

# Passo a Passo AWS Lambda

## Etapa 1: Criar a função do Lambda

- No console do **AWS Lambda**, selecione **Criar uma função**. Observação: o console só mostra esta página se não houver funções do Lambda criadas. Se já tiverem sido criadas funções, a opção será exibida a página Lambda > Funções.

Funções (2)

Última busca há 2 minutos

Ações

Criar função

Filtrar por tags e atributos ou pesquisar por palavra-chave

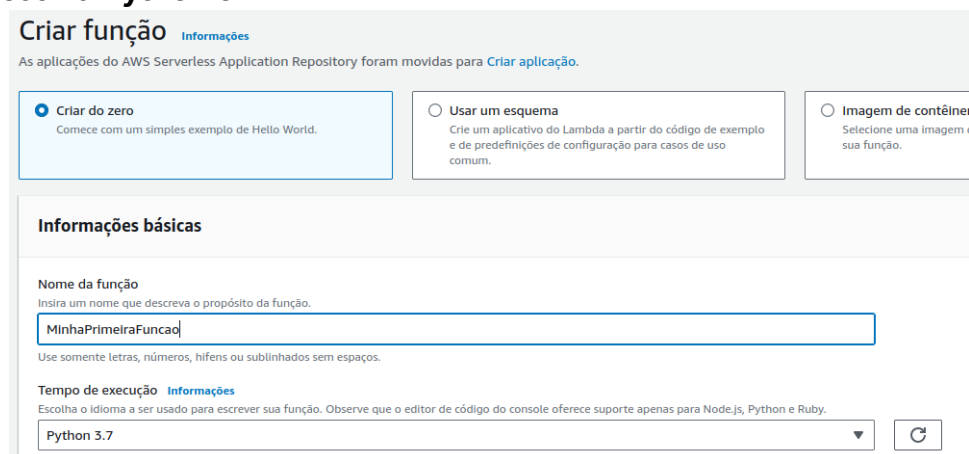
<

1

>

<div><input type="checkbox"/></div>	Nome da função	Descrição	Tipo de pacote	Tempo de execução	Última modificação
<div><input type="checkbox"/></div>	<a href="#">StackSet-account-password-policy--LambdaFunctionV2-bOqt66ab4itF</a>	-	Zip	Node.js 16.x	há 7 dias
<div><input type="checkbox"/></div>	<a href="#">aws-controltower-NotificationForwarder</a>	SNS message forwarding function for aggregating account notifications.	Zip	Python 3.9	há 7 dias

- Selecione **Author from scratch (criar do zero)**
- Em **Function name (nome da função)**, defina o nome da função. Em **Runtime**, escolha **Python 3.7**.



The screenshot shows the 'Criar função' (Create function) wizard in the AWS Lambda console. It has three tabs: 'Informações', 'Permissões', and 'Monitoramento'. The 'Informações' tab is active. Under 'Como criar a função' (How to create the function), 'Criar do zero' (Create from scratch) is selected. Below, in the 'Informações básicas' (Basic information) section, the 'Nome da função' (Function name) is 'MinhaPrimeiraFuncao' and the 'Tempo de execução' (Runtime) is 'Python 3.7'.

**Criar função** [Informações](#)

As aplicações do AWS Serverless Application Repository foram movidas para [Criar aplicação](#).

☒ **Criar do zero**  
Comece com um simples exemplo de Hello World.

☐ **Usar um esquema**  
Crie um aplicativo do Lambda a partir do código de exemplo e de predefinições de configuração para casos de uso comum.

