

Introdução ao Python para AWS com Boto3

Professor Ricardo Teixeira



Professor Ricardo Teixeira



- Pós-Graduado em Cloud Computing (Cloud Treinamentos, 2023)
- Mestre em Engenharia de Sistemas (UPE, 2015),
- Bacharel em Engenharia da Computação (UPE, 2012).

- Gestor de TI na Muda Meu Mundo (2021)
- Professor durante 5 anos na Faculdade SENAI (2017-2022)
- Fundador da Mídias Educativas e Mídias Criativas (2008)

- Aprendendo a programar desde 1999

Programa do curso

- Parte 1: Introdução à Programação em Python
 - Instalação e preparação do ambiente
 - Visão geral de um programa em Python
 - Variáveis e Tipos de dados
 - Operadores e estrutura condicional if-else e
 - Estruturas de repetição for e while
 - Funções
 - Módulos e Bibliotecas

Programa do curso

- Parte 2: Boto3 e Ferramentas de apoio
 - Boto3:
 - Introdução
 - Instalação
 - Session, Client e Resource
 - AWS CLI
 - Instalação
 - Boas práticas
 - Serverless Framework
 - Apresentação
 - Instalação
 - Primeiros passos

Programa do curso

- Parte 3: Python na AWS
 - Visão geral de AWS Lambda
 - Code, Trigger, Event, Layers e Function URL
 - Projeto: certificado de conclusão em PDF serverless.

O Projeto: gerador de certificado serverless

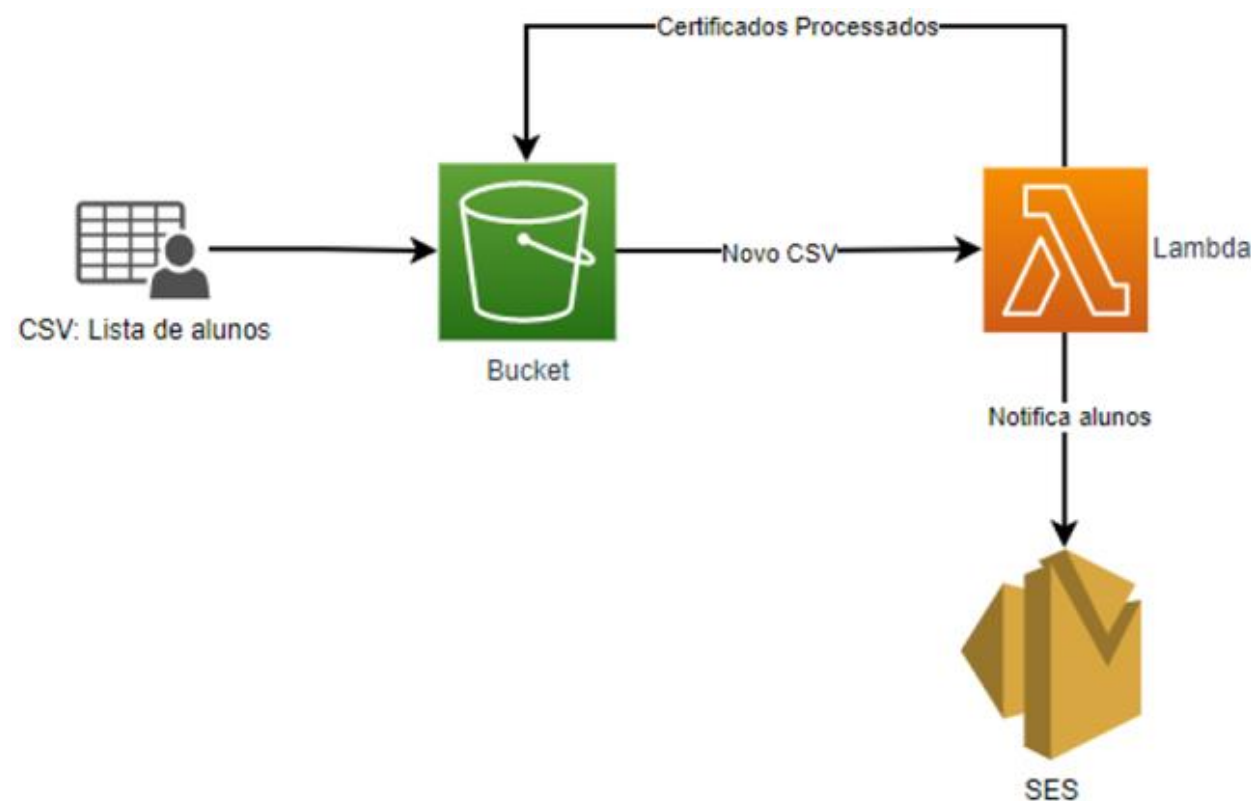
- **Objetivo**

criação de um serviço de processamento e geração de certificado de conclusão em PDF.

