

**LARYSSA STEPHANIE ANDRADE DA COSTA SILVA**  
MAT: 211454050 | POLO: LONDRINA/PR

## **AVA1 – COMPUTAÇÃO EM NUVEM**

## 1. Contextualização

Você foi contratado para desenvolver uma API HTTP com as funções básicas de uma calculadora (soma entre dois números, subtração entre dois números, multiplicação entre dois números e divisão entre dois números). O contratante não restringiu a linguagem de programação que será utilizada, então você poderá desenvolver a API na linguagem de sua preferência. A única restrição imposta pelo contratante é que a API esteja sendo executada em uma instância de máquina virtual da AWS. Dessa forma, você deverá apresentar o passo a passo de cada etapa do projeto seguindo o roteiro a seguir:

1. Criar a API calculadora;
2. Criar uma máquina virtual na AWS;
3. Realizar as configurações necessárias para executar a API;
4. Implantar e executar as funções da API;
5. Excluir a máquina virtual.

Faça um relatório apresentando o passo a passo das etapas listadas no roteiro acima. Nele deverão ser apresentados os prints das telas em cada etapa do processo.

### 1.1.Criação da API Calculadora

A API foi desenvolvida na linguagem Python utilizando Django como framework. As classes implementadas foram Adição, Subtração, Multiplicação e Divisão de 2 números inteiros em cada uma delas (Figura 1). Na divisão por zero, foi feito uma condição para retornar um erro http 400 informando que não será possível realizar esse tipo de divisão. O repositório desse projeto foi versionado no github e pode ser acessado através do link: [https://github.com/laryssastephanie/data\\_science\\_studies/tree/main/Aulas\\_Unifil/comp\\_em\\_nuvem/calculadora\\_api](https://github.com/laryssastephanie/data_science_studies/tree/main/Aulas_Unifil/comp_em_nuvem/calculadora_api).

```
comp_em_nuvem > calculadora_api > calculadora > views.py > ...
1 from rest_framework.views import APIView
2 from django.http import HttpRequest, HttpResponse
3 from rest_framework.response import Response
4 from rest_framework import status
5
6 class Adicao(APIView):
7     def get(self, request: HttpRequest, a: int, b: int) -> HttpResponse:
8         r = a + b
9         return Response(data={"resultado": str(r)}, status=status.HTTP_200_OK)
10
11 class Subtracao(APIView):
12     def get(self, request: HttpRequest, a: int, b: int) -> HttpResponse:
13         r = a - b
14         return Response(data={"resultado": str(r)}, status=status.HTTP_200_OK)
15
16 class Multiplicacao(APIView):
17     def get(self, request: HttpRequest, a: int, b: int) -> HttpResponse:
18         r = a * b
19         return Response(data={"resultado": str(r)}, status=status.HTTP_200_OK)
20
21 class Divisao(APIView):
22     def get(self, request: HttpRequest, a: int, b: int) -> HttpResponse:
23         if b != 0:
24             r = a / b
25             return Response(data={"resultado": str(r)}, status=status.HTTP_200_OK)
26         else:
27             return Response(data={"Error": "Não é possível realizar divisão por zero"}, status=status.HTTP_400_BAD_REQUEST)
28
```

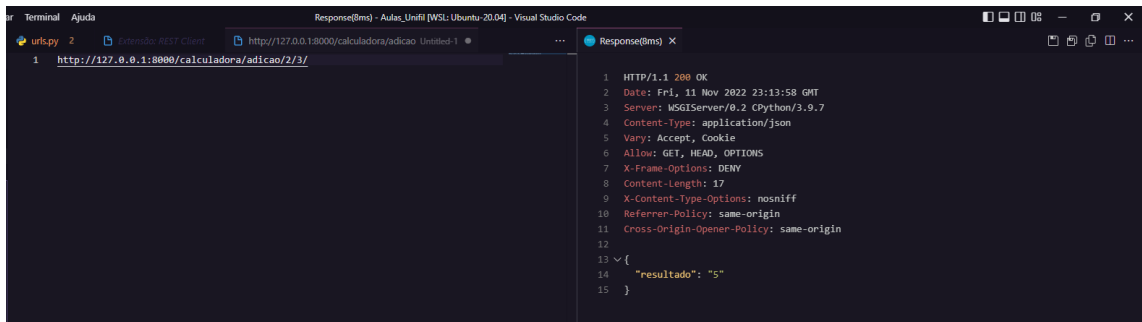
Figura 1 - Classes implementadas na calculadora

Os endpoints foram adicionados no arquivo urls.py para referenciar os paths disponíveis no sistema, conforme figura 2.

```
comp_em_nuvem > calculadora_api > calculadora > urls.py > ...
1 from rest_framework.urls import path
2 from calculadora.views import Adicao, Subtracao, Multiplicacao, Divisao
3
4 urlpatterns = []
5     path('adicao/<int:a>/<int:b>/', Adicao.as_view()),
6     path('subtracao/<int:a>/<int:b>/', Subtracao.as_view()),
7     path('multiplicacao/<int:a>/<int:b>/', Multiplicacao.as_view()),
8     path('divisao/<int:a>/<int:b>/', Divisao.as_view()),
9
```