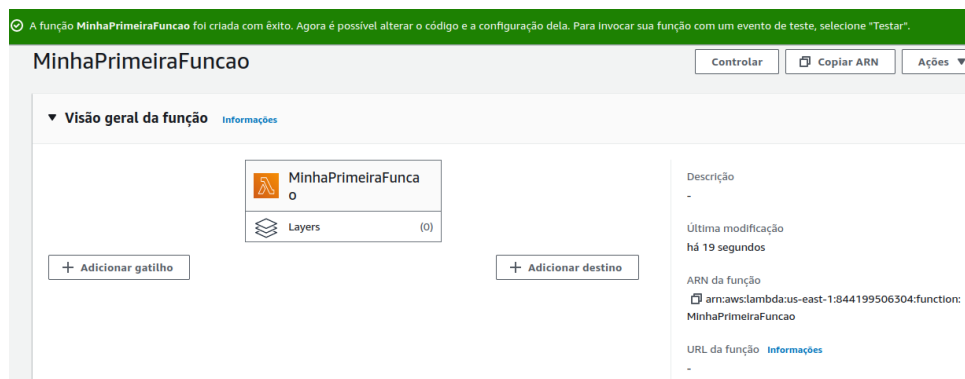
☐ **Imagem de contêiner**  
Selecione uma imagem de contêiner para sua função.

**Informações básicas**

**Nome da função**  
Insira um nome que descreva o propósito da função.  
  
Use somente letras, números, hífens ou sublinhados sem espaços.

**Tempo de execução** [Informações](#)  
Escolha o idioma a ser usado para escrever sua função. Observe que o editor de código do console oferece suporte apenas para Node.js, Python e Ruby.

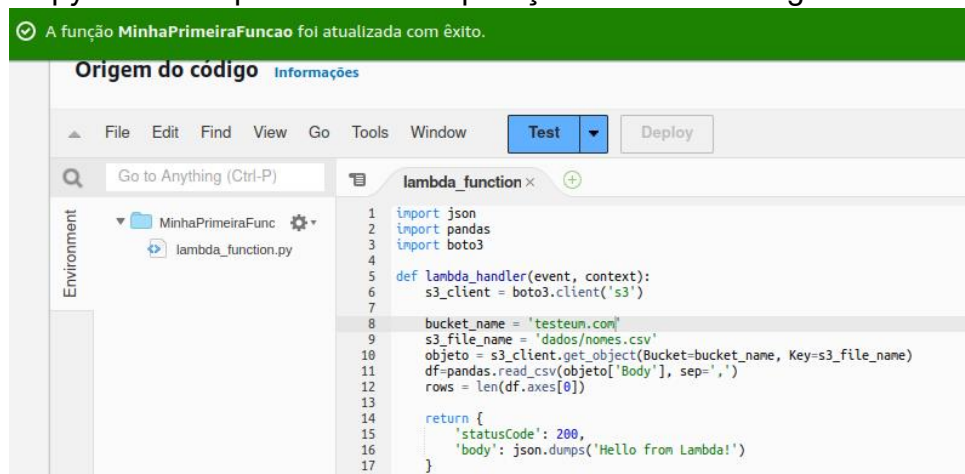
- Para criar a função, selecione **Create (Criar)**.



## Etapa 2: Construir o código:

A função será criada e você será redirecionado para o editor de funções do console. Por padrão, será criado o arquivo nomeado **lambda\_function.py**.

- Substitua **# TODO implement** pelo código que acessa o S3 e utiliza a biblioteca Numpy e Pandas para realizar a operação. Abaixo o código:



- Agora clique em **Deploy** para que a alteração do código seja realizada
- Realize o teste da Lambda clicando em **Test** e escolhendo um nome de teste
- Ao executar, o erro abaixo deve ser exibido:

Configurar evento de teste

Um evento de teste é um objeto JSON que simula a estrutura de solicitações emitidas pelos serviços da AWS para invocar uma função do Lambda. Use-o para visualizar o resultado da invocação da função.

Para invocar a função sem salvar um evento, configure o evento JSON e, em seguida, escolha Testar.

Ação de evento de teste

Criar novo evento

Editar evento salvo

Nome do evento

MeuEventoTeste

Máximo de 25 caracteres, formados por letras, números, pontos, hífens e sublinhados.