- **Descrição:**

Quando o arquivo CSV com uma lista de alunos chegar no Bucket a Lambda vai ler o arquivo e processar um certificado para cada aluno com base em um certificado modelo. Também enviará a URL para download do certificado para cada aluno via e-mail especificado no CSV.

Diagrama simplificado do projeto



Parte 1

Introdução à Programação em Python

Instalação e Preparação do Ambiente

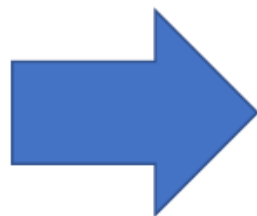
- Compatibilidade com AWS Lambda
 - https://docs.aws.amazon.com/pt_br/lambda/latest/dg/lambda-runtimes.html
- Instalação do Python
 - <https://www.python.org/downloads/>
 - Vamos usar a versão 3.10.xx:
 - <https://www.python.org/ftp/python/3.10.11/>
- Instalação do Visual Studio Code (VSCode)
 - https://code.visualstudio.com/?wt.mc_id=vscom_downloads

Instalação e Preparação do Ambiente

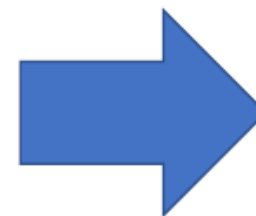
- Ambientes virtuais no Python
 - Possibilita compartimentalizar as bibliotecas na pasta do projeto.
 - Facilita na hora de implantar o projeto

C:\> Python310\

- Lib
 - ctypes
 - urllib
 - json
 - ...



Instalação do **boto3**



C:\> Python310

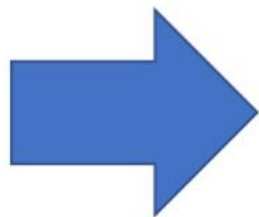
- Lib
 - ctypes
 - urllib
 - json
 - **boto3**
 - **botocore**
 - ...

Instalação e Preparação do Ambiente

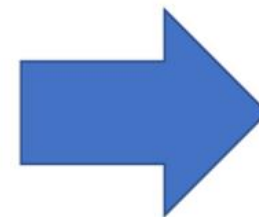
- Ambientes virtuais no Python
 - Possibilita compartimentalizar as bibliotecas na pasta do projeto.
 - Facilita na hora de implantar o projeto

C:\> meu_projeto\env\

- Lib
 - ctype
 - urllib
 - json
 - ...



Instalação do **boto3**



C:\> meu_projeto\env\

- Lib
 - ctype
 - urllib
 - json
 - **boto3**
 - **botocore**
 - ...

Instalação e Preparação do Ambiente

- Ambientes virtuais no Python

- Criando ambientes virtuais

```
C:\meu_projeto\> python -m venv NOME_AMBIENTE
```

- Ativando o ambiente

- Supondo um ambiente chamado **env** , para ativá-lo execute:

- no Windows:

```
C:\meu_projeto\> .\env\Scripts\activate
```

- no Linux e MacOS:

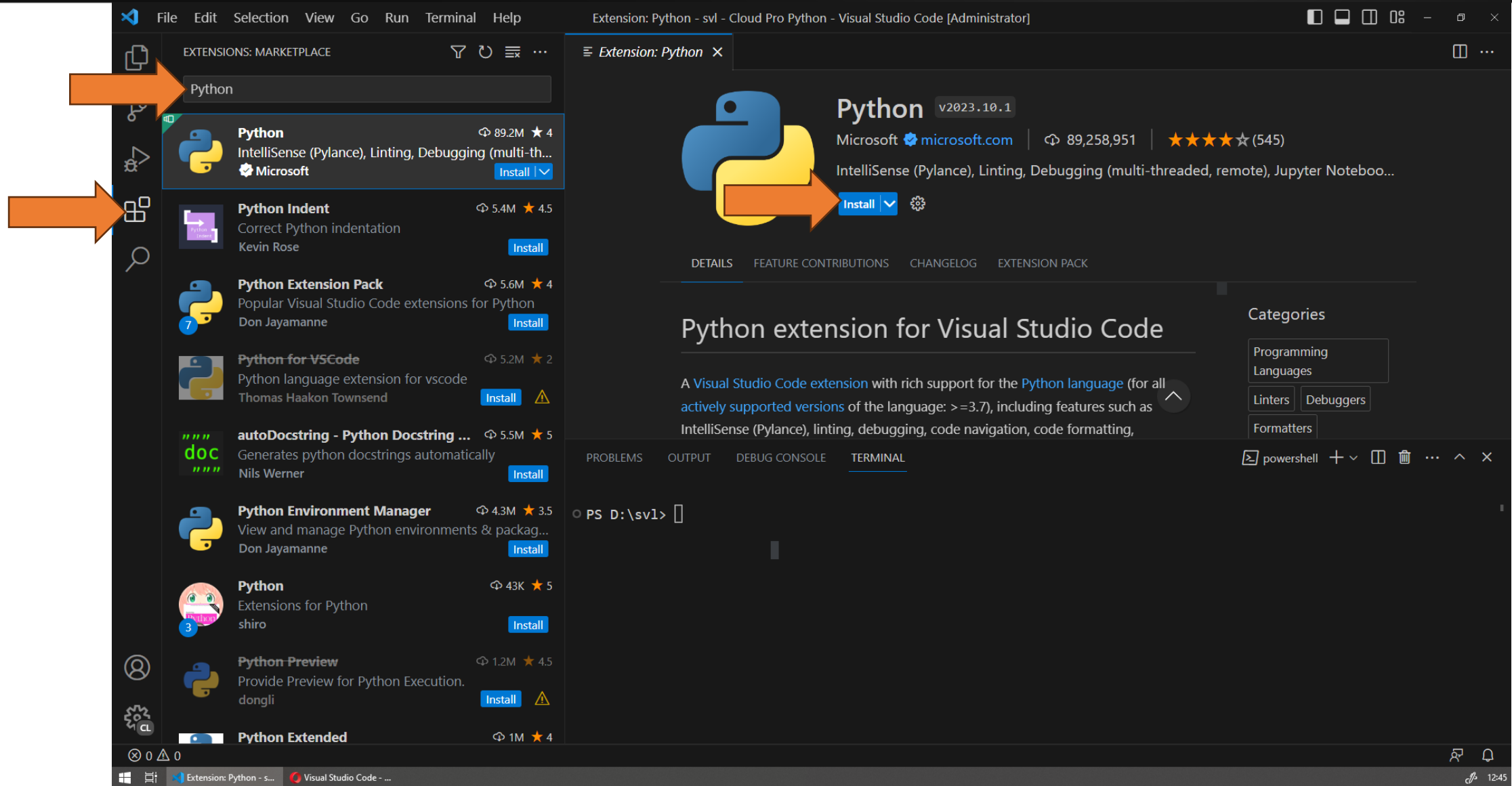
```
user@machine:~/meu_projeto$ source env/bin/activate
```

