

Semana do Desenvolvedor – Dia 1

Aula 1: Ingestão de Pedidos via API e EventBridge

Documentação do Laboratório: Arquitetura de Processamento de Pedidos e Arquivos na AWS

(Duração Estimada: 160 minutos)

1. Visão Geral e Objetivos da Aula

Este laboratório marca o início da semana do desenvolvedor AWS da Escola da Nuvem, voltada para alunos do curso AWS Developer Associate. Nesta sessão, construiremos o fluxo inicial para a ingestão de pedidos. Nosso objetivo é criar um endpoint de API REST que receberá dados de pedidos, passará por uma Lambda de pré-validação, será enfileirado em uma fila SQS FIFO (First-In, First-Out) para garantir a ordem, e então processado por uma segunda Lambda que validará o pedido mais a fundo e publicará um evento em um barramento de eventos customizado no Amazon EventBridge.

Ao final desta aula, você terá:

1. Um endpoint API funcional para receber pedidos.
2. Duas funções Lambda para validação.
3. Uma fila SQS FIFO para desacoplar o processamento.
4. Um barramento de eventos customizado para notificar sobre novos pedidos validados.

Recursos a Serem Criados Nesta Aula:

- IAM Role: lambda-prevalidacao-role-seu-nome
- IAM Role: lambda-validacao-pedidos-role-seu-nome
- Amazon SQS FIFO DLQ: pedidos-fifo-dlq-seu-nome.fifo
- Amazon SQS FIFO Queue: pedidos-fifo-queue-seu-nome.fifo
- AWS Lambda Function: pre-validacao-lambda-seu-nome
- Amazon API Gateway (REST API): pedidos-api-seu-nome
- Amazon EventBridge Custom Event Bus: pedidos-event-bus-seu-nome

- AWS Lambda Function: validacao-pedidos-lambda-seu-nome

Lembrete Importante: Substitua seu-nome em todos os nomes de recursos pelo seu nome ou um identificador único (**ex:** validacao-pedidos-lambda -joaosilva) para evitar conflitos. Trabalhe consistentemente na mesma região AWS.

2. Configuração de Permissões (IAM Roles)

Para que nossos serviços AWS possam interagir uns com os outros de forma segura, precisamos criar Roles no IAM (Identity and Access Management) com as permissões necessárias.

2.1. Criar Role para a Lambda de Pré-Validação (lambda-prevalidacao-role-seu-nome)

Esta role será usada pela nossa primeira função Lambda, que fará a pré-validação e enviará a mensagem para a fila SQS.

1. Acesse o serviço **IAM** no Console AWS.
2. No menu à esquerda, clique em **"Roles"** e depois em **"Create role"**.
3. Em **"Trusted entity type"**, selecione **"AWS service"**.
4. Em **"Use case"**, selecione **"Lambda"** e clique em **"Next"**.
5. Na página **"Add permissions"**:
 - Procure pela política AWSLambdaBasicExecutionRole e marque a caixa de seleção ao lado dela. Esta política permite que a Lambda escreva logs no CloudWatch.
 - Clique em **"Next"**.
6. Em **"Role details"**:
 - **Role name:** lambda-prevalidacao-role-seu-nome
 - **Description:** (Opcional) Adicione uma breve descrição, como "Role para Lambda de pre_validacao de pedidos".
7. Revise as configurações e clique em **"Create role"**.
 - *Nota: Posteriormente, após criar a fila SQS, voltaremos a esta role para adicionar a permissão específica de envio de mensagens para a fila.*

2.2. Criar Role para a Lambda de Validação de Pedidos (lambda-validacao-pedidos-role-seu-nome)

Esta role será usada pela nossa segunda função Lambda, que consumirá mensagens da fila SQS e publicará eventos no EventBridge.

1. Ainda no **IAM**, clique em **"Roles"** e depois em **"Create role"**.
2. **Trusted entity type:** Selecione **"AWS service"**.
3. **Use case:** Selecione **"Lambda"** e clique em **"Next"**.
4. **Permissions policies:**
 - Procure e marque **AWSLambdaBasicExecutionRole**.
 - Clique em **"Next"**.
5. **Role name:** **lambda-validacao-pedidos-role-seu-nome**.
6. Clique em **"Create role"**.
 - *Nota: Após criar a fila SQS e o Event Bus, retornaremos aqui para adicionar as permissões específicas.*

3. Configuração do Enfileiramento de Mensagens (Amazon SQS FIFO)

Usaremos uma fila SQS FIFO para garantir que os pedidos sejam processados na ordem em que chegam e para desacoplar a API da lógica de validação.