Configurações de compartilhamento de eventos

Privado

Compartilhável

Esse evento está disponível apenas no console do Lambda e para o criador do evento. É possível configurar um total de 10. [Saiba mais](#)

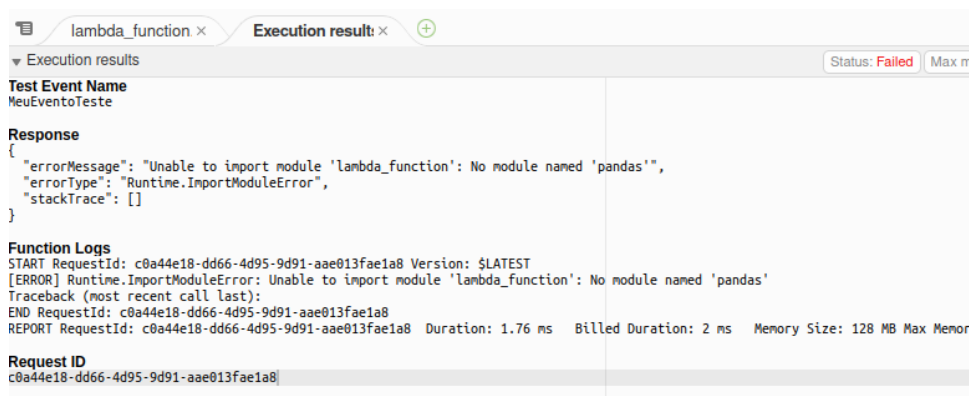
O evento está disponível para usuários do IAM na mesma conta que têm permissões para acessar e usar eventos compartilháveis. [Saiba mais](#)

Modelo - opcional

Cancelar

Invocar

Salvar



Este erro ocorre pois o serviço AWS Lambda não possui a biblioteca pandas. Precisamos de uma **layer** para importar estas bibliotecas necessárias a nossa Lambda.

### ***Etapas3: Criar uma Layer:***

O que são **Layers (camadas)**? De acordo com a documentação, as camadas do Lambda fornecem um modo conveniente de empacotar bibliotecas e outras dependências que você pode usar com suas funções Lambda. O uso de camadas reduz o tamanho dos arquivos de implantação carregados e acelera a implantação do código.

Uma camada é um arquivo compactado (zip) que pode conter código ou dados adicionais. Uma camada pode conter bibliotecas, um tempo de execução personalizado, dados ou arquivos de configuração. As camadas promovem o compartilhamento de código e a separação de responsabilidades para que você possa ater-se à escrita da lógica de negócios.

Quando você inclui uma camada em uma função lambda, o conteúdo é extraído para o diretório /opt no ambiente de execução

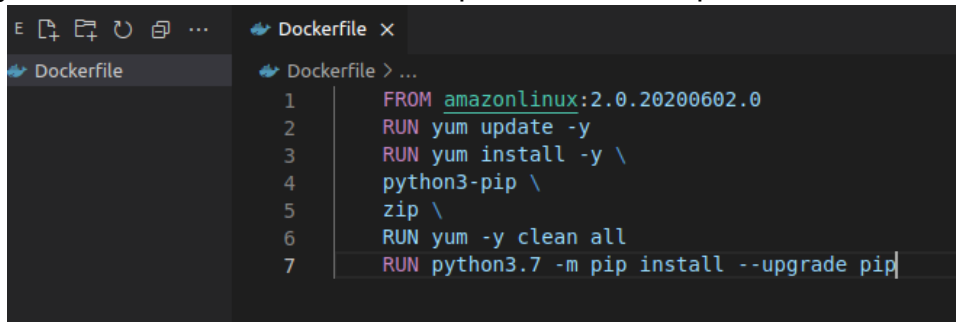
### **Como criar uma?**

É possível criar camadas usando o console da Lambda, a API do **AWS Lambda**, **CloudFormation**, ou **AWS Serverless Application Model (AWS SAM)**. Aqui vamos usar o método do console da Lambda com comandos do prompt e arquivos no formato zip.

Usando esse método, estaremos instalando diretamente as bibliotecas *python* e suas dependências necessárias em pasta de um Container Docker (sistema operacional Amazon Linux) e, em seguida, compactando-os para serem carregados na como camada à função Lambda.