- Em seguida o nome do ambiente aparecerá no console.

```
(env) C:\meu_projeto\>
```

Visão geral de um programa em Python

- Consiste em uma sequência de declarações executadas linha a linha pelo interpretador Python
- Executando `python` em terminal abrirá o interpretador Python onde podemos escrever os comandos e executar a cada linha.
- Na prática os códigos são feitos em arquivos de texto com a extensão **.py**
- As IDEs (*Integrated Development Interface*, ou do inglês, Interface de Desenvolvimento Integrada) podem reconhecer e executar esses arquivos de forma mais simplificada.
- Instale a extensão **Python** para apoiar no desenvolvimento.
 - Veja como no slide seguinte



The screenshot displays the Visual Studio Code interface with the Python extension marketplace open. On the left, the 'EXTENSIONS: MARKETPLACE' sidebar shows a list of Python-related extensions. Two orange arrows point to the search bar (containing 'Python') and the 'Python' extension by Microsoft. The main panel shows the details for the 'Python' extension (v2023.10.1) by Microsoft, including its description, features, and an 'Install' button. The bottom status bar shows the terminal with the command 'PS D:\sv1>'.

EXTENSIONS: MARKETPLACE

- Python** (89.2M ★ 4) by Microsoft. IntelliSense (Pylance), Linting, Debugging (multi-threaded, remote), Jupyter Notebook support. [Install](#)
- Python Indent** (5.4M ★ 4.5) by Kevin Rose. Correct Python indentation. [Install](#)
- Python Extension Pack** (5.6M ★ 4) by Don Jayamanne. Popular Visual Studio Code extensions for Python. [Install](#)
- Python for VSCode** (5.2M ★ 2) by Thomas Haakon Townsend. Python language extension for vscode. [Install](#)
- autoDocstring - Python Docstring ...** (5.5M ★ 5) by Nils Werner. Generates python docstrings automatically. [Install](#)
- Python Environment Manager** (4.3M ★ 3.5) by Don Jayamanne. View and manage Python environments & packages. [Install](#)
- Python** (43K ★ 5) by shiro. Extensions for Python. [Install](#)
- Python Preview** (1.2M ★ 4.5) by dongli. Provide Preview for Python Execution. [Install](#)
- Python Extended** (1M ★ 4)

Extension: Python - svl - Cloud Pro Python - Visual Studio Code [Administrator]

Python v2023.10.1
Microsoft [microsoft.com](#) | 89,258,951 | ★★★★★ (545)
IntelliSense (Pylance), Linting, Debugging (multi-threaded, remote), Jupyter Notebook support

[Install](#)

Python extension for Visual Studio Code

A Visual Studio Code extension with rich support for the Python language (for all actively supported versions of the language: >=3.7), including features such as IntelliSense (Pylance), linting, debugging, code navigation, code formatting,