3.1. Criar a Fila de Mensagens Mortas (DLQ - Dead Letter Queue)

A DLQ armazenará mensagens que não puderam ser processadas com sucesso pela fila principal após um número configurado de tentativas.

1. Acesse o serviço **Amazon SQS** no Console AWS.
2. Clique em **"Create queue"**.
3. **Type:** Selecione **"FIFO"**.
4. **Name:** **pedidos-fifo-dlq-seu-nome.fifo** (o sufixo **.fifo** é obrigatório).
5. Mantenha as demais configurações padrão por agora e clique em **"Create queue"**.
6. **Anote o ARN** desta DLQ. Ele será necessário ao criar a fila principal.

3.2. Criar a Fila Principal de Pedidos (pedidos-fifo-queue-seu-nome.fifo)

1. Ainda no **SQS**, clique em **"Create queue"** novamente.
2. **Type:** Selecione **"FIFO"**.
3. **Name:** **pedidos-fifo-queue-seu-nome.fifo**.

4. Na seção "**Dead-letter queue (DLQ)**":
 - Marque a caixa "**Enabled**".
 - Em "**Amazon Resource Name (ARN) of the dead-letter queue**", selecione o ARN da DLQ pedidos-fifo-dlq-seu-nome.fifo que você criou no passo 3.1.
 - **Maximum receives**: Defina como 3. Isso significa que uma mensagem será tentada 3 vezes antes de ser enviada para a DLQ.
5. Mantenha as demais configurações padrão e clique em "**Create queue**".
6. **Anote o ARN e a URL** desta fila principal. Eles serão usados nas configurações das Lambdas e IAM Roles.

3.3. Atualizar Permissões da Role de Pré-Validação

Agora que a fila SQS principal está criada, vamos conceder à lambda-prevalidacao-role-seu-nome a permissão para enviar mensagens a ela.

1. Volte ao serviço **IAM**.
2. Clique em "**Roles**" e encontre a role lambda-prevalidacao-role-seu-nome. Clique nela.
3. Na aba "**Permissions**", sob "Permissions policies", clique em "**Add permissions**" e selecione "**Create inline policy**".
4. Selecione a aba **JSON**.
5. Cole o seguinte JSON que você encontra [clikando aqui](#), substituindo o campo selecionado na imagem abaixo, pelo ARN da sua fila SQS principal:

Antes

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sqs:SendMessage",
      "Resource": "arn:aws:sqs:REGION:ACCOUNT_ID:pedidos-fifo-queue-seu-nome.fifo"
    }
  ]
}
```

Depois

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "sqs:SendMessage",
      "Resource": "arn:aws:sqs:us-east-1:237482015336:pedidos-fifo-queue-seu-nome.fifo"
    }
  ]
}
```

6. Clique em **"Next"**.
7. Em **"Review and create"** > **Policy details** > campo **Name**: SQSSendMessageToPedidosFIFO-seu-nome.
8. Clique em **"Create policy"**.

4. Implementação da Lógica de Pré-Validação (AWS Lambda)

Esta função Lambda será acionada pelo API Gateway, fará uma verificação inicial dos dados do pedido e, se válida, enviará para a fila SQS.

4.1. Criar a Função Lambda (pre-validacao-lambda-seu-nome)

1. Acesse o serviço **AWS Lambda** no Console AWS.
2. No menu à esquerda, clique em **"Functions"** e depois em **"Create function"**.
3. Selecione a opção **"Author from scratch"**.
4. **Function name**: pre-validacao-lambda-seu-nome.
5. **Runtime**: Selecione **Python 3.9 (13.12)** (ou uma versão mais recente suportada).
6. **Architecture**: Mantenha o padrão x86_64.
7. Em **"Permissions"**, expanda "Change default execution role".
 - Selecione **"Use an existing role"**.
 - Na lista suspensa, escolha a role lambda-prevalidacao-role-seu-nome que você criou.
8. Clique em **"Create function"**.