Abaixo o passo a passo:

- Crie uma pasta nova e nela crie um arquivo chamado Dockerfile. Vamos usar uma imagem de sistema operacional Linux específica da Amazon e instalar o *python* versão 3.7 e a ferramenta para fazer a compressão dos dados.



```
Dockerfile
1 FROM amazonlinux:2.0.20200602.0
2 RUN yum update -y
3 RUN yum install -y \
4 python3-pip \
5 zip \
6 RUN yum -y clean all
7 RUN python3.7 -m pip install --upgrade pip
```

- Vamos usar o arquivo construído acima para criar a imagem do Docker:

```
docker build -t amazonlinuxpython37 .
```

- Agora, execute o comando abaixo na imagem do Docker para acessarmos o *shell* do container. O parâmetro *-it* é para sinalizar que queremos abrir imediatamente um *shell*:

```
docker run -it amazonlinuxpython37 bash
```

- Então você verá o prompt de comando dizer *bash-4.2#* ou algo parecido. Agora precisamos criar a pasta que receberá as bibliotecas necessárias para a layer que criaremos. **!!!Importante!!!**: as bibliotecas devem estar dentro de uma pasta chamada *python*.

```
bash-4.2# cd ~
```

```
bash-4.2# mkdir layer_dir
```

```
bash-4.2# cd layer_dir/
```

```
bash-4.2# mkdir python
```

```
bash-4.2# cd python/
```

```
bash-4.2# pwd
```

No final você estará com a estrutura de diretórios assim: */root/layer\_dir/python*

```

bash-4.2# cd ~
bash-4.2# mkdir layer_dir
bash-4.2# ls
layer_dir
bash-4.2# cd layer_dir/
bash-4.2# ls
bash-4.2# mkdir python
bash-4.2# ls
python
bash-4.2# pwd
/root/layer_dir
bash-4.2# cd python/
bash-4.2# pwd
/root/layer_dir/python
bash-4.2#

```

- Com a pasta criada, agora vamos baixar as bibliotecas e suas dependências para esta pasta python criada.

*bash-4.2# pip3 install pandas -T.*

```

/root/layer_dir/python
bash-4.2# ls
bash-4.2# pip3 install pandas -T .
Collecting pandas
  Downloading pandas-1.3.5-cp37-cp37m-manylinux_2_17_x86_64_manylinux2014_x86_64.whl (11.3 MB)
    11.3/11.3 MB 1.1 MB/s eta 0:00:00
Collecting python-dateutil>=2.7.3 (from pandas)
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
    247.7/247.7 kB 1.6 MB/s eta 0:00:00
Collecting pytz>=2017.3 (from pandas)
  Downloading pytz-2023.3.post1-py2.py3-none-any.whl.metadata (22 kB)
Collecting numpy>=1.17.3 (from pandas)
  Downloading numpy-1.21.6-cp37-cp37m-manylinux_2_12_x86_64_manylinux2010_x86_64.whl (15.7 MB)
    15.7/15.7 MB 504.4 kB/s eta 0:00:00
Collecting six>=1.5 (from python-dateutil>=2.7.3->pandas)
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Downloaded pytz-2023.3.post1-py2.py3-none-any.whl (502 kB)
Installing collected packages: pytz, six, numpy, python-dateutil, pandas
Successfully installed numpy-1.21.6 pandas-1.3.5 python-dateutil-2.8.2 pytz-2023.3.post1 six-1.16.0
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is
recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
bash-4.2# ls
__pycache__  numpy          pandas         pytz           six.py
bin           numpy-1.21.6.dist-info  pandas-1.3.5.dist-info  pytz-2023.3.post1.dist-info  six.py
dateutil     numpy.libs     python_dateutil-2.8.2.dist-info  six-1.16.0.dist-info

```

- Se você navegar para a pasta python, deverá ver as bibliotecas instaladas. Agora, de volta ao layer\_dir, vamos compactar o diretório python
- Compacte todos esses arquivos em um arquivo chamado minha-camada-pandas.zip. Certifique-se que você está no diretório /root/layer\_dir

*bash-4.2# cd ..*

*bash-4.2# zip -r minha-camada-pandas.zip .*

```

bash-4.2# ls
minha-camada-pandas.zip  python
bash-4.2#

```

- Copiar o zip do Container para a máquina local. Para tal, abra outra janela de terminal do seu SO e navegue até o diretório onde seu Dockerfile está. Inicialmente vamos descobrir o ID do Container Docker que está executando.