Categories

- Programming Languages
- Linters
- Debuggers
- Formatters

TERMINAL

```
PS D:\sv1>
```

Visão geral de um programa em Python

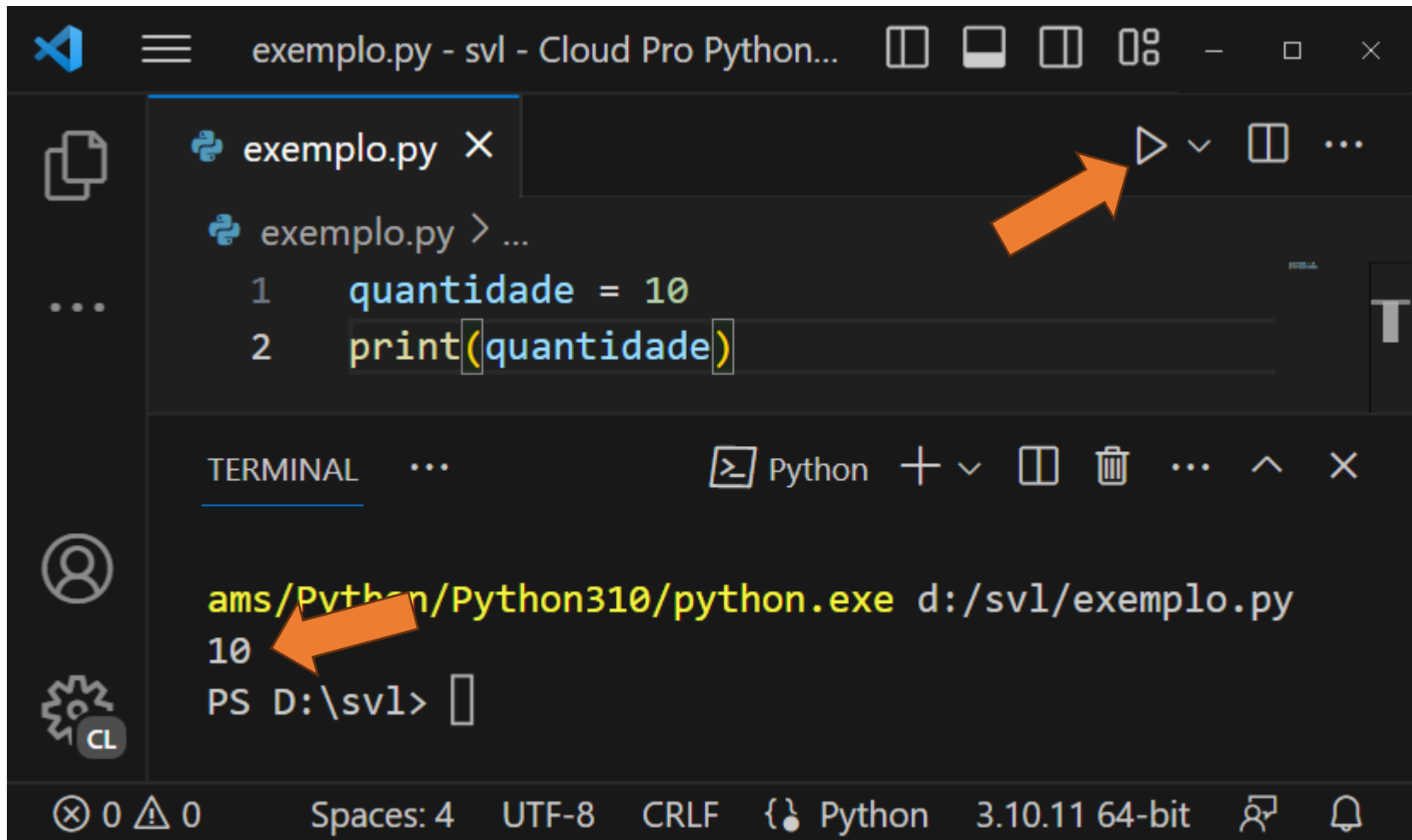
- Um programa Python pode ser tão simples como uma atribuição de valor numérico, como

```
quantidade = 10
```

- Não existe uma regra rígida para essa definição de um programa mínimo.
- Apesar disso precisamos seguir as regras da gramática e sintaxe da linguagem.
 - Gramática: palavras reservadas e operadores da linguagem
 - Sintaxe: regras de formatação do código e de como usar os elementos da gramática.

Visão geral de um programa em Python

- Com a extensão instalada, você pode executar um programa em Python clicando no ícone ▶
- O resultado, se houver, aparecerá no console abaixo.



The screenshot shows the Visual Studio Code interface with a Python file named `exemplo.py` open. The code in the file is:

```
1 quantidade = 10
2 print(quantidade)
```

The terminal window at the bottom shows the command `ams/Python/Python310/python.exe d:/svl/exemplo.py` being executed, resulting in the output `10`. The status bar at the bottom indicates the file is saved, UTF-8 encoding, CRLF line endings, and Python 3.10.11 64-bit is selected.

Vamos programar?