4.2. Configurar o Código e Variáveis de Ambiente

1. Na página da função Lambda, navegue até a aba **"Code"**.

2. No editor de código inline para o arquivo `lambda_function.py`, substitua o conteúdo existente pelo código Python que você encontra, [clcando aqui](#).
 3. Clique no botão **"Deploy"** (acima do editor de código) para salvar as alterações.
 4. Navegue até a aba **"Configuration"** e selecione **"Environment variables"** no menu à esquerda.
 5. Clique em **"Edit"**.
 6. Clique em **"Add environment variable"**.
 - **Key:** `SQS_QUEUE_URL`
 - **Value:** Cole a **URL** da fila `pedidos-fifo-queue-seu-nome.fifo` (anotada anteriormente).
 7. Clique em **"Save"**.
-

5. Criação do Ponto de Entrada (Amazon API Gateway)

Configuraremos um API Gateway para expor um endpoint REST que acionará nossa Lambda de pré-validação.

5.1. Criar a API REST (pedidos-api-seu-nome)

1. Acesse o serviço **Amazon API Gateway** no Console AWS.
2. No menu à esquerda, clique em **"APIs"** e depois em **"Create API"**.
3. Em "Choose an API type", selecione **"REST API"** clicando em **Build**.
4. **API details:**
 - Selecione **"New API"**.
 - **API name:** `pedidos-api-seu-nome`
 - **Description:** (Opcional) Ex: "API para recebimento de pedidos do laboratório".
 - **Endpoint Type:** Selecione **"Regional"**.
5. Clique em **"Create API"**.

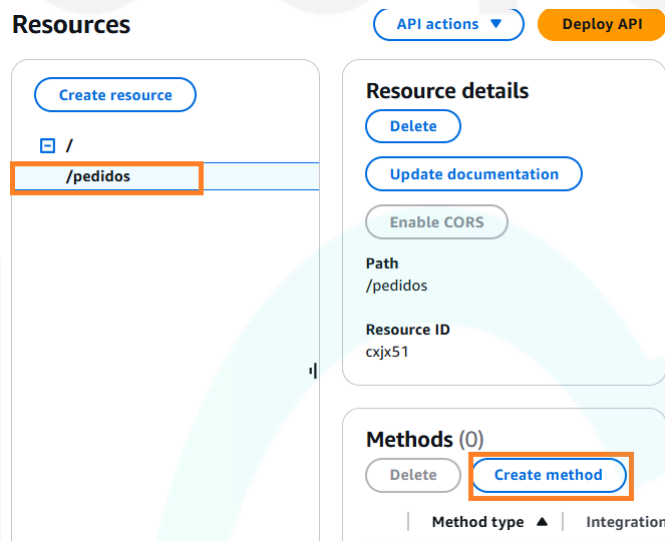
5.2. Configurar o Recurso e Método POST

1. Com sua nova API `pedidos-api-seu-nome` selecionada, no menu lateral, selecione **"Resource"** e escolha **"Create Resource"**.

2. Em **Resource details**:

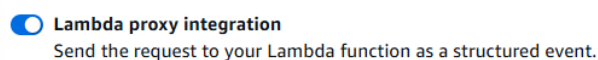
- **Resource Name:** pedidos
- (O "Resource Path" será preenchido automaticamente como /pedidos)
- Clique em **"Create Resource"**.

3. Com o recurso /pedidos selecionado na tela de recursos à direita, clique em **"Create Method"**:



4. Na página **"Create method"** do método POST:

- No menu suspenso **"Method type"**, selecione **POST**.
- **Integration type:** Selecione **"Lambda Function"**.
- Selecione **"Use Lambda Proxy integration"**.