*docker container ls*

```

lins@lins-Lenovo-G460:~/Documentos/doc$ sudo docker container l
s
[sudo] senha para lins:
CONTAINER ID   IMAGE                                COMMAND      CREATED        STATUS        PORTS
NAMES
6dca560d0a13   amazonlinuxpython37                "bash"      20 minutes ago Up 20 minutes
musing_wilbur

```

Com o ID do container listado, vamos copiar o arquivo para máquina local. Substitua <id do container> pelo ID do container listado

```
docker cp <id do container>:/root/layer_dir/minha-camada-pandas.zip ./
```

```

lins@lins-Lenovo-G460:~/Documentos/doc$ sudo docker cp 6
dca560d0a13:/root/layer_dir/minha-camada-pandas.zip ./
Successfully copied 36.5MB to /home/lins/Documentos/doc./

```

- De acordo com a AWS, se a camada possuir mais do que 10 MB, o ideal é fazer via **S3**. Então faça upload do arquivo minha-camada-pandas.zip para um bucket S3.

**Objetos (5)**

Os objetos são as entidades fundamentais armazenadas no Amazon S3. Você pode usar o [inventário do Amazon S3](#) para obter uma lista de todos os objetos em seu bucket. Para outras pessoas acessarem seus objetos, você precisará conceder permissões explicitamente a eles. [Saiba mais](#)

Localizar objetos por prefixo

<input type="checkbox"/>	Nome	Tipo	Última modificação	Tamanho	Classe de armazenamento
<input type="checkbox"/>	404.html	html	25 Oct 2023 08:16:47 PM -03	0 B	Padrão
<input type="checkbox"/>	dados/	Pasta	-	-	-
<input type="checkbox"/>	index.html	html	25 Oct 2023 08:16:48 PM -03	261.0 B	Padrão
<input type="checkbox"/>	minha-camada-pandas.zip	zip	26 Oct 2023 10:31:40 PM -03	34.8 MB	Padrão
<input type="checkbox"/>	queries/	Pasta	-	-	-

- Agora temos a parte final onde carregamos o arquivo zip na Lambda para criar a camada. Retorne para o serviço AWS Lambda e no painel lateral, selecione **Camadas**.

**AWS Lambda**

Painel

Aplicativos

Funções

Recursos adicionais

Configurações de assinatura de código

**Camadas**

Réplicas

**Camadas (0)**

Última busca há 12 segundos

Filtrar camadas

Nome	Versão	Tempos de execução compatíveis	Arquiteturas compatíveis
Nenhum dado a ser exibido.			

- Clique no botão **Criar uma camada**
- Dê o nome de PandasLayer, escolha a opção **Fazer upload de um arquivo do Amazon S3**. Em outra aba retorne ao S3, localize o arquivo minha-camada-pandas.zip que você carregou para o S3 anteriormente e copie a **URL de objeto** que está no S3, por exemplo: <https://programa-bolsas-compass.s3.amazonaws.com/libs/minha-camada-pandas.zip>. Retornando para a aba de criação da camada, cole a URL em **Link do URL do Amazon S3**
- Escolha **x86\_64** em **Arquiteturas compatíveis**, em **Tempos de execução compatíveis** escolha Python 3.7

# Criar uma camada

## Configuração da camada

Nome

PandasLayer

Descrição - *opcional*

Descrição

☐ Fazer upload de um arquivo .zip

☒ Fazer upload de um arquivo do Amazon S3

Link do URL do Amazon S3

Cole um link do URL do S3 no arquivo .zip de código de funções.

https://s3.amazonaws.com/testeum.com/minha-camada-pandas.zip

Arquiteturas compatíveis - *opcional* [Informações](#)

Escolha as arquiteturas de conjunto de instruções compatíveis para a sua camada.

☒ x86\_64

☐ arm64

Tempos de execução compatíveis - *opcional* [Informações](#)

Selecione até 15 tempos de execução.