- Variáveis e Tipos de dados
- Operadores e estrutura condicional if-else e
- Estruturas de repetição for e while
- Funções
- Módulos e Bibliotecas
- Introdução ao Boto3:
 - Session, Client e Resource

Parte 2

Boto3 e Ferramentas de Apoio

Boto3

- Introdução

- O Boto3 é um SDK (*software development kit*, do inglês, kit de desenvolvimento de software).
- Consiste em um conjunto de módulos para criar, configurar e gerenciar serviços da AWS.

- Instalação

- <https://boto3.amazonaws.com/v1/documentation/api/latest/guide/quickstart.html#installation>
- Instalando com PIP (lembre-se de [ativar o ambiente](#))

```
(env) C:\> pip install boto3
```

Boto3

- Autorizando o Boto3
 - Assim como a AWS CLI, o Boto3 precisa de uma credencial para executar ações na conta da AWS.
- Sessão (Session)
 - A autorização é mantida pela sessão do Boto3.
 - Quando usamos o Boto3 no nosso código, uma sessão é criada automaticamente com as configurações de autorização definida.
- Clientes (Clients)
 - A partir da sessão podemos criar os Clients, ou clientes dos serviços.
 - Com eles temos acessos às funções de cada serviço (S3, EC2, Route53 etc.)
 - Podem ser instanciados a partir da sessão padrão ou de uma definida pelo programador.
- Recurso (Resource)
 - Representa literalmente um recurso específico.
 - Enquanto o client é do serviço, como S3, o resource é representa um bucket específico.

Boto3

- Se estiver com a credencial na configuração padrão (default) ou com nas variáveis de ambiente:
 - AWS_ACCESS_KEY_ID
 - AWS_SECRET_ACCESS_KEY

```
import boto3
```

```
s3_client = boto3.client('s3')  
lista = s3_client.list_buckets()  
print(lista)
```

Boto3

- **Não recomendado**, mas é possível passar como parâmetro:

```
import boto3

s3_client = boto3.client(
    's3',
    aws_access_key_id='AKIAUCXNAOTENTEUSAR',
    aws_secret_access_key='t3gnqEtp04qu3n4o51ida4c3rt0+mdVxxzvx1Kt7GB8A'
)

lista = s3_client.list_buckets()
print(lista)
```

Boto3

- Para usar o profile é preciso especificar o nome dela na sessão.

```
import boto3

sessao = boto3.Session(
    profile_name='python-cloud-prod',
    region_name='us-east-1'
)
s3_client = sessao.client('s3')
lista = s3_client.list_buckets()
print(lista)
```

AWS Lambda

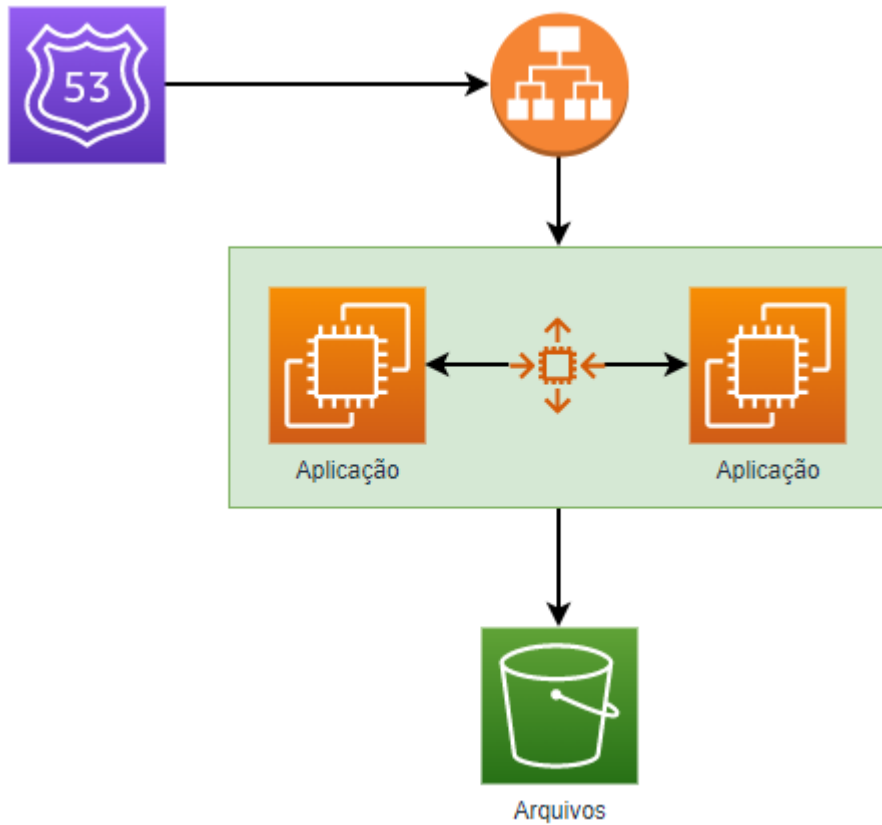
- O AWS Lambda é um serviço computacional sem servidor.
- A AWS gerencia os recursos necessários para executar o código.
- Você paga apenas pelo que usa: número de requisições, quantidade de memória, tempo de execução.
- Vantagens:
 - Não gerenciar o servidor!
 - Não precisa preparar o sistema
 - Não paga por tempo ocioso