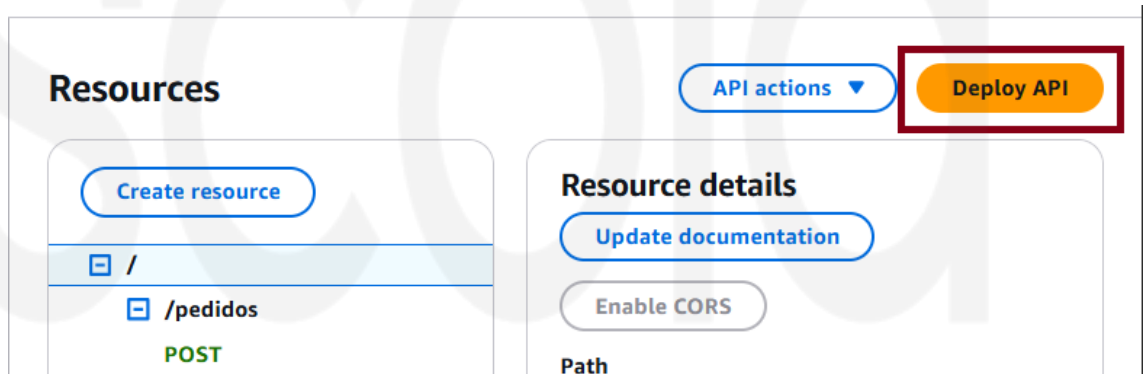


- **Lambda Region:** Certifique-se de que a região correta está selecionada (a região que você está utilizando nesse laboratório).
- **Lambda Function:** Comece a digitar pre-validacao-lambda-seu-nome e selecione sua função Lambda na lista que aparece.
- Clique em **"Create method"**.

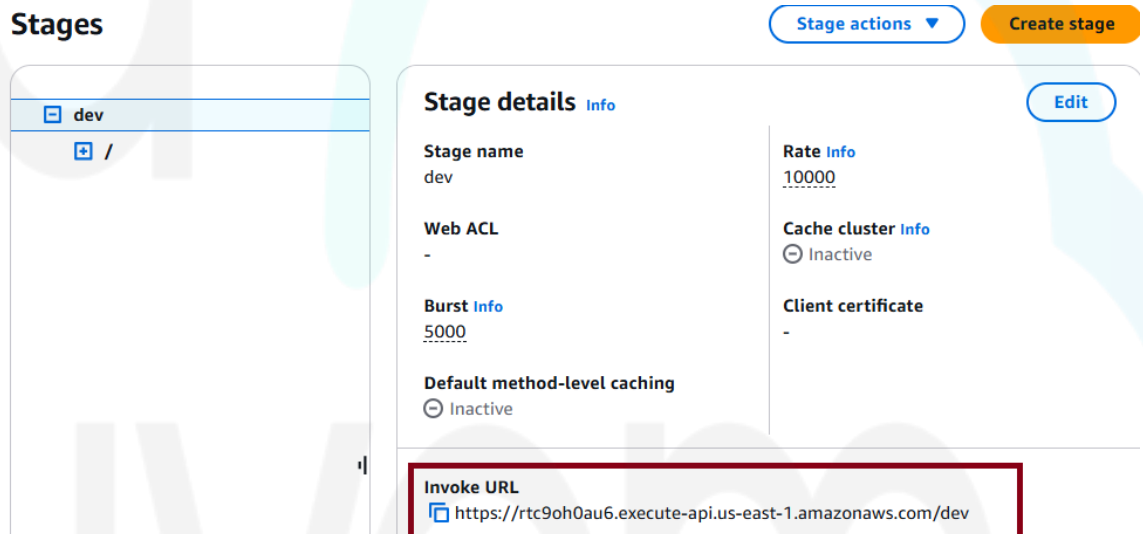
5.3. Deploy da API

Para que a API possa ser acessada externamente, precisamos fazer o deploy dela em um "stage".

1. Com sua API selecionada, clique em **"Deploy API"** no canto direito superior.



2. Na caixa de diálogo "Deploy API":
 - **Stage:** Selecione "[New Stage]".
 - **Stage name:** dev.
 - **Deployment description:** Estágio Dev. Versão 1 (Opcional).
 - Clique em **"Deploy"**.
3. Após o deploy, você será levado para a página do "Stage Editor". Expanda os detalhes do seu stage, clicando no sinal de "+":



4. **Anote a URL de Invocação (Invoke URL).** Ela será usada para testar a API.
Exemplo: `https://abcdef123.execute-api.us-east-1.amazonaws.com/dev`.

6. Teste Inicial do Fluxo de Ingestão (API -> Lambda -> SQS)

Vamos verificar se a primeira parte do nosso fluxo está funcionando corretamente.

6.1. Enviar uma Requisição de Teste usando curl

Você pode usar ferramentas como Postman, Insomnia ou o comando curl no terminal.

Neste laboratório, vamos usar o **curl**, uma ferramenta de linha de comando que permite enviar requisições HTTP diretamente do terminal. Vamos testar a integração com sua API criada no Amazon API Gateway.

1. Prepare a Requisição

Você enviará uma requisição POST para a URL da sua API com um payload em formato JSON. Utilize o comando que você encontrará, [clikando aqui](#), e altere os **valores abaixo**:

- **<INVOKE_URL>**: URL de Invocação (Invoke URL) que você anotou anteriormente.
- **pedidold** e **clienteld**: personalize com seus dados.

