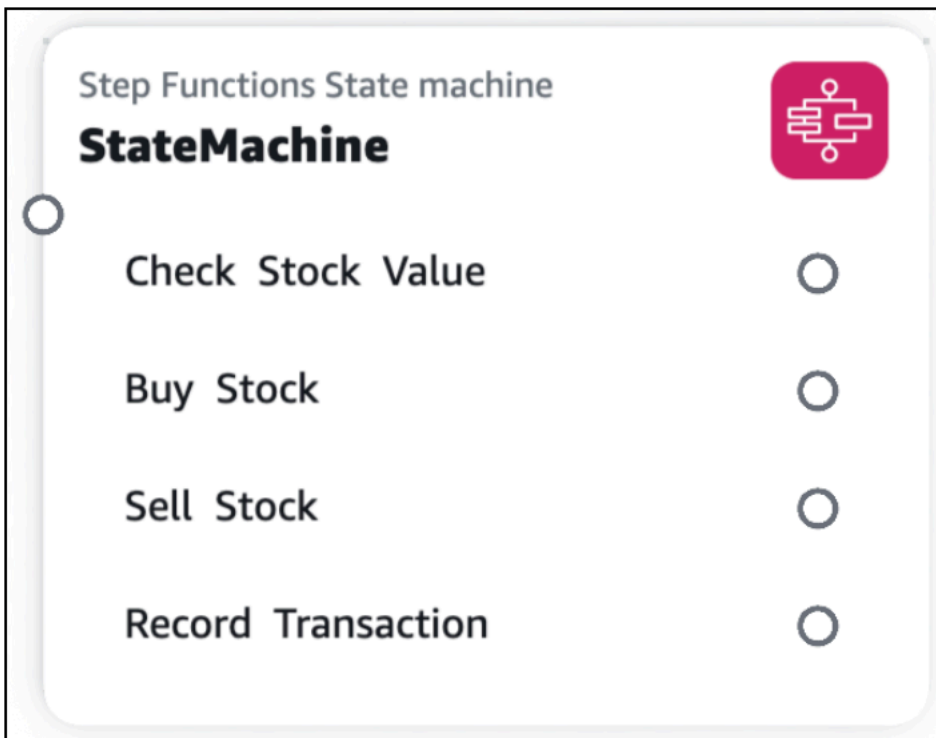
1 {
2   "TableName": "${RecordTransaction}",
3   "Item": {
4     "Column": {
5       "S": "MyEntry"
6     }
7   }

```

Below the JSON editor, a note states: "Must be valid JSON. To reference a node in this state's JSON input, the key must end with ".\$" (for example "key2.\$": "\$inputValue"). [Info](#)

4. Pilih Simpan dan Kembali ke Komposer Infrastruktur.

Tugas dari alur kerja Anda akan divisualisasikan pada kartu mesin Step Functions State.



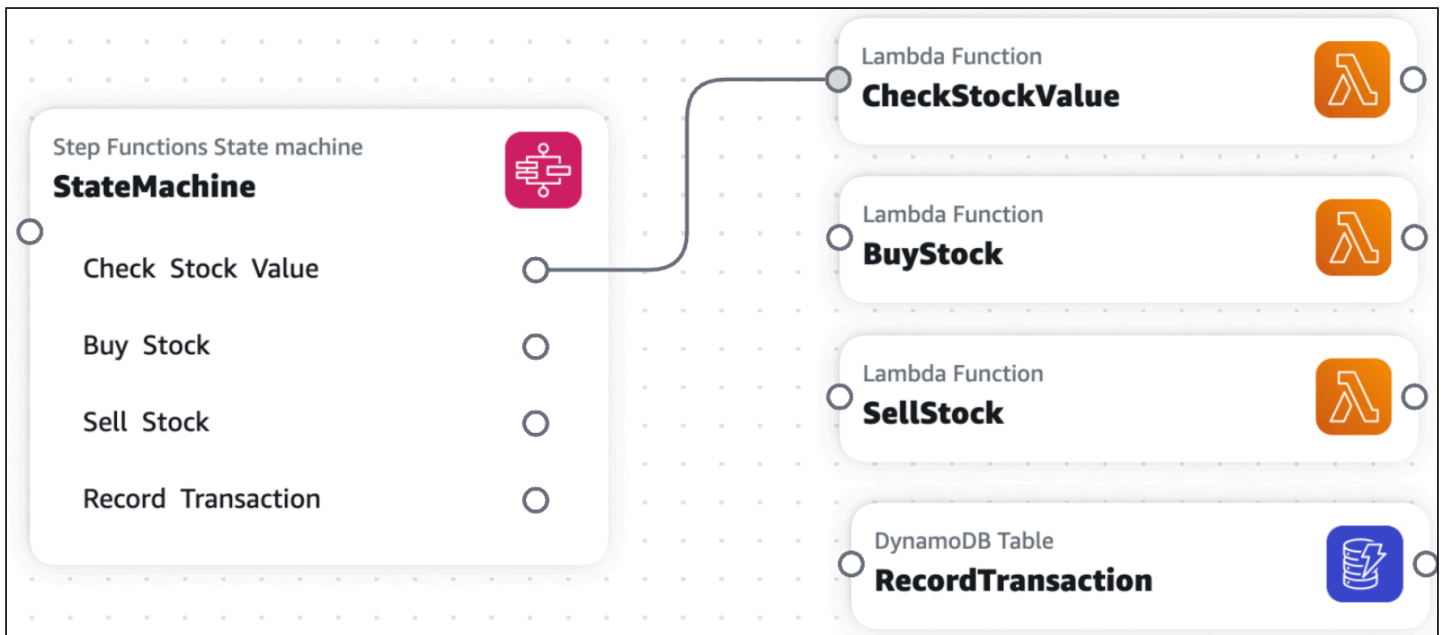
Hubungkan sumber daya ke tugas alur kerja

Anda dapat membuat koneksi di Infrastructure Composer antara tugas alur kerja yang didukung dan kartu Infrastructure Composer yang didukung.

- Tugas alur kerja yang didukung - Tugas Layanan AWS yang dioptimalkan untuk Step Functions. Untuk mempelajari lebih lanjut, lihat [Integrasi yang dioptimalkan untuk Step Functions](#) di Panduan AWS Step Functions Pengembang.
- Kartu Komposer Infrastruktur yang Didukung - Kartu komponen yang disempurnakan didukung. Untuk mempelajari lebih lanjut tentang kartu di Infrastructure Composer, lihat [Mengkonfigurasi dan memodifikasi kartu di Infrastructure Composer](#).

Saat membuat koneksi, tugas dan kartu harus cocok. Layanan AWS Misalnya, Anda dapat menghubungkan tugas alur kerja yang memanggil fungsi Lambda ke kartu komponen yang disempurnakan Fungsi Lambda.

Untuk membuat koneksi, klik dan seret port tugas ke port kiri kartu komponen yang disempurnakan.



Infrastructure Composer akan secara otomatis memperbarui DefinitionSubstitution nilai Anda untuk menentukan koneksi Anda. Berikut adalah contohnya:

```

Transform: AWS::Serverless-2016-10-31
Resources:
  StateMachine:
    Type: AWS::Serverless::StateMachine
    Properties:
      Definition:
        StartAt: Check Stock Value
        States:
          Check Stock Value:
            Type: Task
            Resource: arn:aws:states:::lambda:invoke
            Parameters:
              Payload.$: $
              FunctionName: ${CheckStockValue}
            Next: Choice
          ...
      DefinitionSubstitutions:
        CheckStockValue: !GetAtt CheckStockValue.Arn
        ...
  CheckStockValue:
    Type: AWS::Serverless::Function
    Properties:
      ...

```



## Bekerja dengan file eksternal

Saat Anda membuat alur kerja dari kartu mesin Step Functions State, Infrastructure Composer menyimpan definisi mesin status Anda dalam template Anda menggunakan properti `Definition`. Anda dapat mengonfigurasi Infrastructure Composer untuk menyimpan definisi mesin status Anda pada file eksternal.

### Note

Untuk menggunakan fitur ini dengan Infrastructure Composer dari AWS Management Console, Anda harus mengaktifkan sinkronisasi lokal. Untuk informasi selengkapnya, lihat [Sinkronkan dan simpan proyek Anda secara lokal di konsol Infrastructure Composer](#).

Untuk menyimpan definisi mesin status Anda pada file eksternal

1. Buka panel Resource properties pada kartu mesin Step Functions State Anda.
2. Pilih opsi Gunakan file eksternal untuk definisi mesin negara.
3. Berikan jalur dan nama relatif untuk file definisi mesin status Anda.
4. Pilih Simpan.

Infrastructure Composer akan melakukan hal berikut:

1. Pindahkan definisi mesin status Anda dari `Definition` bidang ke file eksternal Anda.
2. Simpan definisi mesin status Anda dalam file eksternal menggunakan Amazon States Language.
3. Ubah template Anda untuk mereferensikan file eksternal menggunakan `DefinitionUri` bidang.

Pelajari selengkapnya

Untuk mempelajari lebih lanjut tentang Step Functions di Infrastructure Composer, lihat berikut ini:

- [Menggunakan Workflow Studio di Infrastructure Composer](#) di Panduan AWS Step Functions Pengembang.
- [DefinitionSubstitutions dalam AWS SAM template](#) di Panduan AWS Step Functions Pengembang.

## Kartu standar di Infrastructure Composer

Semua AWS CloudFormation sumber daya tersedia untuk digunakan sebagai kartu sumber daya IAC standar dari palet Sumber Daya. Setelah diseret ke kanvas visual, kartu sumber daya IAC standar menjadi kartu komponen standar. Ini berarti kartu adalah satu atau lebih sumber daya IAC standar. Untuk contoh dan detail lebih lanjut, lihat topik di bagian ini.

Anda dapat memodifikasi kode infrastruktur Anda melalui tampilan Template dan melalui jendela Resource properties. Misalnya, berikut ini adalah contoh template awal sumber daya IAC `Alexa::ASK::Skill` standar:

```
Resources:
  Skill:
    Type: Alexa::ASK::Skill
    Properties:
      AuthenticationConfiguration:
        RefreshToken: <String>
        ClientSecret: <String>
        ClientId: <String>
      VendorId: <String>
      SkillPackage:
        S3Bucket: <String>
        S3Key: <String>
```

Templat awal kartu sumber daya IAC standar terdiri dari yang berikut:

- Jenis AWS CloudFormation sumber daya.
- Properti yang diperlukan atau umum digunakan.
- Jenis nilai yang diperlukan untuk menyediakan setiap properti.

### Note

Anda dapat menggunakan Amazon Q untuk menghasilkan saran kode infrastruktur untuk kartu sumber daya standar. Untuk mempelajari selengkapnya, lihat [Menggunakan AWS Infrastructure Composer dengan Amazon Q Developer](#).

## Prosedur

Anda dapat memodifikasi kode infrastruktur untuk setiap sumber daya dalam kartu komponen standar melalui panel Resource properties.

Untuk memodifikasi kartu komponen standar

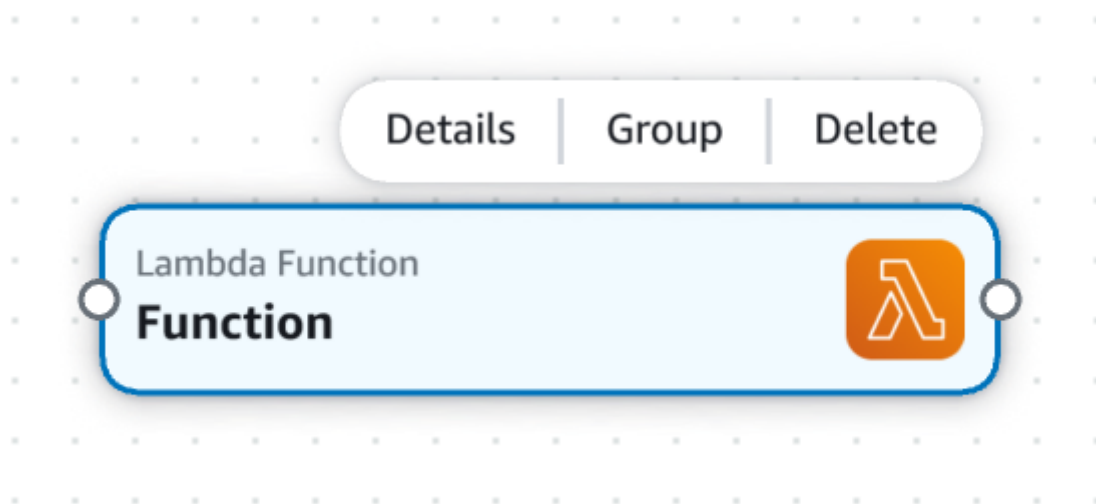
1. Buka panel Resource properties dari kartu komponen IAc standar.
2. Di bidang Editing, pilih sumber daya IAc standar untuk diedit dari daftar dropdown.
3. Ubah kode infrastruktur Anda dan Simpan.

## Hapus kartu di Infrastructure Composer

Bagian ini memberikan instruksi untuk menghapus kartu di AWS Infrastructure Composer.

### Kartu komponen yang disempurnakan

Untuk menghapus kartu komponen yang disempurnakan, pilih kartu yang Anda tempatkan di kanvas visual. Dari menu Tindakan kartu, pilih Hapus.



### Kartu komponen standar

Untuk menghapus kartu komponen standar, Anda harus secara manual menghapus kode infrastruktur untuk setiap AWS CloudFormation sumber daya dari template Anda. Berikut ini adalah cara sederhana untuk mencapai ini:

1. Catat ID logis untuk sumber daya yang akan dihapus.

2. Pada template Anda, cari sumber daya dengan ID logisnya dari Outputs bagian Resources atau.
3. Hapus sumber daya dari template Anda. Ini termasuk ID logis sumber daya dan nilai bersarangnya, seperti Type dan Properties.
4. Periksa tampilan Canvas untuk memverifikasi bahwa sumber daya telah dihapus dari kanvas Anda.

## Lihat pembaruan kode dengan Change Inspector di Infrastructure Composer

Saat Anda mendesain di konsol Infrastructure Composer, kode infrastruktur Anda dibuat secara otomatis. Gunakan Change Inspector untuk melihat pembaruan kode template Anda dan pelajari apa yang dibuat oleh Infrastructure Composer untuk Anda.

Topik ini mencakup penggunaan Infrastructure Composer dari AWS Management Console atau AWS Toolkit for Visual Studio Code ekstensi.

Change Inspector adalah alat visual dalam Infrastructure Composer yang menunjukkan pembaruan kode terbaru.

- Saat Anda mendesain aplikasi Anda, pesan ditampilkan di bagian bawah kanvas visual. Pesan-pesan ini memberikan komentar tentang tindakan yang Anda lakukan.
- Jika didukung, Anda dapat memperluas pesan untuk melihat Change Inspector.
- Change Inspector menampilkan perubahan kode dari interaksi terbaru Anda.

Contoh berikut menunjukkan cara kerja inspektur perubahan:

The screenshot shows the AWS Infrastructure Composer interface. On the left, there is a 'Resources' list with various AWS services. The main canvas displays a 'Lambda Function' resource connected to an 'S3 Bucket' resource. Below the canvas, the 'Change Inspector' panel is open, showing the following CloudFormation template code:

```

15 + Environment:
16 +   Variables:
17 +     BUCKET_BUCKET_NAME: !Ref Bucket
18 +     BUCKET_BUCKET_ARN: !GetAtt Bucket.Arn
19 +   Policies:
20 +     - Statement:
21 +       - Effect: Allow
22 +         Action:
23 +           - s3:GetObject
24 +           - s3:GetObjectAcl
25 +           - s3:GetObjectLegalHold
26 +           - s3:GetObjectRetention
27 +

```

On the right side, the 'Resource properties' panel for the 'S3 Bucket' is visible, showing options like 'Block Public Access' which is checked.

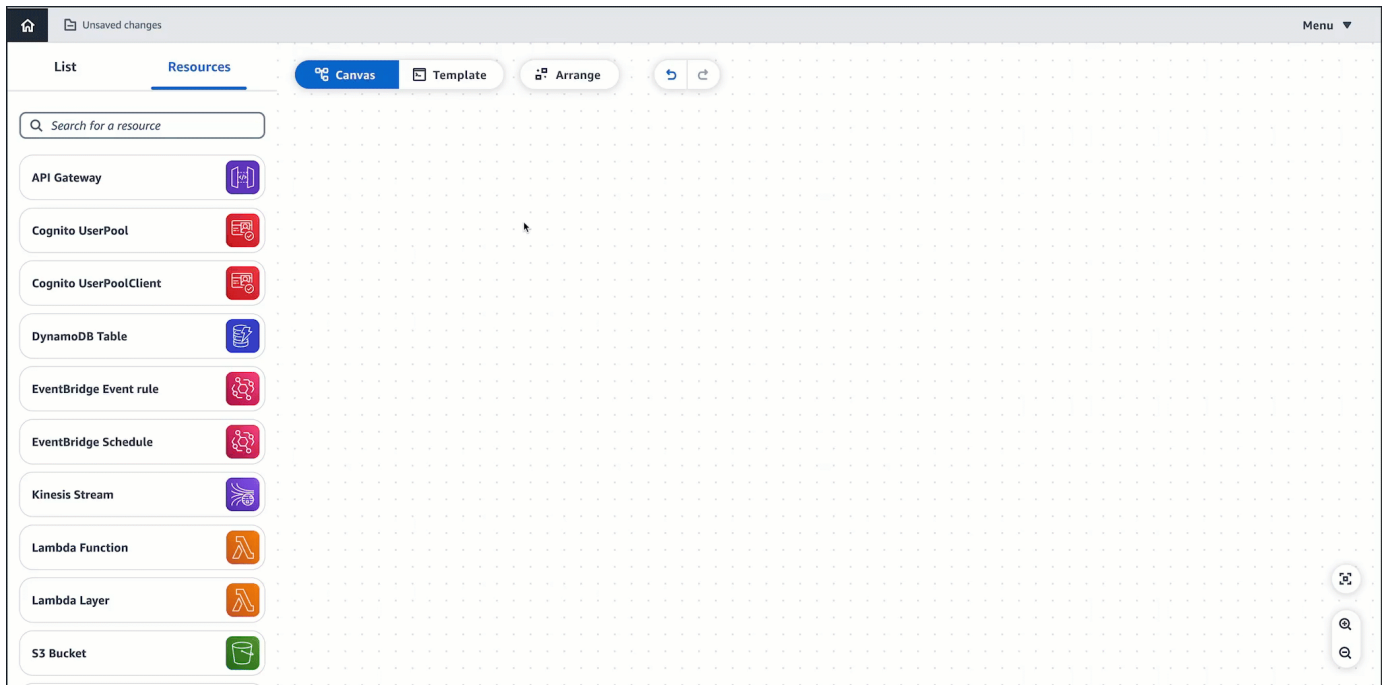
## Keuntungan dari Change Inspector

Change Inspector adalah cara yang bagus untuk melihat kode template yang dibuat oleh Infrastructure Composer untuk Anda. Ini juga merupakan cara yang bagus untuk mempelajari cara menulis kode infrastruktur. Saat Anda mendesain aplikasi di Infrastructure Composer, lihat pembaruan kode di Change Inspector untuk mempelajari tentang kode yang diperlukan untuk menyediakan desain Anda.

## Prosedur

### Menggunakan Change Inspector

1. Perluas pesan untuk memunculkan Change Inspector.



2. Lihat kode yang telah disusun secara otomatis untuk Anda.



- Kode yang disorot hijau menunjukkan kode yang baru ditambahkan.
- Kode yang disorot merah menunjukkan kode yang baru dihapus.
- Nomor baris menunjukkan lokasi dalam template Anda.

3. Ketika beberapa bagian dari template Anda telah diperbarui, Change Inspector mengaturnya. Pilih tombol Sebelumnya dan Berikutnya untuk melihat semua perubahan.



### Note

Untuk Infrastructure Composer dari konsol, Anda dapat melihat perubahan kode dalam konteks seluruh template Anda, dengan menggunakan Template View. Anda juga dapat menyinkronkan Infrastructure Composer dengan lokal IDE dan melihat seluruh template Anda di mesin lokal Anda. Untuk mempelajari selengkapnya, lihat [Connect konsol Infrastructure Composer dengan lokal Anda IDE](#).

## Pelajari selengkapnya

Untuk informasi selengkapnya tentang kode yang dibuat Infrastructure Composer, lihat berikut ini:

- [Koneksi kartu di Infrastructure Composer](#).

## Referensi file eksternal di Infrastructure Composer

Anda dapat menggunakan file eksternal dengan templat AWS Serverless Application Model (AWS SAM) Anda untuk menggunakan kembali kode berulang dan mengatur proyek Anda. Misalnya, Anda mungkin memiliki beberapa REST API sumber daya Amazon API Gateway yang dijelaskan oleh OpenAPI spesifikasi. Alih-alih mereplikasi OpenAPI kode spesifikasi dalam template Anda, Anda dapat membuat satu file eksternal dan referensi untuk setiap sumber daya Anda.

AWS Infrastructure Composer mendukung kasus penggunaan file eksternal berikut:

- API Gerbang REST API sumber daya yang ditentukan oleh eksternal OpenAPI file spesifikasi.
- AWS Step Functions sumber daya mesin negara yang ditentukan oleh file definisi mesin keadaan eksternal.

Untuk mempelajari lebih lanjut tentang mengonfigurasi file eksternal untuk sumber daya yang didukung, lihat berikut ini:

- [DefinitionBody](#) untuk `AWS::Serverless::Api`.
- [DefinitionUri](#) untuk `AWS::Serverless::StateMachine`.

### Note

Untuk mereferensikan file eksternal dengan Infrastructure Composer dari konsol Infrastructure Composer, Anda harus menggunakan Infrastructure Composer dalam mode sinkronisasi lokal. Untuk informasi selengkapnya, lihat [Sinkronkan dan simpan proyek Anda secara lokal di konsol Infrastructure Composer](#).

### Topik

- [Praktik terbaik untuk file referensi eksternal Infrastructure Composer](#)
- [Buat referensi file eksternal di Infrastructure Composer](#)
- [Muat proyek dengan referensi file eksternal di Infrastructure Composer](#)
- [Buat aplikasi yang mereferensikan file eksternal di Infrastructure Composer](#)
- [Referensi sebuah OpenAPI spesifikasi file eksternal dengan Infrastructure Composer](#)



## Praktik terbaik untuk file referensi eksternal Infrastructure Composer

### Gunakan Infrastructure Composer dengan lokal IDE

Bila Anda menggunakan Infrastructure Composer dengan lokal IDE dalam mode sinkronisasi lokal, Anda dapat menggunakan lokal Anda IDE untuk melihat dan memodifikasi file eksternal. Konten dari file eksternal yang didukung yang direferensikan pada template Anda akan secara otomatis diperbarui di kanvas Infrastructure Composer. Untuk mempelajari selengkapnya, lihat [Connect konsol Infrastructure Composer dengan lokal Anda IDE](#).

### Simpan file eksternal dalam direktori induk proyek Anda

Anda dapat membuat subdirektori dalam direktori induk proyek Anda untuk mengatur file eksternal Anda. Infrastructure Composer tidak dapat mengakses file eksternal yang disimpan dalam direktori di luar direktori induk proyek Anda.

### Terapkan aplikasi Anda menggunakan AWS SAM CLI

Saat menerapkan aplikasi Anda ke AWS Cloud, file eksternal lokal harus terlebih dahulu diunggah ke lokasi yang dapat diakses, seperti Amazon Simple Storage Service (Amazon S3). Anda dapat menggunakan AWS SAM CLI untuk memfasilitasi proses ini secara otomatis. Untuk mempelajari selengkapnya, lihat [Mengunggah file lokal saat penerapan](#) di Panduan AWS Serverless Application Model Pengembang.

## Buat referensi file eksternal di Infrastructure Composer

Anda dapat membuat referensi file eksternal dari panel properti sumber daya sumber daya yang didukung.

Untuk membuat referensi file eksternal

1. Dari kartu komponen yang disempurnakan APIGateway atau Step Functions, pilih Detail untuk memunculkan panel properti sumber daya.
2. Temukan dan pilih opsi Gunakan file eksternal.
3. Tentukan jalur relatif ke file eksternal. Ini adalah jalur dari `template.yaml` file Anda ke file eksternal.

Misalnya, untuk mereferensikan file `api-spec.yaml` eksternal dari struktur proyek berikut, tentukan `./api-spec.yaml` sebagai jalur relatif Anda.

```
demo
### api-spec.yaml
### src
# ### Function
# ### index.js
# ### package.json
### template.yaml
```

#### Note

Jika file eksternal dan jalur yang ditentukan tidak ada, Infrastructure Composer akan membuatnya.

4. Simpan perubahan Anda.

## Muat proyek dengan referensi file eksternal di Infrastructure Composer

Ikuti langkah-langkah yang tercantum di halaman ini untuk memuat proyek Infrastructure Composer dengan referensi file eksternal.

Dari konsol Infrastructure Composer

1. Selesaikan langkah-langkah yang tercantum di [Impor template proyek yang ada di konsol Infrastructure Composer](#).
2. Konfirmasi Infrastructure Composer meminta Anda untuk terhubung ke folder root proyek Anda

Jika browser Anda mendukung File System Access API, Infrastructure Composer akan meminta Anda untuk terhubung ke folder root proyek Anda. Infrastructure Composer akan membuka proyek Anda dalam mode sinkronisasi lokal untuk mendukung file eksternal Anda. Jika file eksternal yang direferensikan tidak didukung, Anda akan menerima pesan kesalahan. Untuk informasi selengkapnya tentang pesan galat, lihat [Pemecahan Masalah](#).

Dari Toolkit for VS Code

1. Selesaikan langkah-langkah yang tercantum di [Akses Komposer Infrastruktur dari AWS Toolkit for Visual Studio Code](#).
2. Buka template yang ingin Anda lihat di Infrastructure Composer.

Ketika Anda mengakses Infrastructure Composer dari template, Infrastructure Composer akan secara otomatis mendeteksi file eksternal Anda. Jika file eksternal yang direferensikan tidak didukung, Anda akan menerima pesan kesalahan. Untuk informasi selengkapnya tentang pesan galat, lihat [Pemecahan Masalah](#).

## Buat aplikasi yang mereferensikan file eksternal di Infrastructure Composer

Contoh ini menggunakan AWS SAM CLI untuk membuat aplikasi yang mereferensikan file eksternal untuk definisi mesin statusnya. Anda kemudian memuat proyek Anda di Infrastructure Composer dengan file eksternal Anda direferensikan dengan benar.

### Contoh

1. Pertama, gunakan AWS SAM CLI `sam init` perintah untuk menginisialisasi aplikasi baru bernama `demo`. Selama alur interaktif, pilih template mulai cepat alur kerja multi-langkah.

```
$ sam init
...

Which template source would you like to use?
  1 - AWS Quick Start Templates
  2 - Custom Template Location
Choice: 1

Choose an AWS Quick Start application template
  1 - Hello World Example
  2 - Multi-step workflow
  3 - Serverless API
  4 - Scheduled task
  ...
Template: 2

Which runtime would you like to use?
  1 - dotnet6
  2 - dotnetcore3.1
  ...
 15 - python3.7
 16 - python3.10
 17 - ruby2.7
Runtime: 16
```

```
Based on your selections, the only Package type available is Zip.
We will proceed to selecting the Package type as Zip.
```

```
Based on your selections, the only dependency manager available is pip.
We will proceed copying the template using pip.
```

```
Would you like to enable X-Ray tracing on the function(s) in your application? [y/
N]: ENTER
```

```
Would you like to enable monitoring using CloudWatch Application Insights?
For more info, please view https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/cloudwatch-application-insights.html [y/N]: ENTER
```

```
Project name [sam-app]: demo
```

```
-----
Generating application:
-----
```

```
Name: demo
Runtime: python3.10
Architectures: x86_64
Dependency Manager: pip
Application Template: step-functions-sample-app
Output Directory: .
Configuration file: demo/samconfig.toml
```

```
Next steps can be found in the README file at demo/README.md
```

```
...
```

Aplikasi ini mereferensikan file eksternal untuk definisi mesin negara.

```
...
Resources:
  StockTradingStateMachine:
    Type: AWS::Serverless::StateMachine
    Properties:
      DefinitionUri: statemachine/stock_trader.asl.json
...
```

File eksternal terletak di statemachine subdirektori aplikasi kami.

```
demo
### README.md
### __init__.py
### functions
#   ### __init__.py
#   ### stock_buyer
#   ### stock_checker
#   ### stock_seller
### samconfig.toml
### statemachine
#   ### stock_trader.asl.json
### template.yaml
### tests
```

2. Selanjutnya, muat aplikasi Anda di Infrastructure Composer dari konsol. Dari halaman beranda Infrastructure Composer, pilih Load a CloudFormation template.
3. Pilih folder demo proyek kami dan izinkan prompt untuk melihat file. Pilih `template.yaml` file kami dan pilih Buat. Saat diminta, pilih Simpan perubahan.

### Open project folder

**Project location**  
Select the folder that contains your existing project.

[Select folder](#)

✔ demo

**Template file**  
We will use the project location to automatically detect a template file. If you have multiple files in the folder, select from the dropdown. A copy of your template file will be stored in a folder named `.aws-composer` at the root of your project location.

template.yaml

[Cancel](#) [Create](#)

Infrastructure Composer secara otomatis mendeteksi file definisi mesin keadaan eksternal dan memuatnya. Pilih `StockTradingStateMachines` sumber daya kami dan pilih `Detail` untuk menampilkan panel `Resource properties`. Di sini, Anda dapat melihat bahwa Infrastructure Composer telah secara otomatis terhubung ke file definisi mesin keadaan eksternal kami.

The screenshot displays the AWS Infrastructure Composer interface. On the left, a 'Resources' panel lists various AWS services like API Gateway, Cognito UserPool, and Lambda Function. The central canvas shows a workflow with resources: StockCheckerFunction, StockSellerFunction, StockBuyerFunction, TransactionTable, and ImplicitTimer. A 'Step Functions State machine' resource, 'StockTradingStateMachine', is highlighted with a detailed view on the right. This view shows the state machine definition in JSON format, including tasks like 'Check Stock Value', 'Sell Stock', 'Buy Stock', and 'Record Transaction'. The 'Use external file for state machine definition' checkbox is checked.

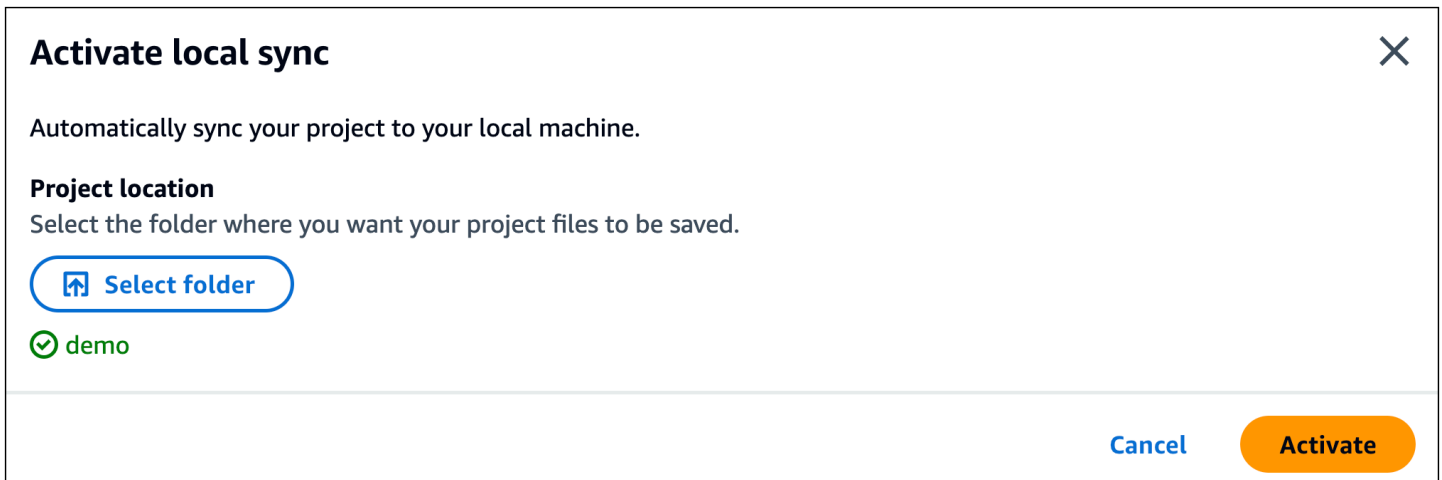
Setiap perubahan yang dilakukan pada file definisi mesin status akan secara otomatis tercermin dalam Infrastructure Composer.

## Referensi sebuah OpenAPI spesifikasi file eksternal dengan Infrastructure Composer

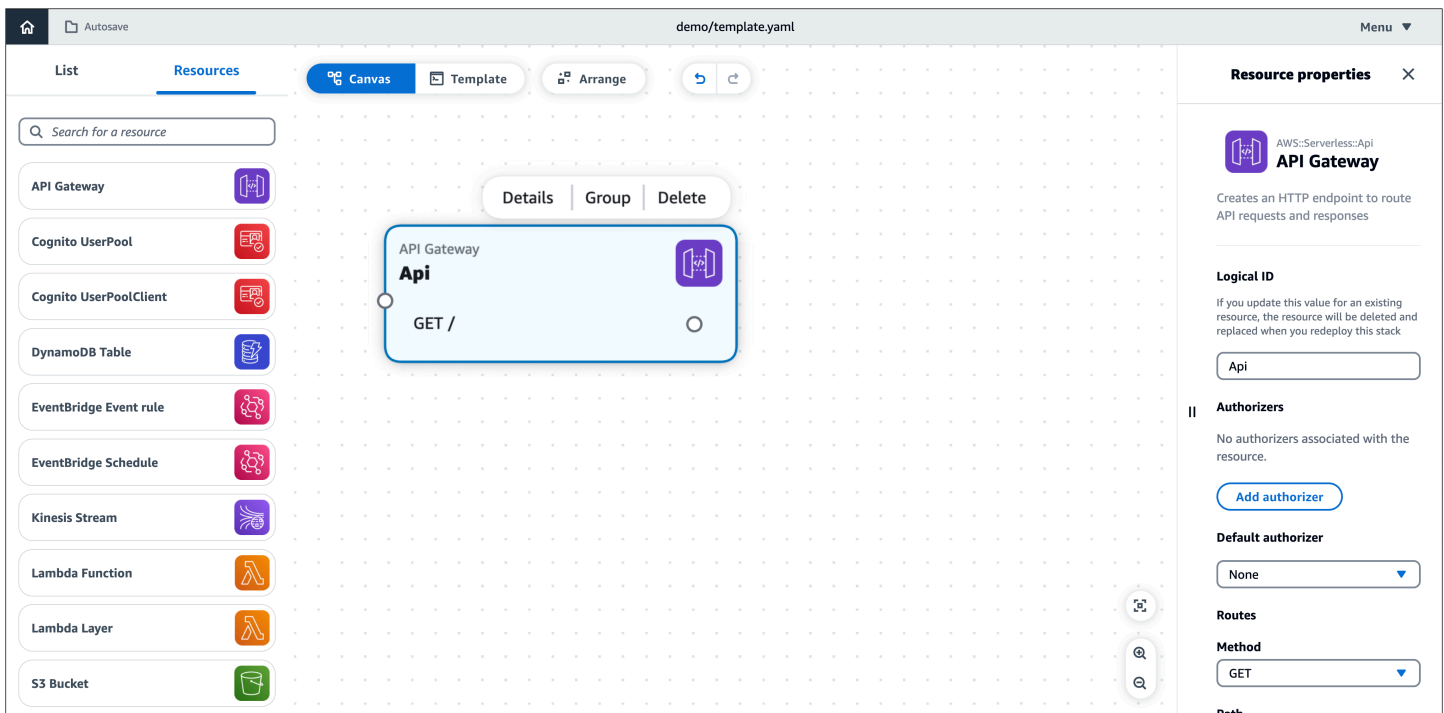
Contoh ini menggunakan Infrastructure Composer dari konsol untuk mereferensikan eksternal OpenAPI file spesifikasi yang mendefinisikan Gateway API REST API.

Pertama, buat proyek baru dari halaman rumah Infrastructure Composer.

Selanjutnya, aktifkan sinkronisasi lokal dengan memilih Aktifkan sinkronisasi lokal dari Menu. Buat folder baru bernama demo, izinkan prompt untuk melihat file, dan pilih Aktifkan. Saat diminta, pilih Simpan perubahan.



Selanjutnya, seret kartu Amazon API Gateway ke kanvas. Pilih Detail untuk memunculkan panel Resource properties.



Dari panel Resource properties, konfigurasi berikut ini dan simpan.

- Pilih opsi Gunakan file eksternal untuk definisi api.
- Masukkan `./api-spec.yaml` sebagai jalur relatif ke file eksternal

## Use external file for api definition



### Relative path to external file