Desvantagens

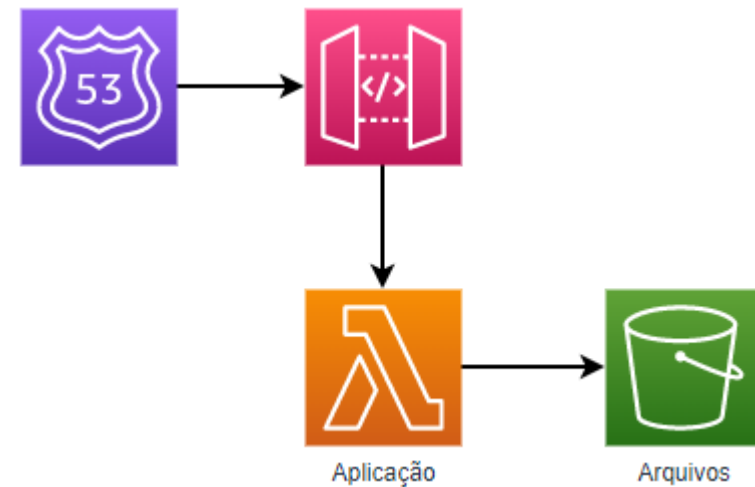
- Não tem acesso ao servidor
- Não é apropriado para realizar uma grande carga de computação

AWS Lambda

Aplicação com servidores EC2

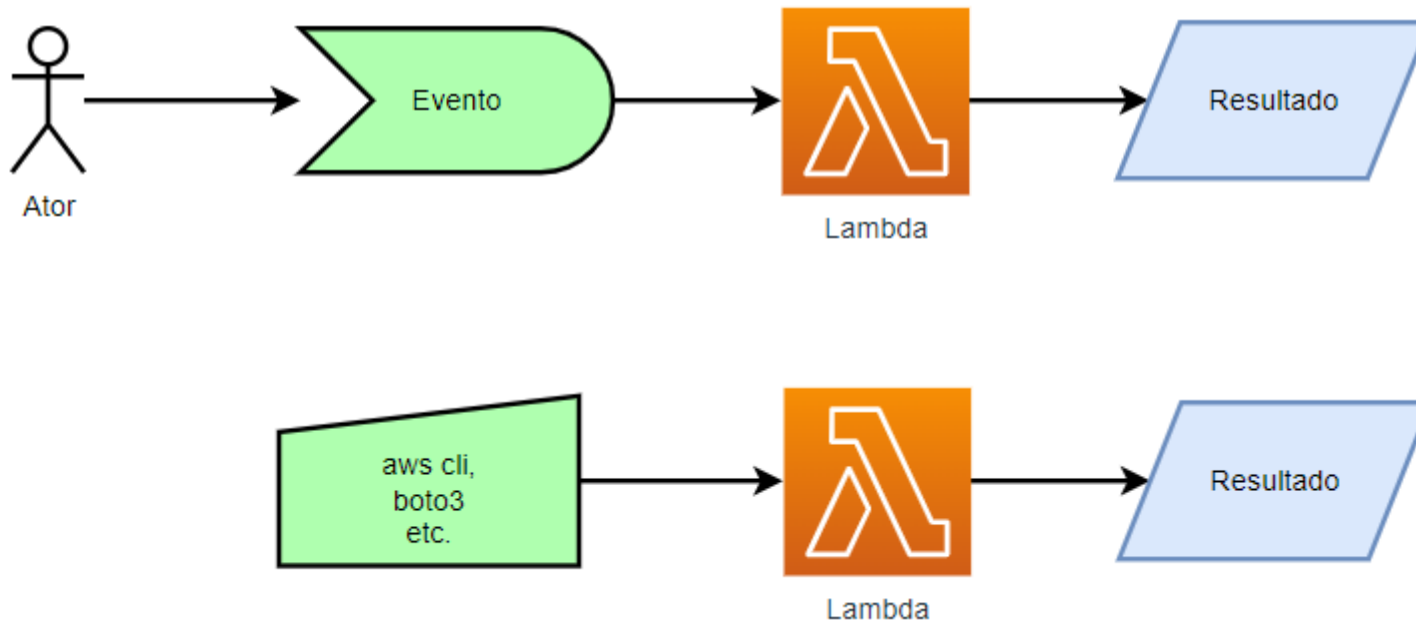


Aplicação com Lambda



Como o AWS Lambda funciona?