2. Exemplo com **URL fictício** de como deverá ficar:

```
curl -X POST https://abcdef123.execute-api.us-east-1.amazonaws.com/dev/pedidos \
-H "Content-Type: application/json" \
-d '{
  "pedidold": "labSemanaDev-josely",
  "clienteld": "clienteXYZ-josely",
  "itens": [
    {"produto": "Caneta Azul", "quantidade": 10},
    {"produto": "Caderno Universitário", "quantidade": 2}
  ]
}'
```

4. Execute no terminal

- **No Windows**, você pode usar o **Git Bash**, **PowerShell**, **WSL**, **terminal do VS Code** ou o **CloudShell**.

Para este laboratório, estou utilizando o **WSL**, mas você pode usar o **CloudShell** e terá a **mesma experiência**.

- O **Subsistema do Windows para Linux (WSL)** é um recurso do Windows que permite executar um ambiente Linux em seu computador Windows, sem a necessidade de uma máquina virtual separada ou inicialização dupla.

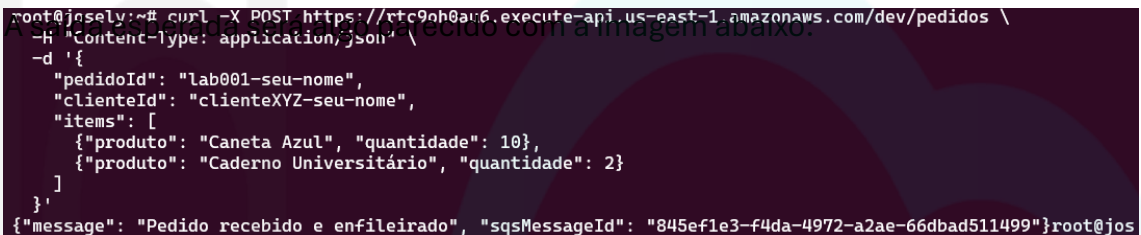
- No Linux/macOS: use o terminal nativo.

Após executar o comando, você deverá receber uma resposta da API indicando o sucesso ou falha do envio.

6.2. Verificar o Resultado

1. **Resposta da API:** Você deve receber uma resposta HTTP 200 OK com um corpo JSON semelhante a:

```
{
  "message": "Pedido recebido e enfileirado",
  "sqsMessageId": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxx"
}
```



A screenshot of a terminal window with a dark background. It shows a curl command being executed: `curl -X POST https://rtc9nh0ar6.execute-api.us-east-1.amazonaws.com/dev/pedidos \`. The command is followed by headers `-H 'Content-type: application/json' \` and a data payload `-d '{ "pedidoId": "lab001-seu-nome", "clienteId": "clienteXYZ-seu-nome", "items": [{ "produto": "Caneta Azul", "quantidade": 10}, { "produto": "Caderno Universitário", "quantidade": 2}] }'`. The output of the command is a JSON object: `{ "message": "Pedido recebido e enfileirado", "sqsMessageId": "845ef1e3-f4da-4972-a2ae-66dbad511499" }`. The prompt `root@josely:~#` is visible at the start and end of the command line.