```
./api-spec.yaml
```

Ini membuat direktori berikut di mesin lokal kami:

```
demo
### api-spec.yaml
```

Sekarang, Anda dapat mengkonfigurasi file eksternal pada mesin lokal kami. Menggunakan kamiIDE, buka yang `api-spec.yaml` terletak di folder proyek Anda. Ganti isinya dengan yang berikut:

```
openapi: '3.0'
info: {}
paths:
  /:
    get:
      responses: {}
    post:
      x-amazon-apigateway-integration:
        credentials:
          Fn::GetAtt:
            - ApiQueuesendmessageRole
            - Arn
        httpMethod: POST
        type: aws
        uri:
          Fn::Sub: arn:${AWS::Partition}:apigateway:${AWS::Region}:sqs:path/
            ${AWS::AccountId}/${Queue.QueueName}
```

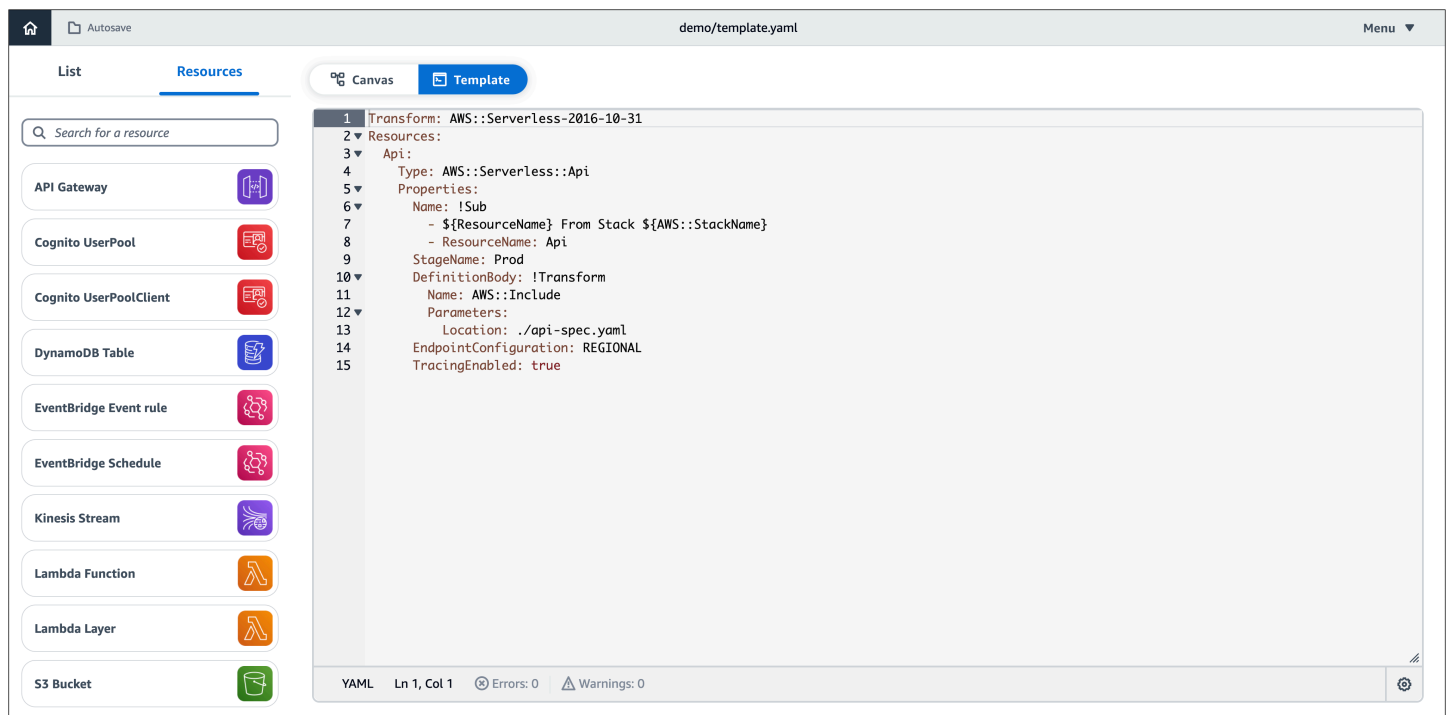


```

requestParameters:
  integration.request.header.Content-Type: ''application/x-www-form-
urlencoded'''
requestTemplates:
  application/json: Action=SendMessage&MessageBody={"data":$input.body}
responses:
  default:
    statusCode: 200
responses:
  '200':
    description: 200 response

```

Dalam tampilan Template Infrastructure Composer, Anda dapat melihat bahwa Infrastructure Composer telah secara otomatis memperbarui template Anda untuk referensi file eksternal.

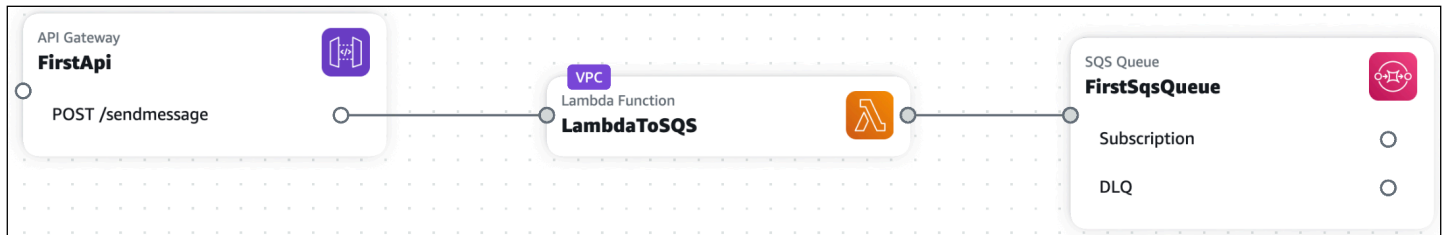


## Integrasikan Komposer Infrastruktur dengan Amazon Virtual Private Cloud (AmazonVPC)

AWS Infrastructure Composer fitur integrasi dengan layanan Amazon Virtual Private Cloud (AmazonVPC). Menggunakan Infrastructure Composer, Anda dapat melakukan hal berikut:

- Identifikasi sumber daya di kanvas Anda yang VPC ada di VPCTag visual.
- Konfigurasi AWS Lambda fungsi dengan VPCs dari templat eksternal.

Gambar berikut menunjukkan adalah contoh aplikasi dengan fungsi Lambda yang dikonfigurasi dengan file. VPC



Untuk mempelajari lebih lanjut tentang AmazonVPC, lihat [Apa itu AmazonVPC?](#) di Panduan VPC Pengguna Amazon.

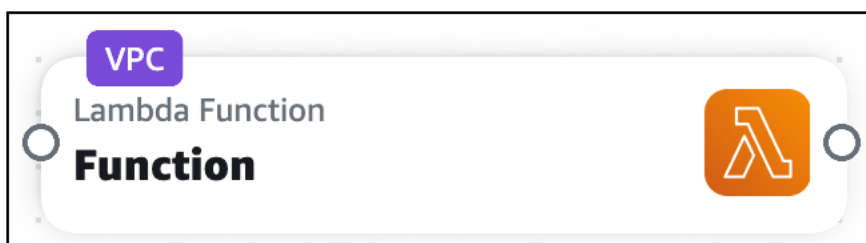
## Topik

- [Identifikasi sumber daya Komposer Infrastruktur dan informasi terkait dalam VPC](#)
- [Konfigurasi fungsi Lambda dengan eksternal VPCs di Infrastructure Composer](#)
- [Parameter dalam template yang diimpor untuk eksternal VPC dengan Infrastructure Composer](#)
- [Menambahkan parameter baru ke template yang diimpor dengan Infrastructure Composer](#)
- [Konfigurasi fungsi Lambda dan VPC didefinisikan dalam template lain dengan Infrastructure Composer](#)

## Identifikasi sumber daya Komposer Infrastruktur dan informasi terkait dalam VPC

Untuk mengintegrasikan Infrastructure Composer dengan AmazonVPC, Anda harus terlebih dahulu mengidentifikasi sumber daya dalam a VPC dan informasi yang diperlukan untuk menyelesaikan integrasi. Ini juga termasuk informasi konfigurasi yang terkait dengan grup keamanan, pengidentifikasi subnet, tipe parameter, tipe, SSM tipe nilai statis.

Infrastructure Composer memvisualisasikan sumber daya dalam VPC menggunakan tag. VPC Tag ini diterapkan pada kartu di kanvas. Berikut ini adalah contoh fungsi Lambda dengan tag: VPC



VPCTag diterapkan ke kartu di kanvas ketika Anda melakukan hal berikut:

- Konfigurasi fungsi Lambda dengan in Infrastructure VPC Composer.
- Impor template yang berisi sumber daya yang dikonfigurasi dengan fileVPC.

## Grup keamanan dan pengidentifikasi subnet

Fungsi Lambda dapat dikonfigurasi dengan beberapa grup keamanan dan subnet. Untuk mengonfigurasi grup keamanan atau subnet untuk fungsi Lambda, berikan nilai dan tipe.

- Nilai — Pengidentifikasi untuk grup keamanan atau subnet. Nilai yang diterima akan bervariasi berdasarkan jenisnya.
- Jenis - Jenis nilai berikut diperbolehkan:
  - Nama parameter
  - AWS Systems Manager (SSM) Penyimpanan Parameter
  - Nilai statis

## Jenis parameter

ParametersBagian AWS CloudFormation template dapat digunakan untuk menyimpan informasi sumber daya di beberapa templat. Untuk informasi selengkapnya tentang parameter, lihat [Parameter](#) di Panduan AWS CloudFormation Pengguna.

Untuk tipe Parameter, Anda dapat memberikan nama parameter. Dalam contoh berikut, kami memberikan nilai nama `PrivateSubnet1` parameter:

**Subnet IDs**

List of VPC subnet identifiers

| <b>Value</b>  | <b>Type</b>  |
|---|--|
| <input style="width: 90%; border: 1px solid #ccc; border-radius: 5px;" type="text" value="PrivateSubnet1"/> <span style="float: right; color: blue; font-size: 1.2em;">✕</span> | <input style="width: 80%; border: 1px solid #ccc; border-radius: 5px;" type="text" value="Parameter"/> <span style="float: right; color: blue; font-size: 1.2em;">▼</span> |

Ketika Anda memberikan nama parameter, Infrastructure Composer mendefinisikannya di `Parameters` bagian template Anda. Kemudian, Infrastructure Composer mereferensikan parameter di sumber daya fungsi Lambda Anda. Berikut adalah contohnya:

...