Tempos de execução

Python 3.7 X

Licença - *opcional* [Informações](#)

Cancelar Criar

- Clique em **Criar**

Lambda > Camadas > PandasLayer

ARN - arn:aws:lambda:us-east-1:844199506304:layer:PandasLayer:1

PandasLayer

Excluir Fazer download Criar versão

✔ Versão 1 da camada PandasLayer criada com êxito.

### Detalhes da versão

Versão	Descrição	Criado
1	-	há 12 segundos
Licença	Tempos de execução compatíveis	Arquiteturas compatíveis
-	python3.7	x86_64

## ***Etapas 4: Utilizando a Layer:***

- No menu, escolha **Função** e localize a função Lambda criada na Etapa 1
- Localize o ícone **Layers** e clique nele ou vá até o rodapé da Lambda até a seção nomeada de **Camadas**

Nenhum dado a ser exibido.

- Clique em **Adicionar uma camada**
- Escolha **Custom Layers (Camadas personalizadas)**, localize a camada e a versão criada na etapa anterior.

- Clique em Adicionar.

Ordem de mesclagem	Nome	Versão da camada	Tempos de execução compatíveis	Arquiteturas compatíveis	ARN da versão
1	PandasLayer	1	python3.7	x86_64	arn:aws:lambda:us-east-1:844199506304:layer:PandasLayer:1

- Agora execute novamente o código criado com o **Test** definido anteriormente. Deve ser retornado algo assim no Response:

```
{
  "statusCode": 200,
  "body": "Este arquivo tem 1825433 linhas"
}
```



Origem do código [Informações](#)

File Edit Find View Go Tools Window **Test** Deploy

Go to Anything (Ctrl-P)

MinhaPrimeiraFunc  
lambda\_function.py

```
1 import json
2 import pandas
3 import boto3
4
5 def lambda_handler(event, context):
6     s3_client = boto3.client('s3')
7
8     bucket_name = 'testeun.com'
9     s3_file_name = 'dados/nomes.csv'
10    objeto = s3_client.get_object(Bucket=bucket_name, Key=s3_file_name)
11    df=pandas.read_csv(objeto['Body'], sep=',')
12    rows = len(df.axes[0])
13
14    return {
15        'statusCode': 200,
16        'body': f"Este arquivo tem {rows} linhas"
17    }
18
```

Origem do código [Informações](#) Fazer upload de

File Edit Find View Go Tools Window **Test** Deploy

Go to Anything (Ctrl-P)

MinhaPrimeiraFunc  
lambda\_function.py

Execution results

Status: **Succeeded** Max memory used: 285 MB Time: 7300.76 ms

**Test Event Name**  
ResEventoTeste

**Response**

```
{
  "statusCode": 200,
  "body": "Este arquivo tem 1825433 linhas"
}
```

**Function Logs**

START RequestId: b0421d99-5ad3-4c84-a6d9-071fc081bc9b Version: \$LATEST  
END RequestId: b0421d99-5ad3-4c84-a6d9-071fc081bc9b  
REPORT RequestId: b0421d99-5ad3-4c84-a6d9-071fc081bc9b Duration: 7300.76 ms Billed Duration: 7301 ms Memory Size: 300 MB Max Memory Used: 285 MB Init Dur

**Request ID**  
b0421d99-5ad3-4c84-a6d9-071fc081bc9b

*Dica:* Provavelmente será necessário aumentar o tempo limite e o tamanho da memória da Lambda.

Configuração geral <a href="#">Informações</a>			<a href="#">Editar</a>
Descrição	Memória	Armazenamento temporário	
-	300 MB	512 MB	
Tempo limite	SnapStart <a href="#">Informações</a>		
0 min 20 seg	None		

**Referências:**

[https://docs.aws.amazon.com/pt\\_br/lambda/latest/dg/getting-started.html#getting-started-create-function](https://docs.aws.amazon.com/pt_br/lambda/latest/dg/getting-started.html#getting-started-create-function)

[https://docs.aws.amazon.com/pt\\_br/lambda/latest/dg/with-s3-example.html](https://docs.aws.amazon.com/pt_br/lambda/latest/dg/with-s3-example.html)

<https://levelup.gitconnected.com/creating-a-lambda-layer-via-docker-python-layers-docker-e4e318717822>