2. **Logs da Lambda:**

- Vá para o serviço **CloudWatch** no Console AWS.
- No menu à esquerda, selecione **"Log groups"**.
- Encontre e clique no log group `/aws/lambda/pre-validacao-lambda-seu-nome`.
- Role para baixo até **"Log streams"**.
- Clique no log stream mais recente para ver os logs da execução. Você deve ver a mensagem `"Evento recebido do API Gateway"` e `"Mensagem enviada para SQS"`:

```
Evento recebido do API Gateway: {'resource': '/pedidos', 'path': '/pedidos', 'httpMethod': 'POST', 'headers': {'accept': '*/', 'content-type': 'application/json', 'Host': 'rtc9oh0au6.execute-api.us-east-1.amazonaws.com', 'User-Agent': 'curl/8.5.0', 'X-Amzn-Trace-Id': 'Root=1-681ba2ee-4a26689118730faa40210df1', 'X-Forwarded-For': '177.10.7.22', 'X-Forwarded-Port': '443', 'X-Forwarded-Proto': 'https'}, 'multiValueHeaders': {'accept': ['/*/*'], 'content-type': ['application/json'], 'Host': ['rtc9oh0au6.execute-api.us-east-1.amazonaws.com'], 'User-Agent': ['curl/8.5.0'], 'X-Amzn-Trace-Id': ['Root=1-681ba2ee-4a26689118730faa40210df1'], 'X-Forwarded-For': ['177.10.7.22'], 'X-Forwarded-Port': ['443'], 'X-Forwarded-Proto': ['https']}, 'queryStringParameters': None, 'multiValueQueryStringParameters': None, 'pathParameters': None, 'stageVariables': None, 'requestContext': {'resourceId': 'cxjx51', 'resourcePath': '/pedidos', 'httpMethod': 'POST', 'extendedRequestId': 'KNZ1YEGXoAMef8g=', 'requestTime': '07/May/2025:18:14:06 +0000', 'path': '/dev/pedidos', 'accountId': '237482015336', 'protocol': 'HTTP/1.1', 'stage': 'dev', 'domainPrefix': 'rtc9oh0au6', 'requestTimeEpoch': 1746641646840, 'requestId': '67029c51-a6ee-45ee-af1a-aeb8ed070b0e', 'identity': {'cognitoIdentityPoolId': None, 'accountId': None, 'cognitoIdentityId': None, 'caller': None, 'sourceIp': '177.10.7.22', 'principalOrgId': None, 'accessKey': None, 'cognitoAuthenticationType': None, 'cognitoAuthenticationProvider': None, 'userArn': None, 'userAgent': 'curl/8.5.0', 'user': None}, 'domainName': 'rtc9oh0au6.execute-api.us-east-1.amazonaws.com', 'deploymentId': 'wx2iwb', 'apiId': 'rtc9oh0au6'}, 'body': '{\n  "pedidoId": "lab001-seu-nome",\n  "clienteId": "clienteXYZ-seu-nome",\n  "items": [\n    {\n      "produto": "Caneta Azul",\n      "quantidade": 10\n    },\n    {\n      "produto": "Caderno Universit\u00e1rio",\n      "quantidade": 2\n    }\n  ]\n}', 'isBase64Encoded': False}
```

Mensagem enviada para SQS: 845ef1e3-f4da-4972-a2ae-66dbad511499

Back to top

3. Mensagem na Fila SQS:

- V\u00e1 para o servi\u00e7o **SQS**.
- Selecione sua fila pedidos-fifo-queue-seu-nome.fifo.
- Clique em **"Send and receive messages"**.
- Na se\u00e7\u00e3o "Receive messages", clique em **"Poll for messages"**.
- Voc\u00ea dever\u00e1 ver uma mensagem na fila. Clique no ID da mensagem para ver seus detalhes e corpo. **N\u00e3o delete a mensagem ainda**, pois ela ser\u00e1 consumida pela pr\u00f3xima Lambda.
- Pode fechar clicando no **"X"**.

7. Configura\u00e7\u00e3o do Barramento de Eventos (Amazon EventBridge)

O EventBridge permitir\u00e1 que diferentes partes do nosso sistema reajam a eventos de forma desacoplada. Criaremos um barramento customizado para os eventos de nossos pedidos.

7.1. Criar o Custom Event Bus (pedidos-event-bus-seu-nome)

1. Acesse o servi\u00e7o **Amazon EventBridge** no Console AWS.
2. No menu \u00e0 esquerda, selecione **"Event buses"**.
3. Clique em **"Create event bus"**.
4. **Name:** pedidos-event-bus-seu-nome.
5. Mantenha as outras op\u00e7\u00f5es padr\u00e3o (n\u00e3o precisamos de Archive ou Schema Registry para este lab).
6. Clique em **"Create"**.

7. **Anote o ARN** deste Event Bus. Ele será usado para configurar permissões e a Lambda.

7.2. Atualizar Permissões da Role de Validação de Pedidos

Agora que a fila SQS e o Event Bus estão criados, vamos conceder à `lambda-validacao-pedidos-role-seu-nome` as permissões necessárias.