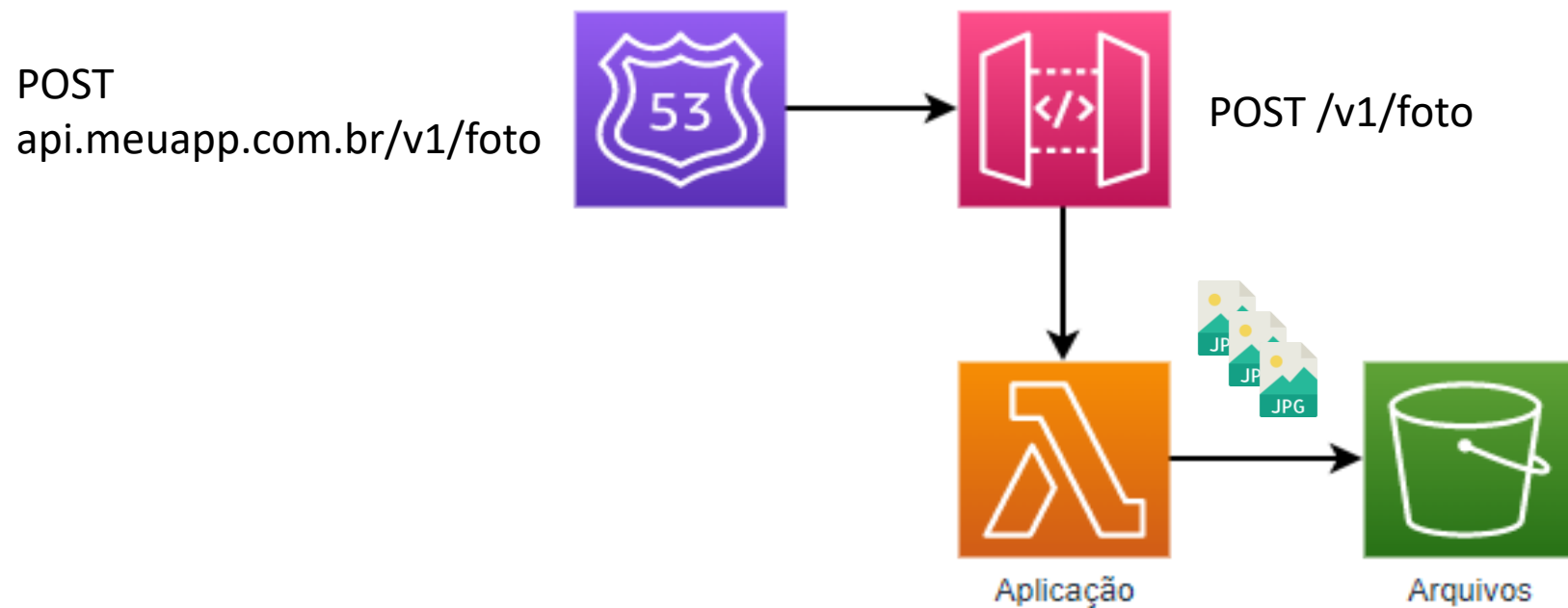
- É um serviço orientado a eventos que pode ser acionado de inúmeras formas



Como o AWS Lambda funciona?

- É um serviço orientado a eventos que pode ser acionado de inúmeras formas

Aplicação com Lambda



O diagrama ilustra a arquitetura de um sistema de upload e processamento de vídeos em AWS, dividido em várias etapas e componentes:

- Interface Web:** O usuário interage com um dispositivo (celular ou tablet) que envia o vídeo para o App Web Cloudfront. O App Web Cloudfront também recebe o vídeo diretamente do dispositivo. O App Web Cloudfront utiliza o Domínio Route53 para roteamento e o Autenticação de usuário Cognito para autenticação.
- Envio do vídeo:** O vídeo é enviado do App Web Cloudfront para o Serviço TUS EC2. O Serviço TUS EC2 utiliza o Arquivos Originais S3 para armazenar o vídeo original.
- Automatiza EC2:** O vídeo original é enviado para a Fila de Processamento SQS, que é processada pela Fila de Finalização SQS. A Função que vai subir a EC2 para processar o vídeo é responsável por isso.
- Processamento do vídeo:** O vídeo é enviado para o Sobe Encoding Lambda, que utiliza o EC2 Encoding para processar o vídeo. O vídeo processado é armazenado no Arquivos Processados S3.
- Fim do Processamento:** O vídeo processado é enviado para o Streaming do Vídeo Cloudfront, que é acessado pelo Ao Player de Vídeo. O vídeo também é enviado para o Fim Encoding Lambda, que realiza algumas ações de pós-processamento. O Fim Encoding Lambda utiliza o Sinaliza Fim SNS para enviar uma notificação de finalização.
- Acessa Registro de Vídeos:** O vídeo é enviado para o Script CRUD Lambda, que utiliza o Dados Vídeos DynamoDB para armazenar o vídeo. O Script CRUD Lambda também utiliza o API CRUD API Gateway para acessar o registro de vídeos.
- VPC Endpoint:** O VPC Endpoint libera acesso a um serviço AWS a partir de subnet privada. O VPC Interface Endpoint (SQS) e o VPC Gateway Endpoint (S3) são utilizados para isso.