```

Resources:
  Function:
    Type: AWS::Serverless::Function
    Properties:
      ...
      VpcConfig:
        SubnetIds:
          - !Ref PrivateSubnet1
Parameters:
  PrivateSubnet1:
    Type: AWS::EC2::Subnet::Id
    Description: Parameter is generated by Infrastructure Composer

```

## SSMjenis

SSMParameter Store menyediakan penyimpanan hierarkis yang aman untuk manajemen data konfigurasi dan manajemen rahasia. Untuk informasi selengkapnya, lihat [Penyimpanan Parameter AWS Systems Manager](#) dalam Panduan Pengguna AWS Systems Manager .

Untuk SSMjenisnya, Anda dapat memberikan nilai berikut:

- Referensi dinamis ke nilai dari SSM Parameter Store.
- ID logis dari `AWS::SSM::Parameter` sumber daya yang ditentukan dalam template Anda.

### Referensi dinamis

Anda dapat mereferensikan nilai dari SSM Parameter Store menggunakan referensi dinamis dalam format berikut: `{{resolve:ssm:reference-key}}`. Untuk informasi selengkapnya, lihat [SSMparameter](#) di Panduan AWS CloudFormation Pengguna.

Infrastructure Composer membuat kode infrastruktur untuk mengkonfigurasi fungsi Lambda Anda dengan nilai dari Parameter StoreSSM. Berikut adalah contohnya:

```

...
Resources:
  Function:
    Type: AWS::Serverless::Function
    Properties:
      ...
      VpcConfig:
        SecurityGroupIds:

```

```
- '{{resolve:ssm:demo-app/sg-0b61d5c742dc2c773}}'
```

```
...
```

## ID Logis

Anda dapat mereferensikan `AWS::SSM::Parameter` sumber daya dalam template yang sama dengan ID logis.

Berikut ini adalah contoh `AWS::SSM::Parameter` sumber daya bernama `PrivateSubnet1Parameter` yang menyimpan ID subnet untuk `PrivateSubnet1`:

```
...
Resources:
  PrivateSubnet1Parameter:
    Type: AWS::SSM::Parameter
    Properties:
      Name: /MyApp/VPC/SubnetIds
      Description: Subnet ID for PrivateSubnet1
      Type: String
      Value: subnet-04df123445678a036
```

Berikut ini adalah contoh dari nilai sumber daya ini yang disediakan oleh ID logis untuk fungsi Lambda:

### Subnet IDs

List of VPC subnet identifiers

| Value  | Type                             |
|--|----------------------------------|
| <input type="text" value="PrivateSubnet1Parameter"/> | <input type="text" value="SSM"/> |

Infrastructure Composer membuat kode infrastruktur untuk mengkonfigurasi fungsi Lambda Anda dengan SSM parameter:

```
...
Resources:
  Function:
    Type: AWS::Serverless::Function
    Properties:
```

```

...
VpcConfig:
  SubnetIds:
    - !Ref PrivateSubnet1Parameter
...
PrivateSubnet1Parameter:
  Type: AWS::SSM::Parameter
  Properties:
    ...

```

## Jenis nilai statis

Saat grup keamanan atau subnet digunakan AWS CloudFormation, nilai ID akan dibuat. Anda dapat memberikan ID ini sebagai nilai statis.

Untuk tipe nilai statis, berikut ini adalah nilai yang valid:

- Untuk kelompok keamanan, berikan `GroupId`. Untuk informasi selengkapnya, lihat [Mengembalikan nilai](#) di Panduan AWS CloudFormation Pengguna. Berikut ini adalah contohnya: `sg-0b61d5c742dc2c773`.
- Untuk subnet, berikan `SubnetId`. Untuk informasi selengkapnya, lihat [Mengembalikan nilai](#) di Panduan AWS CloudFormation Pengguna. Berikut ini adalah contohnya: `subnet-01234567890abcdef`.

Infrastructure Composer membuat kode infrastruktur untuk mengkonfigurasi fungsi Lambda Anda dengan nilai statis. Berikut adalah contohnya:

```

...
Resources:
  Function:
    Type: AWS::Serverless::Function
    Properties:
      ...
      VpcConfig:
        SecurityGroupIds:
          - subnet-01234567890abcdef
        SubnetIds:
          - sg-0b61d5c742dc2c773
    ...

```

## Menggunakan beberapa jenis

Untuk grup keamanan dan subnet, Anda dapat menggunakan beberapa jenis secara bersamaan. Berikut ini adalah contoh yang mengonfigurasi tiga grup keamanan untuk fungsi Lambda dengan memberikan nilai dari berbagai jenis:

### Security group IDs

List of VPC security group identifiers

| Value   | Type  |
|---|---|
| <input type="text" value="MySecurityGroup"/> <span style="float: right; color: blue;">✕</span>                          | Parameter <span style="float: right;">▼</span>    |
| <span style="border: 1px solid blue; border-radius: 15px; padding: 5px 15px;">Remove</span>                             |   |
| <input type="text" value="sg-0b61d5c742dc2c773"/> <span style="float: right; color: blue;">✕</span>                     | Static value <span style="float: right;">▼</span> |
| <span style="border: 1px solid blue; border-radius: 15px; padding: 5px 15px;">Remove</span>                             |   |
| <input type="text" value="{{resolve::ssm::demo/sg-0b61d5c742dc23}}"/> <span style="float: right; color: blue;">✕</span> | SSM <span style="float: right;">▼</span>          |
| <span style="border: 1px solid blue; border-radius: 15px; padding: 5px 15px;">Remove</span>                             |   |

Add new item

Infrastructure Composer mereferensikan ketiga nilai di bawah SecurityGroupIds properti:

```

...
Resources:
  Function:
    Type: AWS::Serverless::Function
    Properties:
  
```

```
...
VpcConfig:
  SecurityGroupIds:
    - !Ref MySecurityGroup
    - sg-0b61d5c742dc2c773
    - '{{resolve::ssm::demo/sg-0b61d5c742dc23}}'
...
Parameters:
  MySecurityGroup:
    Type: AWS::EC2::SecurityGroup::Id
    Description: Parameter is generated by Infrastructure Composer
```

## Konfigurasi fungsi Lambda dengan eksternal VPCs di Infrastructure Composer

Untuk mulai mengkonfigurasi fungsi Lambda dengan VPC yang didefinisikan pada template lain, gunakan kartu komponen yang disempurnakan Fungsi Lambda. Kartu ini mewakili fungsi Lambda menggunakan tipe `AWS::Serverless::Function` sumber daya AWS Serverless Application Model (AWS SAM).

Untuk mengkonfigurasi fungsi Lambda dengan VPC dari template eksternal

1. Dari panel properti sumber daya Fungsi Lambda, perluas bagian VPCdropdown pengaturan (lanjutan).
2. Pilih Tetapkan ke eksternal VPC.
3. Berikan nilai untuk grup keamanan dan subnet untuk dikonfigurasi untuk fungsi Lambda. Lihat [Grup keamanan dan pengidentifikasi subnet](#) untuk detail.
4. Simpan perubahan Anda.

## Parameter dalam template yang diimpor untuk eksternal VPC dengan Infrastructure Composer

Saat Anda mengimpor template yang ada dengan parameter yang ditentukan untuk grup keamanan dan subnet eksternalVPC, Infrastructure Composer menyediakan daftar dropdown untuk memilih parameter Anda.

Berikut ini adalah contoh `Parameters` bagian dari template yang diimpor:



```

...
Parameters:
  VPCSecurityGroups:
    Description: Security group IDs generated by Infrastructure Composer
    Type: List<AWS::EC2::SecurityGroup::Id>
  VPCSubnets:
    Description: Subnet IDs generated by Infrastructure Composer
    Type: List<AWS::EC2::Subnet::Id>
  VPCSubnet:
    Description: Subnet Id generated by Infrastructure Composer
    Type: AWS::EC2::Subnet::Id
...

```

Saat mengonfigurasi eksternal VPC untuk fungsi Lambda baru di kanvas, parameter ini akan tersedia dari daftar dropdown. Berikut adalah contohnya:

### Subnet IDs

List of VPC subnet identifiers

| Value   | Type  |
|---|---|
| <input style="width: 100%; border: 1px solid #ccc; border-radius: 5px;" type="text" value="🔍  "/>   | <div style="border: 1px solid #ccc; border-radius: 5px; padding: 5px; display: inline-block;">           Parameter ▼         </div> |
| <div style="border-bottom: 1px solid #ccc; padding: 5px 5px 0 5px;">VPCSubnets</div> <div style="padding: 5px 5px 0 5px;">VPCSubnet</div> |   |

### Keterbatasan saat mengimpor tipe parameter daftar

Biasanya, Anda dapat menentukan beberapa grup keamanan dan pengidentifikasi subnet untuk setiap fungsi Lambda. Jika template yang ada berisi tipe parameter daftar, seperti `List<AWS::EC2::SecurityGroup::Id>` atau `List<AWS::EC2::Subnet::Id>`, Anda hanya dapat menentukan satu pengenal.

Untuk informasi selengkapnya tentang jenis daftar parameter, lihat [Jenis parameter AWS-spesifik yang didukung](#) di Panduan AWS CloudFormation Pengguna.

Berikut ini adalah contoh template yang mendefinisikan `VPCSecurityGroups` sebagai tipe parameter daftar:

```

...
Parameters:
  VPCSecurityGroups:
    Description: Security group IDs generated by Infrastructure Composer
    Type: List<AWS::EC2::SecurityGroup::Id>
...

```

Di Infrastructure Composer, jika Anda memilih `VPCSecurityGroups` nilai sebagai pengidentifikasi grup keamanan untuk fungsi Lambda, Anda akan melihat pesan berikut:

### Security group IDs

List of VPC security group identifiers

| Value  | Type  |
|--|---|
| <input style="width: 90%; border: 1px solid #ccc; border-radius: 5px;" type="text" value="VPCSecurityGroups"/> <span style="float: right; border: 1px solid #ccc; border-radius: 5px; padding: 2px 5px; color: blue;">×</span> | <div style="border: 1px solid #ccc; border-radius: 5px; padding: 2px 5px; display: inline-block;">Parameter ▼</div> |
| <div style="border: 1px solid #ccc; border-radius: 15px; padding: 5px 15px; display: inline-block; background-color: #f0f0f0;">Add new item</div>  |   |

Only one List<AWS::EC2::SecurityGroup::Id> parameter type can be provided.

Keterbatasan ini terjadi karena `SecurityGroupIds` dan `SubnetIds` properti `AWS::Lambda::Function VpcConfig` objek keduanya hanya menerima daftar nilai string. Karena tipe parameter daftar tunggal berisi daftar string, itu bisa menjadi satu-satunya objek yang disediakan ketika ditentukan.

Untuk jenis parameter daftar, berikut ini adalah contoh bagaimana mereka didefinisikan dalam template ketika dikonfigurasi dengan fungsi Lambda:

```

...
Parameters:
  VPCSecurityGroups:
    Description: Security group IDs generated by Infrastructure Composer
    Type: List<AWS::EC2::SecurityGroup::Id>
  VPCSubnets:
    Description: Subnet IDs generated by Infrastructure Composer
    Type: List<AWS::EC2::Subnet::Id>
Resources:
...

```

```

MyFunction:
  Type: AWS::Serverless::Function
  Properties:
    ...
  VpcConfig:
    SecurityGroupIds: !Ref VPCSecurityGroups
    SubnetIds: !Ref VPCSubnets

```

## Menambahkan parameter baru ke template yang diimpor dengan Infrastructure Composer

Saat Anda mengimpor template yang ada dengan parameter yang ditentukan, Anda juga dapat membuat parameter baru. Alih-alih memilih parameter yang ada dari daftar dropdown, berikan tipe dan nilai baru. Berikut ini adalah contoh yang membuat parameter baru bernama `MySecurityGroup`:

### Security group IDs

List of VPC security group identifiers

| Value   | Type  |
|---|---|
| <input style="width: 90%; border: none;" type="text" value="MySecurityGroup"/> <span style="float: right; border: 1px solid #ccc; border-radius: 50%; padding: 2px 5px;">×</span> | <div style="border: 1px solid #ccc; border-radius: 5px; padding: 2px 5px; display: inline-block;">Parameter ▼</div> |
| Use: "MySecurityGroup"  |   |
| VPCSecurityGroups   |   |

Untuk semua nilai baru yang Anda berikan di panel properti Resource untuk fungsi Lambda, Infrastructure Composer mendefinisikannya dalam daftar di bawah `SecurityGroupIds` atau `SubnetIds` properti fungsi Lambda. Berikut adalah contohnya:

```

...
Resources:
  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      ...
      VpcConfig:
        SecurityGroupIds:
          - sg-94b3a1f6

```

```
SubnetIds:
  - !Ref SubnetParameter
  - !Ref VPCSubnet
```

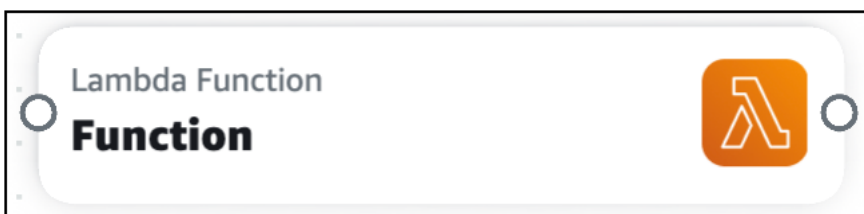
Jika Anda ingin mereferensikan ID logis dari jenis parameter daftar dari template eksternal, kami sarankan Anda menggunakan tampilan Template dan langsung memodifikasi template Anda. ID logis dari tipe parameter daftar harus selalu disediakan sebagai nilai tunggal dan sebagai satu-satunya nilai.

```
...
Parameters:
  VPCSecurityGroups:
    Description: Security group IDs generated by Infrastructure Composer
    Type: List<AWS::EC2::SecurityGroup::Id>
  VPCSubnets:
    Description: Subnet IDs generated by Infrastructure Composer
    Type: List<AWS::EC2::Subnet::Id>
Resources:
  ...
  MyFunction:
    Type: AWS::Serverless::Function
    Properties:
      ...
      VpcConfig:
        SecurityGroupIds: !Ref VPCSecurityGroups # Valid syntax
        SubnetIds:
          - !Ref VPCSubnets # Not valid syntax
```

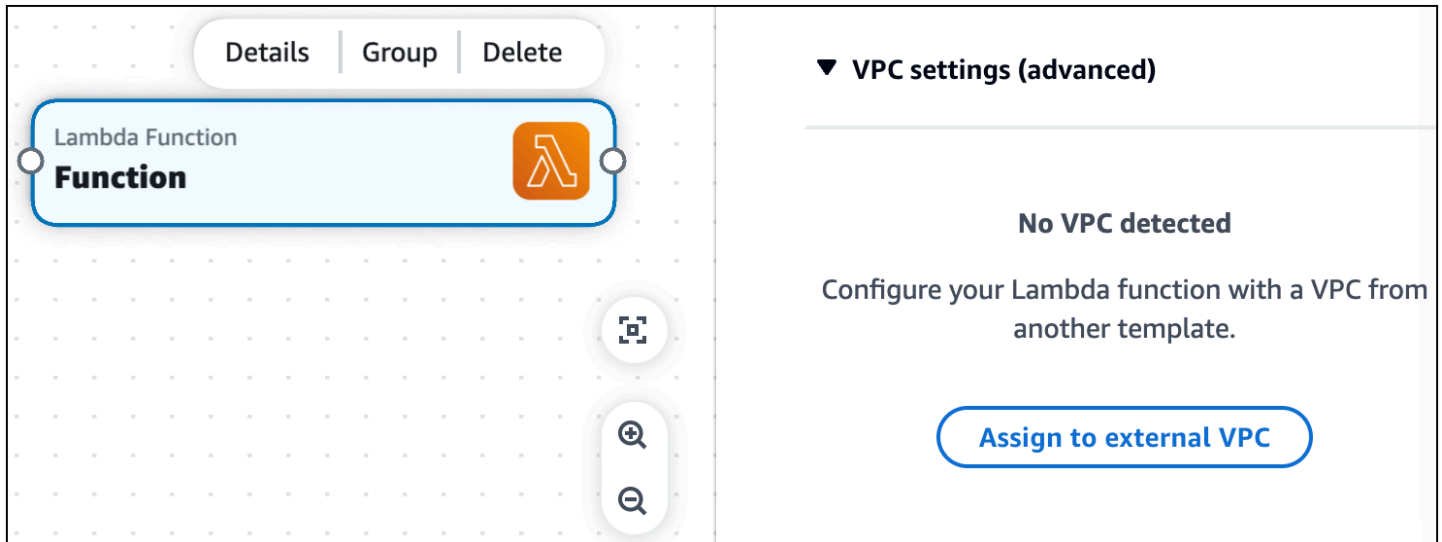
## Konfigurasi fungsi Lambda dan VPC didefinisikan dalam template lain dengan Infrastructure Composer

Dalam contoh ini, kita mengkonfigurasi fungsi Lambda di Infrastructure Composer dengan VPC didefinisikan pada template lain.

Kita mulai dengan menyeret kartu komponen yang disempurnakan Fungsi Lambda ke kanvas.



Selanjutnya, kita membuka panel Resource properties kartu dan memperluas bagian dropdown VPC pengaturan (lanjutan).

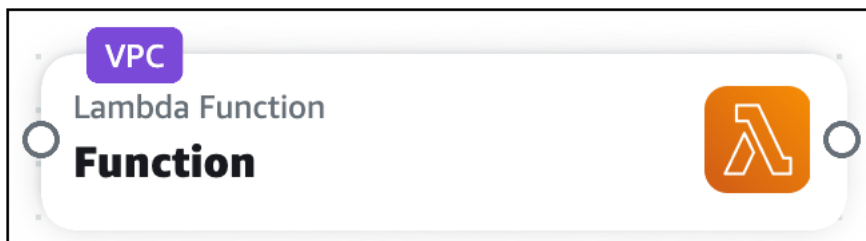


Selanjutnya, kita pilih Tetapkan ke eksternal VPC untuk mulai mengkonfigurasi VPC dari template eksternal.

Dalam contoh ini, kami mereferensikan ID grup keamanan dan ID subnet. Nilai-nilai ini dibuat saat template yang mendefinisikan VPC dikerahkan. Kami memilih tipe nilai Statis dan memasukkan nilai kami IDs. Kami memilih Simpan setelah selesai.

The screenshot shows the configuration interface for a Lambda Function in AWS Infrastructure Composer. On the left, a card for the 'Lambda Function' is displayed with a 'Function' tag and a Lambda icon. The right panel is titled 'Security group IDs' and 'Subnet IDs', both showing a list of VPC identifiers. Each list has a search input, a dropdown menu for 'Type' (set to 'Static value'), and an 'Add new item' button. A 'Remove from VPC' button is located at the bottom right of the configuration area. At the very bottom right, there are 'Cancel' and 'Save' buttons.

Sekarang fungsi Lambda kami dikonfigurasi dengan kamiVPC, VPC tag ditampilkan pada kartu kami.



Infrastructure Composer telah membuat kode infrastruktur untuk mengkonfigurasi fungsi Lambda kami dengan grup keamanan dan subnet eksternal. VPC