1. Volte ao serviço **IAM**.
2. Clique em **"Roles"** e encontre a role `lambda-validacao-pedidos-role-seu-nome`. Clique nela.
3. Na aba **"Permissions"**, clique em **"Add permissions"** e selecione **"Create inline policy"**.
4. Selecione a aba **JSON**.
5. Apague o código existente.
1. Cole o JSON abaixo, substituindo as seguintes informações:
 - **COLE AQUI O ARN DA SUA FILA PRINCIPAL FIFO:** Pelo o ARN da sua fila FIFO Principal. **Ex:** `arn:aws:sqs:us-east-1:237482015336:pedidos-fifo-queue-seu-nome.fifo`
 - **COLE AQUI O ARN DO SEU EVENTBUS:** Pelo o ARN do Event bus recém-criado. **Ex:** `arn:aws:events:us-east-1:237482015336:event-bus/pedidos-event-bus-seu-nome`
 - Se dê algum erro ao colar o seu JSON, verifique se não tem espaços entre as aspas ao colar os seus ARNs do Event Bus e fila SQS.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "sqs:ReceiveMessage",
        "sqs:DeleteMessage",
        "sqs:GetQueueAttributes"
      ],
      "Resource": "COLE AQUI O ARN DA SUA FILA PRINCIPAL FIFO"
    },
    {
      "Effect": "Allow",
      "Action": "events:PutEvents",
      "Resource": "COLE AQUI O ARN DO SEU EVENT BUS"
    }
  ]
}
```

6. Clique em **"Next"**.
7. **Policy name:** SQSReadEventBridgePutPolicy-seu-nome.
8. Clique em **"Create policy"**.

8. Implementação da Lógica de Validação de Pedidos (AWS Lambda)

Esta segunda função Lambda será acionada por novas mensagens na fila SQS, fará uma validação mais detalhada do pedido e publicará um evento no EventBridge.

8.1. Criar a Função Lambda (validacao-pedidos-lambda-seu-nome)

1. Acesse o serviço **AWS Lambda**.
2. Clique em **"Create function"**.
3. Selecione **"Author from scratch"**.
4. **Function name:** validacao-pedidos-lambda-seu-nome.
5. **Runtime:** Selecione **Python 3.12**.

6. **Permissions:** Expanda "Change default execution role", selecione "**Use an existing role**", e escolha lambda-validacao-pedidos-role-seu-nome.
7. Clique em "**Create function**".

8.2. Configurar o Código, Variáveis de Ambiente e Trigger SQS

1. Na página da função, aba "**Code**", substitua o conteúdo de lambda_function.py pelo código em python que você encontrará [clcando aqui](#).
2. Clique em "**Deploy**".
3. Vá para a aba "**Configuration**" > "**Environment variables**".
4. Clique em "**Edit**", depois em "**Add environment variable**".
 - **Key:** EVENT_BUS_NAME
 - **Value:** pedidos-event-bus-seu-nome (o nome do seu Event Bus).
5. Clique em "**Save**".
6. Ainda na aba "**Configuration**", selecione "**Triggers**" no menu à esquerda.
7. Clique em "**Add trigger**".
8. **Trigger configuration:**
 - No menu suspenso, selecione "**SQS**" como a fonte.
 - **SQS queue:** Selecione sua fila pedidos-fifo-queue-seu-nome.fifo.
 - **Batch size:** Defina como 1 (isso é importante para processamento FIFO e para que a Lambda processe uma mensagem por invocação).
9. Clique em "**Add**".

9. Teste Completo do Fluxo do DIA 1

Agora, vamos testar todo o fluxo construído nesta aula. Se você ainda tem a mensagem do teste anterior na fila SQS (pedidos-fifo-queue-seu-nome.fifo), a Lambda validacao-pedidos-lambda-seu-nome deverá processá-la automaticamente em breve. Caso contrário:

9.1. Enviar uma Nova Requisição. Repita o passo 6.1


1. Use curl para enviar uma **nova** requisição POST para sua API Gateway, como no Passo 6.1, mas com um pedidoId diferente:

```
curl -X POST <INVOKE_URL>/pedidos \
-H "Content-Type: application/json" \
-d '{
  "pedidoId": "lab001-seu-nome",
  "clienteId": "clienteABC-seu-nome",
  "itens": [
    {"produto": "Borracha Branca", "quantidade": 5}
  ]
}'
```

- **<INVOKE_URL>**: URL de Invocação (Invoke URL). Ex: <https://rtc9oh0au6.execute-api.us-east-1.amazonaws.com/dev>
- **pedidoId** e **clienteId**: personalize com seus dados.

9.2. Verificar o Fluxo Completo

1. **Logs da pre-validacao-lambda-seu-nome**: Verifique os logs no CloudWatch para confirmar que a API acionou esta Lambda e que ela enviou a mensagem para a SQS:

Log groups (2)  [Actions](#) [View in Logs Insights](#) [Start tailing](#) [Create log group](#)

By default, we only load up to 10000 log groups.

☐ Exact match < 1 >

<input type="checkbox"/>	Log group	Log class	Anomaly d...
<input type="checkbox"/>	/aws/lambda/pre-validacao-lambda-seu-nome	Standard	Configure
<input type="checkbox"/>	/aws/lambda/validacao-pedidos-lambda-seu-nome	Standard	Configure

2. **Fila SQS**: Ela deve ser consumida rapidamente.
3. **Logs da validacao-pedidos-lambda-seu-nome**: Verifique os logs no CloudWatch para esta Lambda. Você deve ver:
 - "Evento SQS recebido".
 - "Processando pedido do SQS".
 - "Pedido X validado com sucesso".
 - "Evento publicado no EventBridge".

Message

No older events at this moment. [Retry](#)

INIT_START Runtime Version: python:3.12.v65 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:74b4a601bc20ef933c2b521f880902b0287c

START RequestId: ece1bffd-0303-5908-a6e8-df68c0431f35 Version: \$LATEST

Evento SQS recebido: {'Records': [{'messageId': 'dbc81358-06af-48f9-80b1-33c55227e15c', 'receiptHandle': 'AQEBZe5eXWRYdjzbiH+UADn1vvxjC8kyjU5qjCr185Lbb+kY8/VXW7jHdT2vruNDuItEwJjig7KZ9iRd7hEgAPL0rUZV6LtgMh1NG89IqNZr5QbtPZv4a9JI94RWhu7Xdg+ZinkWwFbzs0=', 'body': '{"pedidoId": "lab002-seu-nome", "clienteId": "clienteABC-seu-nome", "items": [{"produto": "Borracha Branca", "quantidade": 6b30eb422bb7a26363a46b35;Parent=678739a74dff0a42;Sampled=0;Lineage=1:4d852b33:0', 'SentTimestamp': '1746649362135', 'SequenceNumber': '18890d01-457a-a58b-868e10566302', 'ApproximateFirstReceiveTimestamp': '1746649362135', 'messageAttributes': {}, 'md5OfBody': '3695948de278686c

Processando pedido do SQS: {'pedidoId': 'lab002-seu-nome', 'clienteId': 'clienteABC-seu-nome', 'items': [{'produto': 'Borracha Branca', 'qu

Pedido lab002-seu-nome validado com sucesso.

Evento publicado no EventBridge: {'FailedEntryCount': 0, 'Entries': [{'EventId': 'a7b9bb1b-e0e2-99d1-c148-7097be41db4d'}], 'ResponseMetadata': {'content-type': 'application/x-amz-json-1.1', 'content-length': '85', 'date': 'Wed, 07 May 2025 20:22:43 GMT'}, 'RetryAttempts': 0}}

END RequestId: ece1bffd-0303-5908-a6e8-df68c0431f35

REPORT RequestId: ece1bffd-0303-5908-a6e8-df68c0431f35 Duration: 288.74 ms Billed Duration: 289 ms Memory Size: 128 MB Max Memory

No newer events at this moment. *Auto retry paused.* [Resume](#)

4. EventBridge Monitoring:

- Vá para **Amazon EventBridge > "Event buses"**.
- Selecione seu pedidos-event-bus-seu-nome.
- Clique na aba **"Monitoring"**.
- Você deverá ver um aumento na métrica PutEvents_Invocations ou MatchedEvents (pode levar alguns minutos para os gráficos atualizarem). Para uma verificação mais imediata de que o evento chegou, na próxima aula configuraremos uma regra que envia esses eventos para uma nova fila SQS.

Conclusão da Aula 1

Parabéns! Você construiu com sucesso a primeira etapa da nossa arquitetura, implementando um fluxo de ingestão de pedidos que utiliza API Gateway, Lambda, SQS FIFO e EventBridge. Você aprendeu a:

- Criar e configurar IAM Roles para permissões granulares.
- Configurar filas SQS FIFO com DLQs.
- Desenvolver funções Lambda em Python para processamento de dados.
- Expor Lambdas como endpoints REST usando API Gateway.
- Publicar eventos customizados em um EventBridge.

Na próxima aula, nos concentraremos na ingestão de arquivos via S3, outra forma comum de entrada de dados em sistemas.

IMPORTANTE:

Não exclua os recursos criados neste laboratório, pois daremos sequência na parte 2 da Semana do Desenvolvedor AWS na próxima aula.

Prepare-se para a Aula 2!