Figura 2 - Endpoints da calculadora

Para testes da ferramenta antes de ser implementado no AWS, foi utilizado o rest client, uma extensão do VS Code, para fazer as chamadas na API localmente (Figura 3).



```
1 http://127.0.0.1:8000/calculadora/adicao/2/3/

1 HTTP/1.1 200 OK
2 Date: Fri, 11 Nov 2022 23:13:58 GMT
3 Server: WSGIServer/0.2 Python/3.9.7
4 Content-Type: application/json
5 Vary: Accept, Cookie
6 Allow: GET, HEAD, OPTIONS
7 X-Frame-Options: DENY
8 Content-Length: 17
9 X-Content-Type-Options: nosniff
10 Referrer-Policy: same-origin
11 Cross-Origin-Opener-Policy: same-origin
12
13 {
14   "resultado": "5"
15 }
```

Figura 3 - RestClient chamando a função de Adição

## 1.2.Criando uma máquina Virtual na AWS

Após fazer o cadastro na página da AWS, para fazer a criação de uma VM é necessário buscar por EC2 e iniciar a criação clicando no botão “Executar Instância”. Damos um nome para a instância e iniciamos a configuração de acordo com o seu propósito.

Algumas configurações são necessárias de serem feitas durante a criação da instância, ou após caso seja preciso. As figuras 4 e 5 mostram como foram feitas as configurações da instância, bem como as regras de entrada e saída para que a calculadora seja executada posteriormente. Aqui foi utilizado a porta 8000, pois é o padrão do Django.

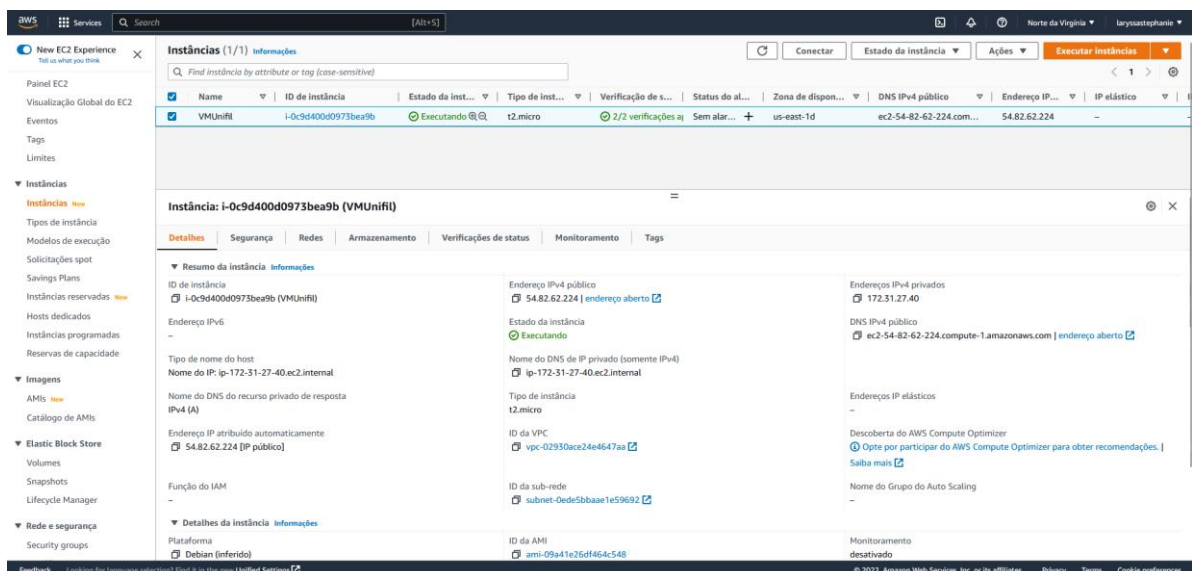


Figura 4 - Configurações da Instância criada

#### ▼ Regras de entrada

Regras de filtro				
ID da regra do grupo de s...	Intervalo de po...	Protocolo	Origem	Grupos de segurança
sgr-0a925a2d7bceb783b	8000	TCP	0.0.0.0/0	launch-wizard-1
sgr-0c5a66258f0f6873c	8000	TCP	::/0	launch-wizard-1
sgr-0f7844131a005718f	22	TCP	0.0.0.0/0	launch-wizard-1
sgr-02b4749d7842b5f90	80	TCP	0.0.0.0/0	launch-wizard-1
sgr-0c805607a97701767	443	TCP	0.0.0.0/0	launch-wizard-1