AWS CLI

- Instalação

- https://docs.aws.amazon.com/pt_br/cli/latest/userguide/getting-started-install.html
- Criação de profile com credenciais do IAM

```
C:\> aws configure --profile nome_profile
```

- Boas práticas

- Nunca escreva credenciais no código
- Exclua credenciais quando não precisar mais delas
- Desative a credencial se não estiver em uso
- Rotacione as credenciais
- Nunca versione ou nem salve em repositório público ou não criptografado.

Serverless Framework

- Apresentação
 - É uma ferramenta para criação de IaC (*Infrastructure as a Code*, do inglês, Infraestrutura como um código).
 - Cria uma camada sobre o Cloudformation quando usa a AWS como provider
 - Permite o provisionamento e execução de infraestrutura a partir de um modelo especificado em um código com sintaxe YAML.
 - Comumente utilizado para desenvolver aplicações serverless.
- Instalação
 - Instalar NodeJS: <https://nodejs.org/en/download>
 - Instalar Serverless Framework CLI: <https://www.serverless.com/framework/docs/getting-started>

Serverless Framework

- Principais comandos

- Inicialização: inicializa a partir de um modelo:

```
C:\meu_projeto> serverless
```

- Publicação: cria na AWS toda a infra necessária e sobre o código junto com as dependências.

```
C:\meu_projeto> serverless deploy
```

- Executar Lambda:

```
C:\meu_projeto> serverless invoke -f nome_lambda
```

- Remover infra: apaga toda a infra criada pelo projeto.

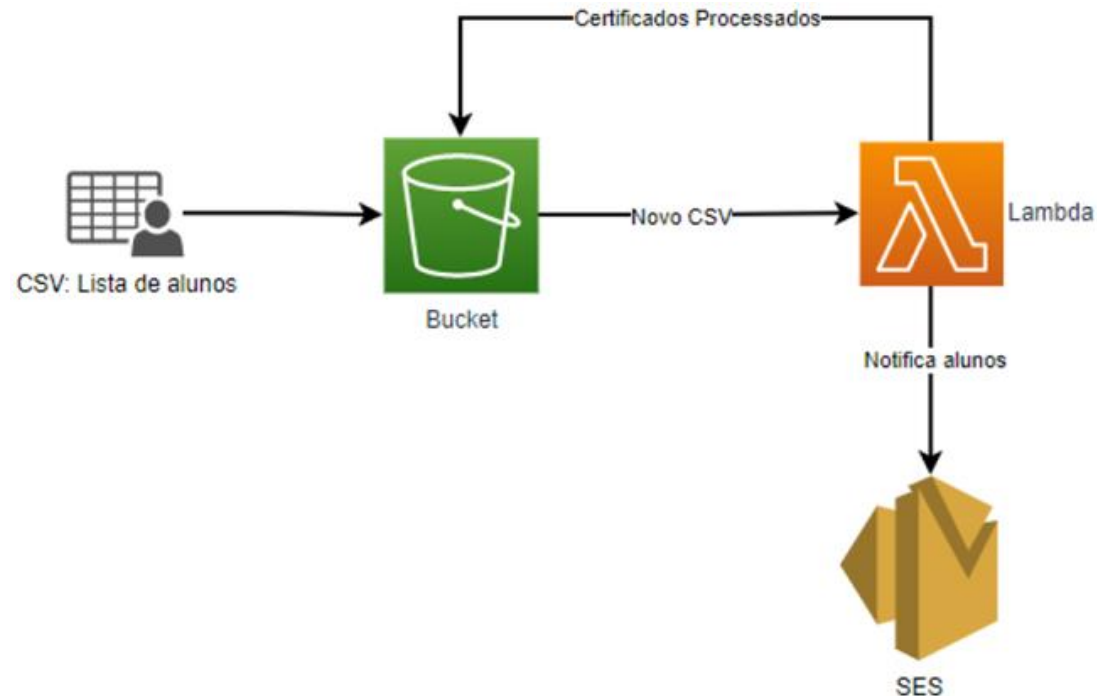
```
C:\meu_projeto> serverless remove
```

Parte 3

Python na AWS

Prática

- AWS Lambda com Serverless Framework
- Implementação do projeto do gerador de certificado serverless



Introdução ao Python para AWS com Boto3

Professor Ricardo Teixeira

Ricardo Teixeira

linkedin.com/in/ricardoteix



Referências

- Repositório do projeto no Github
<https://github.com/ricardoteix/python-cloud-prod>
- Como rotacionar chaves de acesso para usuários do IAM
<https://aws.amazon.com/pt/blogs/aws-brasil/como-rotacionar-chaves-de-acesso-para-usuarios-do-iam/>
- Documentação do Boto3
<https://boto3.amazonaws.com/v1/documentation/api/latest/index.html>