```

Transform: AWS::Serverless-2016-10-31
Resources:
  Function:
    Type: AWS::Serverless::Function
    Properties:
      Description: !Sub
        - Stack ${AWS::StackName} Function ${ResourceName}
        - ResourceName: Function
      CodeUri: src/Function
  
```

```
Handler: index.handler
Runtime: nodejs18.x
MemorySize: 3008
Timeout: 30
Tracing: Active
VpcConfig:
  SecurityGroupIds:
    - sg-10f35d07e1be09e15
  SubnetIds:
    - subnet-0d80727ca90325716
FunctionLogGroup:
  Type: AWS::Logs::LogGroup
  DeletionPolicy: Retain
  Properties:
    LogGroupName: !Sub /aws/lambda/${Function}
```

# Menerapkan aplikasi tanpa server Infrastructure Composer Anda ke Cloud AWS

Gunakan AWS Infrastructure Composer untuk merancang aplikasi tanpa server yang siap digunakan. Untuk menyebarkan, gunakan layanan yang AWS CloudFormation kompatibel. Kami merekomendasikan menggunakan [AWS Serverless Application Model \(AWS SAM\)](#).

AWS SAM adalah kerangka kerja sumber terbuka yang menyediakan alat pengembang untuk membangun dan menjalankan aplikasi tanpa server. AWS Dengan AWS SAM sintaks singkatan, pengembang mendeklarasikan AWS CloudFormation sumber daya dan sumber daya tanpa server khusus yang diubah menjadi infrastruktur selama penerapan.

## AWS SAM Konsep penting

Sebelum Anda menggunakannya AWS SAM, penting bagi Anda untuk membiasakan diri dengan beberapa konsep fundamentalnya.

- [Cara AWS SAM kerja](#): Topik ini, yang ada di Panduan AWS Serverless Application Model Pengembang, memberikan informasi penting tentang komponen utama yang Anda gunakan untuk membuat aplikasi tanpa serverless Anda: AWS SAM CLI, AWS SAM proyek, dan AWS SAM template.
- [Cara menggunakan AWS Serverless Application Model \(AWS SAM\)](#): Topik ini, yang ada di Panduan AWS Serverless Application Model Pengembang, memberikan ikhtisar tingkat tinggi tentang langkah-langkah yang perlu Anda selesaikan untuk digunakan AWS SAM untuk menyebarkan aplikasi Anda ke Cloud. AWS

Saat Anda mendesain aplikasi Anda di Infrastructure Composer, Anda dapat menggunakan `aws sam sync` perintah untuk memiliki AWS SAM CLI secara otomatis mendeteksi perubahan lokal dan menerapkan perubahan tersebut ke AWS CloudFormation. Untuk mempelajari lebih lanjut, lihat [Menggunakan sinkronisasi sam](#) di Panduan AWS Serverless Application Model Pengembang.

## Langkah selanjutnya

Lihat [Siapkan untuk digunakan dengan AWS SAM CLI dan Komposer Infrastruktur](#) untuk mempersiapkan penerapan aplikasi Anda.



# Siapkan untuk digunakan dengan AWS SAM CLI dan Komposer Infrastruktur

Untuk menggunakan aplikasi Anda AWS SAM, Anda harus menginstal dan mengakses aplikasi terlebih dahulu AWS CLI dan AWS SAM CLI. Topik di bagian ini memberikan rincian tentang melakukan ini.

## Instal AWS CLI

Kami merekomendasikan menginstal dan mengatur AWS CLI sebelum menginstal AWS SAM CLI. Untuk petunjuk, lihat [Menginstal atau memperbarui ke versi terbaru](#) dari Panduan AWS Command Line Interface Pengguna. AWS CLI

### Note

Setelah menginstal AWS CLI, Anda harus mengkonfigurasi AWS kredensial. Untuk mempelajari selengkapnya, lihat [Penyiapan cepat](#) di Panduan AWS Command Line Interface Pengguna.

## Instal AWS SAM CLI

Untuk menginstal AWS SAM CLI, lihat [Memasang AWS SAM CLI](#) di Panduan Developer AWS Serverless Application Model .

## Akses AWS SAM CLI

Jika Anda menggunakan Infrastructure Composer dari AWS Management Console, Anda memiliki opsi berikut untuk menggunakan AWS SAM CLI.

### Aktifkan mode sinkronisasi lokal

Dengan mode sinkronisasi lokal, folder proyek Anda, termasuk AWS SAM template, secara otomatis disimpan ke mesin lokal Anda. Infrastructure Composer menyusun direktori proyek Anda dengan cara yang AWS SAM mengenali. Anda dapat menjalankan AWS SAM CLI dari direktori root proyek Anda.

Untuk informasi selengkapnya tentang mode sinkronisasi lokal, lihat [Sinkronkan dan simpan proyek Anda secara lokal di konsol Infrastructure Composer](#).

## Ekspor template Anda

Anda dapat mengekspor template Anda ke mesin lokal Anda. Kemudian, jalankan AWS SAM CLI dari folder induk yang berisi template. Anda juga dapat menggunakan `--template-file` opsi dengan apa saja AWS SAM CLI perintah dan berikan jalur ke template Anda.

## Gunakan Infrastructure Composer dari AWS Toolkit for Visual Studio Code

Anda dapat menggunakan Infrastructure Composer dari Toolkit for VS Code untuk membawa Infrastructure Composer ke mesin lokal Anda. Kemudian, gunakan Infrastructure Composer dan AWS SAM CLI dari VS Code.

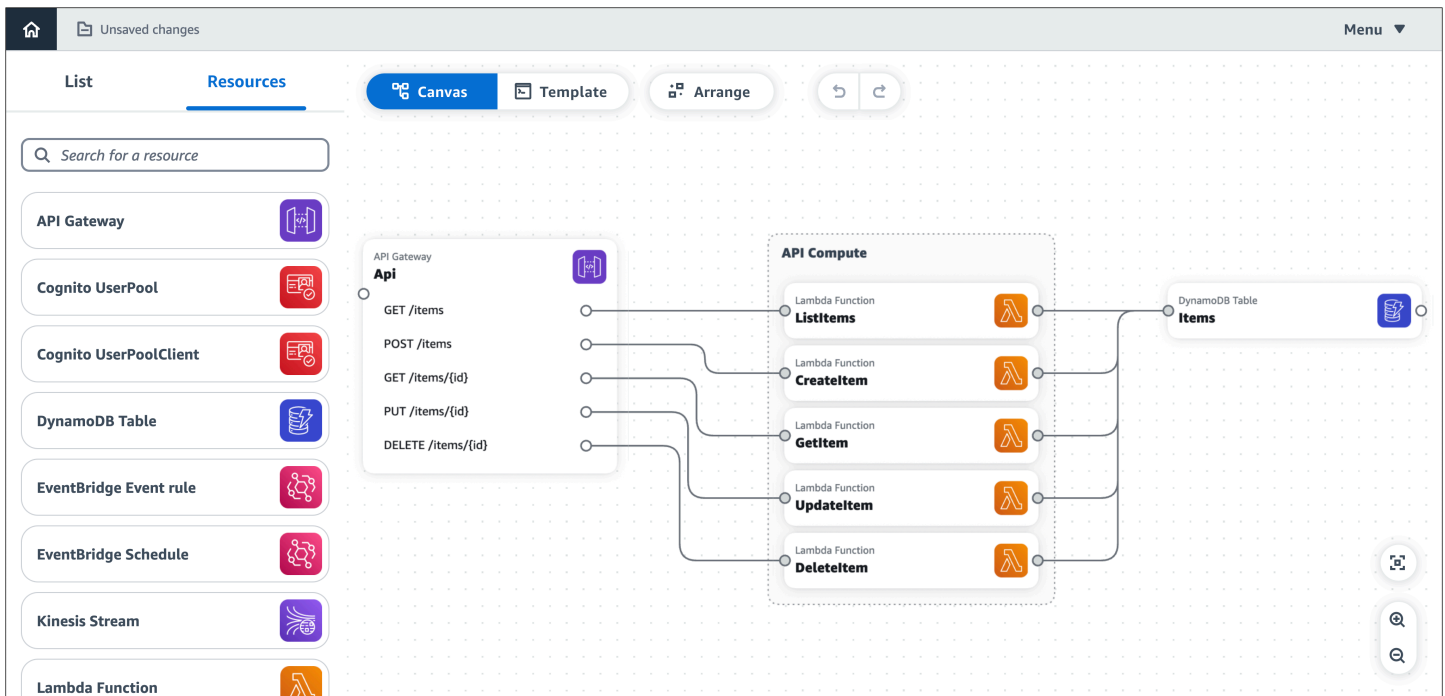
## Langkah selanjutnya

Untuk menerapkan aplikasi Anda, lihat. [Gunakan Infrastructure Composer AWS SAM untuk membangun dan menyebarkan](#)

## Gunakan Infrastructure Composer AWS SAM untuk membangun dan menyebarkan

Sekarang setelah Anda selesai [Siapkan untuk digunakan dengan AWS SAM CLI dan Komposer Infrastruktur](#), Anda dapat menerapkan aplikasi Anda dengan AWS SAM dan Infrastructure Composer. Bagian ini memberikan contoh yang merinci bagaimana Anda dapat melakukan ini. Anda juga dapat merujuk ke [Menerapkan aplikasi dan sumber daya Anda AWS SAM](#) di Panduan AWS Serverless Application Model Pengembang untuk petunjuk tentang penerapan aplikasi Anda. AWS SAM

Contoh ini menunjukkan cara membangun dan menyebarkan aplikasi demo Infrastructure Composer. Aplikasi demo memiliki sumber daya berikut:



### Note

- Untuk mempelajari lebih lanjut tentang aplikasi demo, lihat [Memuat dan memodifikasi proyek demo Infrastructure Composer](#).
- Untuk contoh ini, kami menggunakan Infrastructure Composer dengan sinkronisasi lokal diaktifkan.

## 1. Gunakan sam build perintah untuk membangun aplikasi.

```
$ sam build
...
Build Succeeded

Built Artifacts : .aws-sam/build
Built Template  : .aws-sam/build/template.yaml

Commands you can use next
=====
[*] Validate SAM template: sam validate
[*] Invoke Function: sam local invoke
[*] Test Function in the Cloud: sam sync --stack-name {{stack-name}} --watch
```

```
[*] Deploy: sam deploy --guided
```

The AWS SAM CLI membuat `./aws-sam` direktori di folder proyek. Direktori ini berisi artefak build untuk fungsi Lambda aplikasi. Berikut adalah output dari direktori proyek:

```
.
### README.md
### samconfig.toml
### src
#   ### CreateItem
# #   ### index.js
# #   ### package.json
#   ### DeleteItem
# #   ### index.js
# #   ### package.json
#   ### GetItem
# #   ### index.js
# #   ### package.json
#   ### ListItems
# #   ### index.js
# #   ### package.json
#   ### UpdateItem
#     ### index.js
#     ### package.json
### template.yaml
```

2. Sekarang, aplikasi siap digunakan. Kami akan menggunakan `sam deploy --guided`. Ini mempersiapkan aplikasi Anda untuk penyebaran melalui serangkaian petunjuk.

```
$ sam deploy --guided
...
Configuring SAM deploy
=====

Looking for config file [samconfig.toml] : Found
Reading default arguments : Success

Setting default arguments for 'sam deploy'
=====
Stack Name [aws-app-composer-basic-api]:
AWS Region [us-west-2]:
```

```

#Shows you resources changes to be deployed and require a 'Y' to initiate
deploy
Confirm changes before deploy [y/N]:
#SAM needs permission to be able to create roles to connect to the resources in
your template
Allow SAM CLI IAM role creation [Y/n]:
#Preserves the state of previously provisioned resources when an operation
fails
Disable rollback [y/N]:
ListItems may not have authorization defined, Is this okay? [y/N]: y
CreateItem may not have authorization defined, Is this okay? [y/N]: y
GetItem may not have authorization defined, Is this okay? [y/N]: y
UpdateItem may not have authorization defined, Is this okay? [y/N]: y
DeleteItem may not have authorization defined, Is this okay? [y/N]: y
Save arguments to configuration file [Y/n]:
SAM configuration file [samconfig.toml]:
SAM configuration environment [default]:

```

The AWS SAM CLI menampilkan ringkasan dari apa yang akan digunakan:

```

Deploying with following values
=====
Stack name           : aws-app-composer-basic-api
Region              : us-west-2
Confirm changeset   : False
Disable rollback    : False
Deployment s3 bucket : aws-sam-cli-managed-default-samcliarn-s3-demo-
bucket-1b3x26zbcdkqr
Capabilities         : ["CAPABILITY_IAM"]
Parameter overrides : {}
Signing Profiles    : {}

```

The AWS SAM CLI menyebarkan aplikasi, pertama dengan membuat AWS CloudFormation changeset:

```

Initiating deployment
=====
Uploading to aws-app-composer-basic-api/4181c909ee2440a728a7a129dafb83d4.template
7087 / 7087 (100.00%)

Waiting for changeset to be created..
CloudFormation stack changeset

```

| Operation<br>ResourceType            | LogicalResourceId<br>Replacement         |
|--------------------------------------|--|
| + Add<br>AWS::ApiGateway::Deployment | ApiDeploymentcc153d135b<br>N/A           |
| + Add<br>AWS::ApiGateway::Stage      | ApiProdStage<br>N/A                      |
| + Add<br>AWS::ApiGateway::RestApi    | Api<br>N/A                               |
| + Add<br>AWS::Lambda::Permission     | CreateItemApiPOSTitemsPermissionP<br>N/A |
| + Add<br>AWS::IAM::Role              | rod<br>CreateItemRole<br>N/A             |
| + Add<br>AWS::Lambda::Function       | CreateItem<br>N/A                        |
| + Add<br>AWS::Lambda::Permission     | DeleteItemApiDELETEitemsidPermiss<br>N/A |
| + Add<br>AWS::IAM::Role              | ionProd<br>DeleteItemRole<br>N/A         |
| + Add<br>AWS::Lambda::Function       | DeleteItem<br>N/A                        |
| + Add<br>AWS::Lambda::Permission     | GetItemApiGETitemsidPermissionPro<br>N/A |
| + Add<br>AWS::IAM::Role              | d<br>GetItemRole<br>N/A                  |
| + Add<br>AWS::Lambda::Function       | GetItem<br>N/A                           |
| + Add<br>AWS::DynamoDB::Table        | Items<br>N/A                             |
| + Add<br>AWS::Lambda::Permission     | ListItemsApiGETitemsPermissionPro<br>N/A |
| + Add<br>AWS::IAM::Role              | d<br>ListItemsRole<br>N/A                |
| + Add<br>AWS::Lambda::Function       | ListItems<br>N/A                         |
| + Add<br>AWS::Lambda::Permission     | UpdateItemApiPUTitemsidPermission<br>N/A |
|                                      | Prod                                     |

```

+ Add UpdateItemRole
  AWS::IAM::Role N/A
+ Add UpdateItem
  AWS::Lambda::Function N/A
-----

```

```

Changeset created successfully. arn:aws:cloudformation:us-
west-2:513423067560:changeSet/samcli-deploy1677472539/967ab543-f916-4170-b97d-
c11a6f9308ea

```

Kemudian, AWS SAM CLI menyebarkan aplikasi:

```

CloudFormation events from stack operations (refresh every 0.5 seconds)
-----

```

| ResourceStatus     | LogicalResourceId | ResourceType         | ResourceStatusReason        |
|--------------------|-------------------|----------------------|-----------------------------|
| CREATE_IN_PROGRESS | -                 | AWS::DynamoDB::Table | Items                       |
| CREATE_IN_PROGRESS | -                 | AWS::DynamoDB::Table | Items                       |
| CREATE_COMPLETE    | -                 | AWS::DynamoDB::Table | Items                       |
| CREATE_IN_PROGRESS | -                 | AWS::IAM::Role       |                             |
|                    | DeleteItemRole    | -                    |                             |
| CREATE_IN_PROGRESS | -                 | AWS::IAM::Role       |                             |
|                    | ListItemsRole     | -                    |                             |
| CREATE_IN_PROGRESS | -                 | AWS::IAM::Role       |                             |
|                    | UpdateItemRole    | -                    |                             |
| CREATE_IN_PROGRESS | -                 | AWS::IAM::Role       | GetItemRole                 |
| CREATE_IN_PROGRESS | -                 | AWS::IAM::Role       |                             |
|                    | CreateItemRole    | -                    |                             |
| CREATE_IN_PROGRESS | -                 | AWS::IAM::Role       | Resource creation Initiated |
|                    | DeleteItemRole    |                      |                             |
| CREATE_IN_PROGRESS | -                 | AWS::IAM::Role       | Resource creation Initiated |
|                    | ListItemsRole     |                      |                             |
| CREATE_IN_PROGRESS | -                 | AWS::IAM::Role       | GetItemRole                 |
|                    | CreateItemRole    |                      |                             |
| CREATE_IN_PROGRESS | -                 | AWS::IAM::Role       | Resource creation Initiated |
|                    | DeleteItemRole    |                      |                             |
| CREATE_IN_PROGRESS | -                 | AWS::IAM::Role       | Resource creation Initiated |
|                    | CreateItemRole    |                      |                             |

|                    |                             |                          |             |
|--------------------|-----------------------------|--------------------------|-------------|
| CREATE_COMPLETE    |                             | AWS::IAM::Role           |             |
| DeleteItemRole     | -                           |                          |             |
| CREATE_COMPLETE    |                             | AWS::IAM::Role           |             |
| ListItemsRole      | -                           |                          |             |
| CREATE_COMPLETE    |                             | AWS::IAM::Role           | GetItemRole |
|                    | -                           |                          |             |
| CREATE_COMPLETE    |                             | AWS::IAM::Role           |             |
| UpdateItemRole     | -                           |                          |             |
| CREATE_COMPLETE    |                             | AWS::IAM::Role           |             |
| CreateItemRole     | -                           |                          |             |
| CREATE_IN_PROGRESS |                             | AWS::Lambda::Function    | DeleteItem  |
|                    | -                           |                          |             |
| CREATE_IN_PROGRESS |                             | AWS::Lambda::Function    | CreateItem  |
|                    | -                           |                          |             |
| CREATE_IN_PROGRESS |                             | AWS::Lambda::Function    | ListItems   |
|                    | -                           |                          |             |
| CREATE_IN_PROGRESS |                             | AWS::Lambda::Function    | UpdateItem  |
|                    | -                           |                          |             |
| CREATE_IN_PROGRESS |                             | AWS::Lambda::Function    | DeleteItem  |
|                    | Resource creation Initiated |                          |             |
| CREATE_IN_PROGRESS |                             | AWS::Lambda::Function    | GetItem     |
|                    | -                           |                          |             |
| CREATE_IN_PROGRESS |                             | AWS::Lambda::Function    | ListItems   |
|                    | Resource creation Initiated |                          |             |
| CREATE_IN_PROGRESS |                             | AWS::Lambda::Function    | CreateItem  |
|                    | Resource creation Initiated |                          |             |
| CREATE_IN_PROGRESS |                             | AWS::Lambda::Function    | UpdateItem  |
|                    | Resource creation Initiated |                          |             |
| CREATE_IN_PROGRESS |                             | AWS::Lambda::Function    | GetItem     |
|                    | Resource creation Initiated |                          |             |
| CREATE_COMPLETE    |                             | AWS::Lambda::Function    | DeleteItem  |
|                    | -                           |                          |             |
| CREATE_COMPLETE    |                             | AWS::Lambda::Function    | ListItems   |
|                    | -                           |                          |             |
| CREATE_COMPLETE    |                             | AWS::Lambda::Function    | CreateItem  |
|                    | -                           |                          |             |
| CREATE_COMPLETE    |                             | AWS::Lambda::Function    | UpdateItem  |
|                    | -                           |                          |             |
| CREATE_COMPLETE    |                             | AWS::Lambda::Function    | GetItem     |
|                    | -                           |                          |             |
| CREATE_IN_PROGRESS |                             | AWS::ApiGateway::RestApi | Api         |
|                    | -                           |                          |             |
| CREATE_IN_PROGRESS |                             | AWS::ApiGateway::RestApi | Api         |
|                    | Resource creation Initiated |                          |             |



|                    |                                   |                             |                             |
|--------------------|-----------------------------------|-----------------------------|-----------------------------|
| CREATE_COMPLETE    | -                                 | AWS::ApiGateway::RestApi    | Api                         |
| CREATE_IN_PROGRESS | GetItemApiGETItemsidPermissionPro | AWS::Lambda::Permission     | d                           |
| CREATE_IN_PROGRESS | ListItemsApiGETItemsPermissionPro | AWS::Lambda::Permission     | d                           |
| CREATE_IN_PROGRESS | DeleteItemApiDELETEItemsidPermiss | AWS::Lambda::Permission     | ionProd                     |
| CREATE_IN_PROGRESS | ApiDeploymentccc153d135b          | AWS::ApiGateway::Deployment |                             |
| CREATE_IN_PROGRESS | UpdateItemApiPUTItemsidPermission | AWS::Lambda::Permission     | Prod                        |
| CREATE_IN_PROGRESS | CreateItemApiPOSTItemsPermissionP | AWS::Lambda::Permission     | rod                         |
| CREATE_IN_PROGRESS | GetItemApiGETItemsidPermissionPro | AWS::Lambda::Permission     | d                           |
| CREATE_IN_PROGRESS | UpdateItemApiPUTItemsidPermission | AWS::Lambda::Permission     | Prod                        |
| CREATE_IN_PROGRESS | CreateItemApiPOSTItemsPermissionP | AWS::Lambda::Permission     | rod                         |
| CREATE_IN_PROGRESS | ListItemsApiGETItemsPermissionPro | AWS::Lambda::Permission     | d                           |
| CREATE_IN_PROGRESS | DeleteItemApiDELETEItemsidPermiss | AWS::Lambda::Permission     | ionProd                     |
| CREATE_IN_PROGRESS | ApiDeploymentccc153d135b          | AWS::ApiGateway::Deployment |                             |
| CREATE_COMPLETE    | ApiDeploymentccc153d135b          | AWS::ApiGateway::Deployment |                             |
| CREATE_IN_PROGRESS | ApiProdStage                      | AWS::ApiGateway::Stage      |                             |
| CREATE_IN_PROGRESS | ApiProdStage                      | AWS::ApiGateway::Stage      | Resource creation Initiated |
| CREATE_COMPLETE    | ApiProdStage                      | AWS::ApiGateway::Stage      |                             |

```

CREATE_COMPLETE          AWS::Lambda::Permission
  CreateItemApiPOSTitemsPermissionP -
                                                                    rod
CREATE_COMPLETE          AWS::Lambda::Permission
  UpdateItemApiPUTitemsidPermission -
                                                                    Prod
CREATE_COMPLETE          AWS::Lambda::Permission
  ListItemsApiGETitemsPermissionPro -
                                                                    d
CREATE_COMPLETE          AWS::Lambda::Permission
  DeleteItemApiDELETEitemsidPermiss -
                                                                    ionProd
CREATE_COMPLETE          AWS::Lambda::Permission
  GetItemApiGETitemsidPermissionPro -
                                                                    d
CREATE_COMPLETE          AWS::CloudFormation::Stack
  composer-basic-api      -
                                                                    aws-app-
-----

```

Akhirnya, sebuah pesan ditampilkan, memberi tahu Anda bahwa penerapan berhasil:

```
Successfully created/updated stack - aws-app-composer-basic-api in us-west-2
```

## Gunakan Infrastructure Composer AWS SAM untuk menghapus tumpukan

Contoh ini menunjukkan cara menghapus AWS CloudFormation tumpukan menggunakan sam delete perintah.

Masukkan perintah sam delete di AWS SAM CLI dan konfirmasi apakah Anda ingin menghapus tumpukan dan template:

```

$ sam delete
Are you sure you want to delete the stack aws-app-composer-basic-api in the region us-west-2 ? [y/N]: y
Do you want to delete the template file 30439348c0be6e1b85043b7a935b34ab.template in S3? [y/N]: y
- Deleting S3 object with key eb226ca86d1bc4e9914ad85eb485fed8
- Deleting S3 object with key 875e4bcf4b10a6a1144ad83158d84b6d
- Deleting S3 object with key 20b869d98d61746dedd9aa33aa08a6fb

```

- Deleting S3 object with key c513cedc4db6bc184ce30e94602741d6
- Deleting S3 object with key c7a15d7d8d1c24b77a1eddf8caebc665
- Deleting S3 object with key e8b8984f881c3732bfb34257cdd58f1e
- Deleting S3 object with key 3185c59b550594ee7fca7f8c36686119.template
- Deleting S3 object with key 30439348c0be6e1b85043b7a935b34ab.template
- Deleting Cloudformation stack aws-app-composer-basic-api

Deleted successfully

# AWS Infrastructure Composer pemecahan masalah

Topik di bagian ini memberikan panduan tentang pemecahan masalah pesan kesalahan saat menggunakan AWS Infrastructure Composer

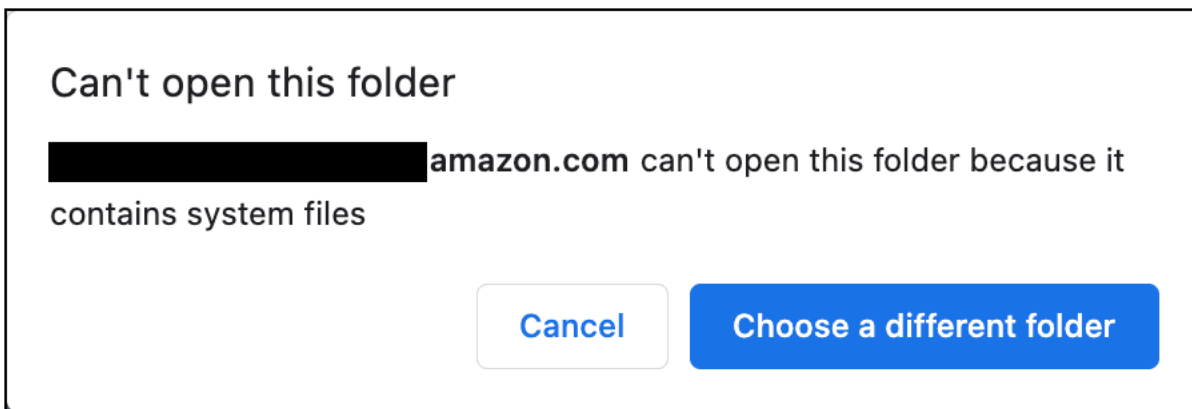
Topik

- [Pesan kesalahan](#)

## Pesan kesalahan

“Tidak dapat membuka folder ini”

Contoh kesalahan:



Kemungkinan penyebabnya: Infrastructure Composer tidak dapat mengakses direktori sensitif menggunakan mode sinkronisasi lokal.

Untuk mempelajari lebih lanjut tentang kesalahan ini, lihat [Data Infrastructure Composer mendapatkan akses ke](#).

Coba sambungkan ke direktori lokal yang berbeda atau gunakan Infrastructure Composer dengan sinkronisasi lokal dinonaktifkan.

“Template yang tidak kompatibel”

Contoh kesalahan: Saat memuat proyek baru di Infrastructure Composer, Anda melihat yang berikut:

Kemungkinan penyebabnya: Proyek Anda berisi file yang direferensikan secara eksternal yang tidak didukung di Infrastructure Composer.

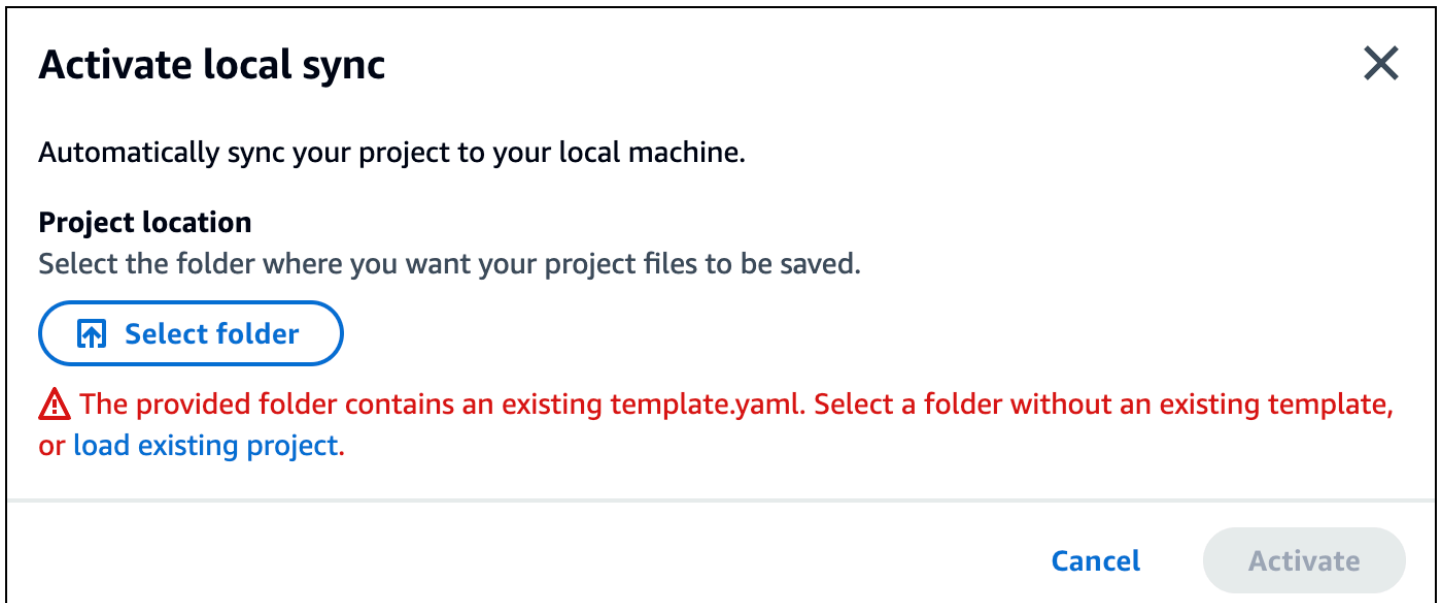
Untuk mempelajari tentang file eksternal yang didukung di Infrastructure Composer, lihat [Referensi file eksternal](#).

Kemungkinan penyebabnya: Proyek Anda menautkan ke file eksternal di direktori lokal yang berbeda.

Pindahkan file yang direferensikan secara eksternal ke subdirektori direktori yang Anda pilih untuk digunakan dengan mode sinkronisasi lokal Infrastructure Composer.

## “Folder yang disediakan berisi template.yaml yang ada”

Saat mencoba mengaktifkan sinkronisasi lokal, Anda melihat kesalahan berikut:



Kemungkinan penyebabnya: Folder yang Anda pilih sudah berisi file template.yaml.

Pilih direktori lain yang tidak berisi templat aplikasi, atau buat direktori baru.

“Browser Anda tidak memiliki izin untuk menyimpan proyek Anda di folder itu...”

Kemungkinan penyebabnya: Infrastructure Composer tidak dapat mengakses direktori sensitif menggunakan mode sinkronisasi lokal.

Untuk mempelajari lebih lanjut tentang kesalahan ini, lihat [Data Infrastructure Composer mendapatkan akses ke](#).

Coba sambungkan ke direktori lokal yang berbeda atau gunakan Infrastructure Composer dengan sinkronisasi lokal dinonaktifkan.

# Keamanan di AWS Infrastructure Composer

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. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari [Program AWS Kepatuhan Program AWS Kepatuhan](#) . Untuk mempelajari tentang program kepatuhan yang berlaku AWS Infrastructure Composer, lihat [AWS Layanan dalam Lingkup oleh AWS Layanan Program Kepatuhan](#) .
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, yang mencakup sensitivitas data Anda, persyaratan perusahaan Anda, serta undang-undang dan peraturan yang berlaku.

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan Infrastructure Composer. Topik berikut menunjukkan cara mengkonfigurasi Infrastructure Composer untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga mempelajari cara menggunakan AWS layanan lain yang membantu Anda memantau dan mengamankan sumber daya Infrastructure Composer Anda.

Topik

- [Perlindungan data di AWS Infrastructure Composer](#)
- [AWS Identity and Access Management untuk AWS Infrastructure Composer](#)
- [Validasi kepatuhan untuk AWS Infrastructure Composer](#)
- [Ketahanan di AWS Infrastructure Composer](#)

## Perlindungan data di AWS Infrastructure Composer

[Model tanggung jawab AWS bersama model tanggung](#) berlaku untuk perlindungan data di AWS Infrastructure Composer. 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. Untuk informasi selengkapnya tentang privasi data, lihat [Privasi Data FAQ](#). Untuk informasi tentang perlindungan data di Eropa, lihat [Model Tanggung Jawab AWS Bersama dan](#) posting GDPR blog di Blog AWS Keamanan.

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensial 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 otentikasi multi-faktor (MFA) dengan setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan AWS sumber daya. Kami membutuhkan TLS 1.2 dan merekomendasikan 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 FIPS 140-3 modul kriptografi yang divalidasi saat mengakses AWS melalui antarmuka baris perintah atau, gunakan titik akhir. API FIPS Untuk informasi selengkapnya tentang FIPS titik akhir yang tersedia, lihat [Federal Information Processing Standard \(FIPS\) 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 ketika Anda bekerja dengan Infrastructure Composer 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. Jika Anda memberikan URL ke server eksternal, kami sangat menyarankan agar Anda tidak menyertakan informasi kredensial dalam URL untuk memvalidasi permintaan Anda ke server tersebut.



**Note**

Semua data yang Anda masukkan ke Infrastructure Composer digunakan untuk tujuan tunggal menyediakan fungsionalitas dalam Infrastructure Composer dan menghasilkan file proyek dan direktori yang disimpan secara lokal ke mesin Anda. Infrastructure Composer tidak menyimpan, menyimpan, atau mengirimkan data ini.

## Enkripsi data

Infrastructure Composer tidak mengenkripsi konten pelanggan karena data tidak disimpan, disimpan atau ditransmisikan.

### Enkripsi diam

Infrastructure Composer tidak mengenkripsi konten pelanggan karena data tidak disimpan, disimpan atau ditransmisikan.

### Enkripsi bergerak

Infrastructure Composer tidak mengenkripsi konten pelanggan karena data tidak disimpan, disimpan atau ditransmisikan.

## Manajemen kunci

Infrastructure Composer tidak mendukung manajemen kunci karena konten pelanggan tidak disimpan, disimpan, atau ditransmisikan.

## Privasi lalu lintas antar jaringan

Infrastructure Composer tidak menghasilkan lalu lintas dengan klien dan aplikasi on-premise.

## AWS Identity and Access Management untuk AWS Infrastructure Composer

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya dengan aman. IAM administrator mengontrol siapa yang dapat diautentikasi (masuk) dan diberi wewenang (memiliki izin) untuk menggunakan sumber

daya Infrastructure Composer. IAM adalah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

## Topik

- [Audiens](#)
- [Mengautentikasi dengan identitas](#)
- [Mengelola akses menggunakan kebijakan](#)
- [Bagaimana AWS Infrastructure Composer bekerja dengan IAM](#)

## Audiens

Infrastructure Composer membutuhkan, setidaknya, akses hanya-baca ke file. AWS Management Console Setiap pengguna dengan otorisasi ini dapat menggunakan semua fitur Infrastructure Composer. Akses granular ke fitur spesifik Infrastructure Composer tidak didukung.

## Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensi identitas Anda. Anda harus diautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai IAM pengguna, atau dengan mengambil peran IAM.

Anda dapat masuk AWS sebagai identitas federasi dengan menggunakan kredensi yang disediakan melalui sumber identitas. AWS IAM Identity Center Pengguna (Pusat IAM Identitas), autentikasi masuk tunggal perusahaan Anda, dan kredensi Google atau Facebook Anda adalah contoh identitas federasi. Saat Anda masuk sebagai identitas federasi, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan IAM peran. Ketika Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil peran.

Bergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau portal AWS akses. Untuk informasi selengkapnya tentang masuk AWS, lihat [Cara masuk ke Panduan AWS Sign-In Pengguna Anda Akun AWS](#).

Jika Anda mengakses AWS secara terprogram, AWS sediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensial Anda. Jika Anda tidak menggunakan AWS alat, Anda harus menandatangani permintaan sendiri. Untuk informasi selengkapnya tentang menggunakan metode yang disarankan untuk menandatangani permintaan sendiri, lihat [Versi AWS Tanda Tangan 4 untuk API permintaan](#) di Panduan IAM Pengguna.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, lihat [Autentikasi multi-faktor](#) di Panduan AWS IAM Identity Center Pengguna dan [Autentikasi AWS multi-faktor IAM di Panduan Pengguna](#). IAM

## Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut pengguna Akun AWS root dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihat [Tugas yang memerlukan kredensi pengguna root](#) di IAMPanduan Pengguna.

## Identitas gabungan

Sebagai praktik terbaik, mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensi sementara.

Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, direktori Pusat Identitas AWS Directory Service, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensial yang disediakan melalui sumber identitas. Ketika identitas federasi mengakses Akun AWS, mereka mengambil peran, dan peran memberikan kredensi sementara.

Untuk manajemen akses terpusat, kami sarankan Anda menggunakan AWS IAM Identity Center. Anda dapat membuat pengguna dan grup di Pusat IAM Identitas, atau Anda dapat menghubungkan dan menyinkronkan ke sekumpulan pengguna dan grup di sumber identitas Anda sendiri untuk digunakan di semua aplikasi Akun AWS dan aplikasi Anda. Untuk informasi tentang Pusat IAM Identitas, lihat [Apa itu Pusat IAM Identitas?](#) dalam AWS IAM Identity Center User Guide.

## Pengguna dan grup IAM

[IAMPengguna](#) adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, sebaiknya mengandalkan kredensi sementara daripada

membuat IAM pengguna yang memiliki kredensi jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan khusus yang memerlukan kredensi jangka panjang dengan IAM pengguna, kami sarankan Anda memutar kunci akses. Untuk informasi selengkapnya, lihat [Memutar kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensi jangka panjang](#) di IAMPanduan Pengguna.

[IAMGrup](#) adalah identitas yang menentukan kumpulan IAM pengguna. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin bagi beberapa pengguna sekaligus. Grup mempermudah manajemen izin untuk sejumlah besar pengguna sekaligus. Misalnya, Anda dapat memiliki grup bernama IAMAdmins dan memberikan izin grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari selengkapnya, lihat [Kasus penggunaan untuk IAM pengguna](#) di Panduan IAM Pengguna.

## Peran IAM

[IAMPeran](#) adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Ini mirip dengan IAM pengguna, tetapi tidak terkait dengan orang tertentu. Untuk mengambil IAM peran sementara di dalam AWS Management Console, Anda dapat [beralih dari pengguna ke IAM peran \(konsol\)](#). Anda dapat mengambil peran dengan memanggil AWS CLI atau AWS API operasi atau dengan menggunakan kustom URL. Untuk informasi selengkapnya tentang metode penggunaan peran, lihat [Metode untuk mengambil peran](#) dalam Panduan IAM Pengguna.

IAMperan dengan kredensi sementara berguna dalam situasi berikut:

- Akses pengguna terfederasi – Untuk menetapkan izin ke identitas terfederasi, Anda membuat peran dan menentukan izin untuk peran tersebut. Ketika identitas terfederasi mengotentikasi, identitas tersebut terhubung dengan peran dan diberi izin yang ditentukan oleh peran. Untuk informasi tentang peran untuk federasi, lihat [Membuat peran untuk penyedia identitas pihak ketiga \(federasi\)](#) di Panduan IAM Pengguna. Jika Anda menggunakan Pusat IAM Identitas, Anda mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah diautentikasi, Pusat IAM Identitas mengkorelasikan izin yang disetel ke peran. Untuk informasi tentang set izin, lihat [Set izin](#) dalam Panduan Pengguna AWS IAM Identity Center .
- Izin IAM pengguna sementara — IAM Pengguna atau peran dapat mengambil IAM peran untuk sementara mengambil izin yang berbeda untuk tugas tertentu.

- Akses lintas akun — Anda dapat menggunakan IAM peran untuk memungkinkan seseorang (prinsipal tepercaya) di akun lain mengakses sumber daya di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, dengan beberapa Layanan AWS, Anda dapat melampirkan kebijakan langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Untuk mempelajari perbedaan antara peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Akses sumber daya lintas akun di IAM](#) Panduan Pengguna. IAM
- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Misalnya, saat Anda melakukan panggilan dalam suatu layanan, biasanya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Sebuah layanan mungkin melakukannya menggunakan izin prinsipal yang memanggil, menggunakan peran layanan, atau peran terkait layanan.
- Sesi akses teruskan (FAS) — Saat Anda menggunakan IAM pengguna atau peran untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan tindakan yang kemudian memulai 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. FAS permintaan 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 saat membuat FAS permintaan, lihat [Meneruskan sesi akses](#).
- Peran layanan — Peran layanan adalah [IAM peran](#) yang diasumsikan layanan untuk melakukan tindakan atas nama Anda. IAM Administrator dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Membuat peran untuk mendelegasikan izin ke Layanan AWS](#) dalam IAM Panduan Pengguna.
- Peran terkait layanan — 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 Akun AWS dan dimiliki oleh layanan. IAM Administrator dapat melihat, tetapi tidak mengedit izin untuk peran terkait layanan.
- Aplikasi yang berjalan di Amazon EC2 — Anda dapat menggunakan IAM peran untuk mengelola kredensial sementara untuk aplikasi yang berjalan pada EC2 instance dan membuat AWS CLI atau AWS API meminta. Ini lebih baik untuk menyimpan kunci akses dalam EC2 instance. Untuk menetapkan AWS peran ke EC2 instance dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instance yang dilampirkan ke instance. Profil instance berisi peran dan memungkinkan program yang berjalan pada EC2 instance untuk mendapatkan kredensial

sementara. Untuk informasi selengkapnya, lihat [Menggunakan IAM peran untuk memberikan izin ke aplikasi yang berjalan di EC2 instans Amazon](#) di IAMPanduan Pengguna.

## Mengelola akses menggunakan kebijakan

Anda mengontrol akses AWS dengan membuat kebijakan dan melampirkannya ke AWS identitas atau sumber daya. Kebijakan adalah objek AWS yang, ketika dikaitkan dengan identitas atau sumber daya, menentukan izinnya. AWS mengevaluasi kebijakan ini ketika prinsipal (pengguna, pengguna root, atau sesi peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai JSON dokumen. Untuk informasi selengkapnya tentang struktur dan isi dokumen JSON kebijakan, lihat [Ringkasan JSON kebijakan](#) di Panduan IAM Pengguna.

Administrator dapat menggunakan AWS JSON kebijakan untuk menentukan siapa yang memiliki akses ke apa. Yaitu, principal dapat melakukan tindakan pada suatu sumber daya, dan dalam suatu syarat.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan pada sumber daya yang mereka butuhkan, IAM administrator dapat membuat IAM kebijakan. Administrator kemudian dapat menambahkan IAM kebijakan ke peran, dan pengguna dapat mengambil peran.

Kebijakan IAM mendefinisikan izin untuk tindakan terlepas dari metode yang Anda gunakan untuk melakukan operasi. Misalnya, anggaplah Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan itu bisa mendapatkan informasi peran dari AWS Management Console, AWS CLI, atau AWS API.

### Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan JSON izin yang dapat Anda lampirkan ke identitas, seperti pengguna, grup IAM pengguna, atau peran. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat [Menentukan IAM izin khusus dengan kebijakan yang dikelola pelanggan di Panduan Pengguna](#). IAM

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan yang dikelola. Kebijakan inline disematkan langsung ke satu pengguna, grup, atau peran. Kebijakan terkelola adalah kebijakan mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran dalam. Akun AWS Kebijakan AWS terkelola mencakup kebijakan terkelola dan kebijakan

yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan terkelola atau kebijakan sebaris, lihat [Memilih antara kebijakan terkelola dan kebijakan sebaris](#) di IAMPanduan Pengguna.

## Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen JSON kebijakan yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan IAM peran 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

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan kebijakan AWS terkelola IAM dalam kebijakan berbasis sumber daya.

## Daftar kontrol akses (ACLs)

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

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung ACLs. Untuk mempelajari selengkapnya ACLs, lihat [Ikhtisar daftar kontrol akses \(ACL\)](#) di Panduan Pengembang Layanan Penyimpanan Sederhana Amazon.

## Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Jenis-jenis kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda oleh jenis kebijakan yang lebih umum.

- **Batas izin** — Batas izin adalah fitur lanjutan tempat Anda menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas (pengguna atau peran). IAM Anda dapat menetapkan batasan izin untuk suatu entitas. Izin yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas milik entitas dan batasan izinnya. Kebijakan berbasis sumber daya yang menentukan pengguna atau peran dalam bidang `Principal` tidak dibatasi oleh batasan izin. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya tentang batas izin, lihat [Batas izin untuk IAM entitas](#) di IAMPanduan Pengguna.

- Kebijakan kontrol layanan (SCPs) — SCPs adalah JSON kebijakan yang menentukan izin maksimum untuk organisasi atau unit organisasi (OU) di AWS Organizations. AWS Organizations adalah layanan untuk mengelompokkan dan mengelola secara terpusat beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur dalam suatu organisasi, maka Anda dapat menerapkan kebijakan kontrol layanan (SCPs) ke salah satu atau semua akun Anda. SCPMembatasi izin untuk entitas di akun anggota, termasuk masing-masing Pengguna root akun AWS. Untuk informasi selengkapnya tentang Organizations danSCPs, lihat [Kebijakan kontrol layanan](#) di Panduan AWS Organizations Pengguna.
- Kebijakan kontrol sumber daya (RCPs) — RCPs adalah JSON kebijakan yang dapat Anda gunakan untuk menetapkan izin maksimum yang tersedia untuk sumber daya di akun Anda tanpa memperbarui IAM kebijakan yang dilampirkan ke setiap sumber daya yang Anda miliki. RCPMembatasi izin untuk sumber daya di akun anggota dan dapat memengaruhi izin efektif untuk identitas, termasuk Pengguna root akun AWS, terlepas dari apakah itu milik organisasi Anda. Untuk informasi selengkapnya tentang Organizations danRCPs, termasuk daftar dukungan Layanan AWS tersebutRCPs, lihat [Kebijakan kontrol sumber daya \(RCPs\)](#) di Panduan AWS Organizations Pengguna.
- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter ketika Anda membuat sesi sementara secara programatis untuk peran atau pengguna terfederasi. Izin sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin. Untuk informasi selengkapnya, lihat [Kebijakan sesi](#) di Panduan IAM Pengguna.

## Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari cara AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat [Logika evaluasi kebijakan](#) di Panduan IAM Pengguna.

## Bagaimana AWS Infrastructure Composer bekerja dengan IAM

AWS Infrastructure Composer membutuhkan, setidaknya, akses hanya-baca ke. AWS Management Console Setiap pengguna dengan otorisasi ini dapat menggunakan semua fitur Infrastructure Composer. Akses granular ke fitur spesifik Infrastructure Composer tidak didukung.

Ketika Anda menyebarkan template dan file proyek Anda AWS CloudFormation, Anda akan memerlukan izin yang diperlukan untuk berada di tempat. Untuk mempelajari lebih lanjut, lihat



[Mengontrol akses dengan AWS Identity and Access Management](#) di Panduan AWS CloudFormation Pengguna.

Tabel berikut menunjukkan IAM fitur apa yang dapat digunakan AWS Infrastructure Composer.

| IAMfitur                                       | Dukungan Komposer Infrastruktur |
|--|---------------------------------|
| <a href="#">Kebijakan berbasis identitas</a>   | Tidak                           |
| <a href="#">Kebijakan berbasis sumber daya</a> | Tidak                           |
| <a href="#">Tindakan kebijakan</a>             | Tidak                           |
| <a href="#">Sumber daya kebijakan</a>          | Tidak                           |
| <a href="#">Kunci kondisi kebijakan</a>        | Tidak                           |
| <a href="#">ACLs</a>                           | Tidak                           |
| <a href="#">ABAC(tag dalam kebijakan)</a>      | Tidak                           |
| <a href="#">Kredensial sementara</a>           | Ya                              |
| <a href="#">Izin principal</a>                 | Tidak                           |
| <a href="#">Peran layanan</a>                  | Tidak                           |
| <a href="#">Peran terkait layanan</a>          | Tidak                           |

Untuk mendapatkan tampilan tingkat tinggi tentang cara kerja Infrastructure Composer dan AWS layanan lainnya dengan sebagian besar IAM fitur, lihat [AWS layanan yang berfungsi IAM](#) di IAMPanduan Pengguna.

## Kebijakan berbasis identitas untuk Infrastructure Composer

Mendukung kebijakan berbasis identitas: Tidak

Kebijakan berbasis identitas adalah dokumen kebijakan JSON izin yang dapat Anda lampirkan ke identitas, seperti pengguna, grup IAM pengguna, atau peran. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi

seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat [Menentukan IAM izin khusus dengan kebijakan yang dikelola pelanggan di Panduan Pengguna](#). IAM

Dengan kebijakan IAM berbasis identitas, Anda dapat menentukan tindakan dan sumber daya yang diizinkan atau ditolak serta kondisi di mana tindakan diizinkan atau ditolak. 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 JSON kebijakan, lihat [referensi elemen IAM JSON kebijakan](#) di Panduan IAM Pengguna.

## Kebijakan berbasis sumber daya dalam Infrastructure Composer

Mendukung kebijakan berbasis sumber daya: Tidak

Kebijakan berbasis sumber daya adalah dokumen JSON kebijakan yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan IAM peran 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 seluruh akun atau entitas IAM di akun lain sebagai penanggung jawab 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, IAM administrator 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 IAM di](#) Panduan IAM Pengguna.

## Tindakan kebijakan untuk Komposer Infrastruktur

Mendukung tindakan kebijakan: Tidak

Administrator dapat menggunakan AWS JSON kebijakan untuk menentukan siapa yang memiliki akses ke apa. Yaitu, principal dapat melakukan tindakan pada suatu sumber daya, dan dalam suatu syarat.

ActionElement JSON kebijakan menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan AWS API operasi terkait. Ada beberapa pengecualian, seperti tindakan khusus izin yang tidak memiliki operasi yang cocok. API 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 Infrastructure Composer, lihat [Actions Defined by AWS Infrastructure Composer di Referensi](#) Otorisasi Layanan.

## Sumber daya kebijakan untuk Komposer Infrastruktur

Mendukung sumber daya kebijakan: Tidak

Administrator dapat menggunakan AWS JSON kebijakan untuk menentukan siapa yang memiliki akses ke apa. Yaitu, principal dapat melakukan tindakan pada suatu sumber daya, dan dalam suatu syarat.

Elemen Resource JSON kebijakan menentukan objek atau objek yang tindakan tersebut berlaku. Pernyataan harus menyertakan elemen Resource atau NotResource. Sebagai praktik terbaik, tentukan sumber daya menggunakan [Amazon Resource Name \(ARN\)](#). Anda dapat melakukan ini untuk tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, misalnya operasi pencantuman, gunakan wildcard (\*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

```
"Resource": "*" 
```

Untuk melihat daftar jenis sumber daya Infrastructure Composer dan jenisnya ARNs, lihat [Resources Defined by AWS Infrastructure Composer di Referensi](#) Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat [Tindakan yang Ditentukan oleh Komposer AWS Infrastruktur](#).

## Kunci kondisi kebijakan untuk Komposer Infrastruktur

Mendukung kunci kondisi kebijakan khusus layanan: Tidak

Administrator dapat menggunakan AWS JSON kebijakan untuk menentukan siapa yang memiliki akses ke apa. Yaitu, di mana utama dapat melakukan tindakan pada sumber daya, dan dalam kondisi apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan kondisi yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan [operator kondisi](#), misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa elemen `Condition` dalam sebuah pernyataan, atau beberapa kunci dalam elemen `Condition` tunggal, maka AWS akan mengevaluasinya menggunakan operasi AND logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS mengevaluasi kondisi menggunakan OR operasi logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Misalnya, Anda dapat memberikan izin IAM pengguna untuk mengakses sumber daya hanya jika ditandai dengan nama IAM pengguna mereka. Untuk informasi selengkapnya, lihat [elemen IAM kebijakan: variabel dan tag](#) di Panduan IAM Pengguna.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua kunci kondisi AWS global, lihat [kunci konteks kondisi AWS global](#) di Panduan IAM Pengguna.

Untuk melihat daftar kunci kondisi Infrastructure Composer, lihat [Condition Keys for AWS Infrastructure Composer di Referensi](#) Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan kunci kondisi, lihat [Tindakan yang Ditentukan oleh Komposer AWS Infrastruktur](#).

## ACLs di Infrastructure Composer

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 Infrastructure Composer

Mendukung ABAC (tag dalam kebijakan): Tidak

Attribute-based access control (ABAC) adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut. Dalam AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke IAM entitas (pengguna atau peran) dan ke banyak AWS sumber daya. Menandai entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian Anda merancang ABAC kebijakan untuk mengizinkan operasi ketika tag prinsipal cocok dengan tag pada sumber daya yang mereka coba akses.

ABAC membantu dalam lingkungan yang berkembang pesat dan membantu dengan situasi di mana manajemen kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tanda, berikan informasi tentang tanda di [elemen kondisi](#) dari kebijakan menggunakan kunci kondisi `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, atau `aws:TagKeys`.

Jika sebuah layanan mendukung ketiga kunci kondisi untuk setiap jenis sumber daya, nilainya adalah Ya untuk layanan tersebut. Jika suatu layanan mendukung ketiga kunci kondisi untuk hanya beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi selengkapnya ABAC, lihat [Menentukan izin dengan ABAC otorisasi](#) di IAM Panduan Pengguna. Untuk melihat tutorial dengan langkah-langkah penyiapan ABAC, lihat [Menggunakan kontrol akses berbasis atribut \(ABAC\)](#) di IAM Panduan Pengguna.

## Menggunakan kredensi sementara dengan Infrastructure Composer

Mendukung kredensial sementara: Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensi sementara. Untuk informasi tambahan, termasuk yang Layanan AWS bekerja dengan kredensial sementara, lihat [Layanan AWS yang berfungsi IAM](#) di IAM Panduan Pengguna.

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 link sign-on (SSO) tunggal perusahaan Anda, proses tersebut secara otomatis membuat kredensi sementara. Anda juga akan secara otomatis membuat kredensial sementara ketika Anda masuk ke konsol sebagai seorang pengguna lalu beralih peran. Untuk informasi selengkapnya tentang beralih peran, lihat [Beralih dari pengguna ke IAM peran \(konsol\)](#) di Panduan IAM Pengguna.

Anda dapat secara manual membuat kredensi sementara menggunakan `awscli` atau `awscli` API. Anda kemudian dapat menggunakan kredensi sementara tersebut untuk mengakses AWS. AWS merekomendasikan agar Anda secara dinamis menghasilkan kredensi sementara alih-alih

menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat [Kredensi keamanan sementara](#) di IAM

Anda dapat menggunakan kredensi sementara untuk mengakses Infrastructure Composer melalui file. AWS Management Console Sebagai contoh, lihat [Mengaktifkan akses broker identitas kustom ke AWS konsol](#) di Panduan IAM Pengguna.

## Izin utama lintas layanan untuk Komposer Infrastruktur

Mendukung sesi akses maju (FAS): Tidak

Saat Anda menggunakan IAM pengguna atau peran untuk melakukan tindakan AWS, Anda dianggap sebagai prinsipal. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan tindakan yang kemudian memulai 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. FAS permintaan 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 saat membuat FAS permintaan, lihat [Meneruskan sesi akses](#).

## Peran layanan untuk Komposer Infrastruktur

Mendukung peran layanan: Tidak

Peran layanan adalah [IAM peran](#) yang diasumsikan layanan untuk melakukan tindakan atas nama Anda. IAM Administrator dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Membuat peran untuk mendelegasikan izin ke Layanan AWS](#) dalam IAMPanduan Pengguna.

### Warning

Mengubah izin untuk peran layanan dapat merusak fungsionalitas Infrastructure Composer. Edit peran layanan hanya ketika Infrastructure Composer memberikan panduan untuk melakukannya.

## Peran terkait layanan untuk Komposer Infrastruktur

Mendukung peran terkait layanan: Tidak

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. IAM Administrator dapat melihat, tetapi tidak mengedit izin untuk peran terkait layanan.

Untuk detail tentang membuat atau mengelola peran terkait layanan, lihat [AWS layanan yang berfungsi](#) dengannya. 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.

## Validasi kepatuhan untuk AWS Infrastructure Composer

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.
- [Arsitektur untuk HIPAA Keamanan dan Kepatuhan di Amazon Web Services](#) — Whitepaper ini menjelaskan bagaimana perusahaan dapat menggunakan AWS untuk membuat HIPAA aplikasi yang memenuhi syarat.

### Note

Tidak semua Layanan AWS HIPAA memenuhi syarat. Untuk informasi selengkapnya, lihat [Referensi Layanan yang HIPAA Memenuhi Syarat](#).

- [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 mencurigakan dan berbahaya. GuardDuty dapat membantu Anda mengatasi berbagai persyaratan kepatuhan, seperti PCIDSS, dengan memenuhi persyaratan deteksi intrusi yang diamanatkan oleh kerangka kerja kepatuhan tertentu.
- [AWS Audit Manager](#) Ini Layanan AWS membantu Anda terus mengaudit AWS penggunaan Anda untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap peraturan dan standar industri.

## Ketahanan di AWS Infrastructure Composer

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](#).



Semua data yang Anda masukkan ke Infrastructure Composer digunakan untuk tujuan tunggal menyediakan fungsionalitas dalam Infrastructure Composer dan menghasilkan file proyek dan direktori yang disimpan secara lokal ke mesin Anda. Infrastructure Composer tidak menyimpan atau menyimpan data ini.

# Sejarah dokumen untuk Komposer Infrastruktur

Tabel berikut menjelaskan rilis dokumentasi penting untuk Infrastructure Composer. Untuk pemberitahuan tentang pembaruan dokumentasi ini, Anda dapat berlangganan RSS umpan.

- Pembaruan dokumentasi terbaru: 30 November 2023

| Perubahan   | Deskripsi  | Tanggal          |
|---|--|------------------|
| <a href="#">Konten yang direstrukturisasi dan diperbarui di seluruh panduan pengembang</a>  | Menata ulang dan merestrukturisasi panduan untuk meningkatkan kemampuan ditemukan dan kegunaan. Judul yang diperbarui dan ditingkatkan. Memberikan detail tambahan saat memperkenalkan topik dan konsep.   | Agustus 1, 2024  |
| <a href="#">Menambahkan dokumentasi untuk menggunakan Infrastructure Composer dalam mode CloudFormation konsol dan merestrukturisasi Infrastructure Composer Developer Guide.</a> | AWS Infrastructure Composer sekarang dapat digunakan dalam mode AWS CloudFormation konsol. Untuk mempelajari lebih lanjut, lihat <a href="#">Menggunakan Infrastructure Composer dalam mode CloudFormation konsol</a> . Selain itu, banyak konten dalam panduan pengguna telah ditata ulang untuk menciptakan pengalaman yang efisien. | Maret 28, 2024   |
| <a href="#">Ditambahkan dokumentasi untuk integrasi Infrastructure Composer dengan CodeWhisperer</a>  | AWS Infrastructure Composer dari Toolkit for VS Code menyediakan integrasi dengan Amazon CodeWhisperer.  | 30 November 2023 |

---

|  |   |                         |
|--|---|-------------------------|
|  | <p>Untuk mempelajari lebih lanjut, lihat <a href="#">Menggunakan AWS Infrastructure Composer dengan Amazon CodeWhisperer</a>.</p>   |                         |
| <p><a href="#">Menambahkan dokumentasi untuk menerapkan aplikasi Anda dengan Infrastructure Composer dari AWS Toolkit for Visual Studio Code</a></p> | <p>Gunakan tombol sinkronisasi dari kanvas Infrastructure Composer untuk menyebarkan aplikasi Anda ke AWS Cloud. Untuk mempelajari lebih lanjut, lihat <a href="#">Menerapkan aplikasi Anda dengan sam sync</a>.</p>  | <p>30 November 2023</p> |
| <p><a href="#">Ditambahkan dokumentasi untuk Infrastructure Composer dari AWS Toolkit for Visual Studio Code</a></p>                                 | <p>Anda sekarang dapat menggunakan Infrastructure Composer dari VS Code dengan file. AWS Toolkit for Visual Studio Code. Untuk mempelajari lebih lanjut, lihat <a href="#">Menggunakan AWS Infrastructure Composer dari AWS Toolkit for Visual Studio Code</a>.</p> | <p>30 November 2023</p> |
| <p><a href="#">Menambahkan integrasi Step Functions Workflow Studio</a></p>  | <p>Luncurkan Step Functions Workflow Studio dari kanvas Infrastructure Composer. Untuk mempelajari lebih lanjut, lihat <a href="#">Menggunakan AWS Infrastructure Composer dengan AWS Step Functions</a>.</p>   | <p>27 November 2023</p> |

[Ditambahkan konsol Lambda dan integrasi Infrastruktur Komposer](#)

Luncurkan kanvas Infrastructure Composer dari konsol Lambda. Untuk mempelajari lebih lanjut, lihat [Menggunakan AWS Infrastructure Composer dengan AWS Lambda konsol](#).

14 November 2023

[Ditambahkan Amazon VPC sebagai layanan unggulan dengan Infrastructure Composer](#)

Infrastructure Composer memperkenalkan VPC tag untuk memvisualisasikan sumber daya yang dikonfigurasi dengan file. VPC Anda juga dapat mengonfigurasi fungsi Lambda dengan VPCs didefinisikan pada template eksternal. Untuk mempelajari lebih lanjut, lihat [Menggunakan Infrastructure Composer dengan Amazon VPC](#).

17 Oktober 2023

[Ditambahkan Amazon RDS sebagai layanan unggulan dengan Infrastructure Composer](#)

Hubungkan aplikasi Infrastructure Composer Anda ke cluster Amazon RDS DB atau instance yang ditentukan pada template eksternal. Untuk mempelajari lebih lanjut, lihat [Menggunakan Infrastructure Composer dengan Amazon RDS](#).

17 Oktober 2023

[Ditambahkan dukungan Infrastructure Composer untuk merancang dengan semua sumber daya AWS CloudFormation](#)

Pilih AWS CloudFormation sumber daya apa pun dari palet Resources untuk mendesain aplikasi Anda. Untuk mempelajari lebih lanjut, lihat [Bekerja dengan AWS CloudFormation sumber daya apa pun](#).

26 September 2023

[Ditambahkan dokumentasi untuk kartu di Infrastructure Composer](#)

Infrastructure Composer mendukung beberapa jenis kartu yang dapat Anda gunakan untuk merancang dan membangun aplikasi Anda. Untuk mempelajari lebih lanjut, lihat [Mendesain dengan kartu di Infrastructure Composer](#).

20 September 2023

[Ditambahkan dokumentasi untuk fitur undo dan redo](#)

Gunakan tombol undo dan redo pada kanvas Infrastructure Composer. Untuk mempelajari lebih lanjut, lihat [Membatalkan dan mengulang](#).

1 Agustus 2023

[Ditambahkan dokumentasi untuk modus sinkronisasi lokal](#)

Gunakan mode sinkronisasi lokal untuk secara otomatis menyinkronkan dan menyimpan proyek Anda ke mesin lokal Anda. Untuk mempelajari lebih lanjut, lihat [Mode sinkronisasi lokal](#).

1 Agustus 2023

[Ditambahkan dokumentasi untuk fitur kanvas ekspor](#)

Gunakan fitur ekspor kanvas untuk mengeksport kanvas aplikasi Anda sebagai gambar ke mesin lokal Anda. Untuk mempelajari selengkapnya, lihat [Mengekspor kanvas](#).

1 Agustus 2023

[Dukungan Komposer Infrastruktur untuk referensi file eksternal](#)

Referensi file eksternal untuk sumber daya yang didukung di Infrastructure Composer. Untuk mempelajari lebih lanjut, lihat [Bekerja dengan templat yang mereferensikan file eksternal](#).

17 Mei 2023

[Dokumentasi baru tentang menghubungkan sumber daya](#)

Hubungkan sumber daya bersama-sama untuk menentukan hubungan berbasis peristiwa antar sumber daya dalam aplikasi Anda. Untuk mempelajari lebih lanjut, lihat [Menghubungkan sumber daya bersama menggunakan kanvas visual Infrastructure Composer](#).

7 Maret 2023

[Fitur baru Change Inspector](#)

Gunakan Change Inspector untuk melihat pembaruan kode template Anda dan pelajari apa yang dibuat oleh Infrastructure Composer untuk Anda. Untuk mempelajari selengkapnya, lihat [Melihat pembaruan kode dengan Change Inspector](#).

7 Maret 2023

[Komposer Infrastruktur sekarang tersedia secara umum](#)

AWS Infrastructure Composer sekarang tersedia secara umum. Untuk mempelajari lebih lanjut, lihat [AWS Infrastructure Composer sekarang tersedia secara umum - Membangun aplikasi tanpa server secara visual](#) dengan cepat.

7 Maret 2023

[Diperluas pada manfaat menggunakan mode terhubung](#)

Gunakan Infrastructure Composer dalam mode terhubung dengan lokal Anda IDE untuk mempercepat pengembangan. Untuk mempelajari lebih lanjut, lihat [Menggunakan Infrastructure Composer dengan lokal IDE Anda](#).

7 Maret 2023

[Topik terbaru tentang penggunaan AWS layanan lain untuk menyebarkan aplikasi Anda](#)

Gunakan Infrastructure Composer untuk merancang aplikasi tanpa server yang siap digunakan. Gunakan AWS SAM untuk menyebarkan aplikasi tanpa server Anda. Untuk mempelajari lebih lanjut, lihat [Menggunakan Infrastructure Composer dengan AWS CloudFormation dan AWS SAM](#).

3 Maret 2023

[Ditambahkan bagian konsep tanpa server](#)

Pelajari tentang konsep dasar tanpa server sebelum menggunakan Infrastructure Composer. Untuk mempelajari lebih lanjut, lihat [Konsep tanpa server](#).

2 Maret 2023

[Rilis publik](#)

Rilis publik awal Infrastructure Composer.

Desember 1, 2022



Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.