#### ▼ Regras de saída

Regras de filtro				
ID da regra do grupo de s...	Intervalo de po...	Protocolo	Destino	Grupos de segurança
sgr-06b75fe6fe42076be	Todos	Todos	0.0.0.0/0	launch-wizard-1

Figura 5 - Regras de entrada e saída configuradas

Após a configuração da instância, podemos conectá-la clicando em “Ações” e em “Conectar”, ainda na página da AWS. Os próximos passos será mostrado na tela, e assim podemos conectar dando o comando “ssh -i ‘exemplo\_unifil.pem’ admin@ec2-54-82-62-224.compute-1.amazonaws.com (Figura 6).

```
o ~/workspace/data_science_studies/Aulas_Unifil/comp_em_nuvem (main*) » ssh -i "exemplo_unifil.pem" admin@ec2-54-82-62-224.compute-1.amazonaws.com
Linux ip-172-31-27-40 5.10.0-14-cloud-amd64 #1 SMP Debian 5.10.113-1 (2022-04-29) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Nov 10 18:23:20 2022 from 177.82.12.53
admin@ip-172-31-27-40:~$
```

Figura 6 - Instância conectada

### 1.3.Realizar as configurações necessárias para executar a API

Com a VM devidamente criada e configurada, o próximo passo é prepara-la para executar a API. Os primeiros comandos podem ser para atualizar o sistema e deixar tudo na última versão. Posteriormente devemos instalar o Python, o PIP e o Django, que serão essenciais nesse projeto (Figura 7).

```
admin@ip-172-31-27-40:~$ py
py3clean      py3rsa-decrypt  py3rsa-keygen  py3rsa-sign    py3versions   pydoc3.9       pygettext3.9   python3         python3.9
py3compile    py3rsa-encrypt py3rsa-priv2pub py3rsa-verify  pydoc3        pygettext3     pyjwt3         python3-config  python3.9-config
admin@ip-172-31-27-40:~$ pi
pic           pido           pinentry       ping            ping6          pip             pip3
pic2graph     piconv        pidwait        pinentry-curses ping4           pinky           pip3
admin@ip-172-31-27-40:~$ pi
```

Figura 7 - Verificando instalações necessárias

Com a VM devidamente configurada para rodar o projeto, podemos copiar os arquivos do mesmo da nossa máquina para a VM através do comando “scp -i”, conforme mostra a figura 8. Para evitar copiar até mesmo os arquivos das bibliotecas

instaladas para rodar a aplicação, podemos apenas enviar a pasta contendo a estrutura básica da calculadora, e depois, dentro da VM, podemos criar um environment através do comando “python3.9 -m venv env”, ativá-lo através do comando “source/bin/activate”, instalar o Django com o environment ativado e iniciar o projeto através do comando “django-admin startproject project .”. Assim, a cópia dos arquivos para a VM será mais rápida, e podemos editar os arquivos necessários via terminal.

```
~/workspace/data_science_studies/Aulas_Unifil/comp_em_nuvem (main*) » scp -i exemplo_unifil.pem -r calculadora_api/calculadora admin@ec2-54-82-62-224.compute-1.amazonaws.com:/home/admin
admin.py 100% 63 0.4KB/s 00:00
admin.cpython-39.pyc 100% 238 1.4KB/s 00:00
views.cpython-39.pyc 100% 1790 10.4KB/s 00:00
models.cpython-39.pyc 100% 235 1.3KB/s 00:00
__init__.cpython-39.pyc 100% 197 1.1KB/s 00:00
apps.cpython-39.pyc 100% 484 2.8KB/s 00:00
urls.cpython-39.pyc 100% 560 3.2KB/s 00:00
models.py 100% 57 0.3KB/s 00:00
tests.py 100% 60 0.4KB/s 00:00
urls.py 100% 369 2.2KB/s 00:00
views.py 100% 1133 6.6KB/s 00:00
__init__.cpython-39.pyc 100% 208 1.2KB/s 00:00
__init__.py 100% 0 0.0KB/s 00:00
__init__.py 100% 0 0.0KB/s 00:00
apps.py 100% 154 0.9KB/s 00:00
```

Figura 8 - Copiando os arquivos da máquina física para a máquina virtual

#### 1.4.Implantar e executar as funções da API

Após alterações feitas do diretório da calculadora dentro da VM, devemos rodar alguns comandos para poder migrá-las. Os comandos são “python3.9 manage.py makemigrations” e “python3.9 manage.py migrate”. Feito isso, podemos iniciar a execução da aplicação através do comando “python3.9 manage.py runserver”. Uma mensagem aparecerá na tela informando que um servidor está em execução em um determinado endereço (Figura 9).

```
(env) admin@ip-172-31-27-40:~/calculadora_api$ python3.9 manage.py runserver 0.0.0.0:8000
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
November 12, 2022 - 03:28:01
Django version 4.1.3, using settings 'project.settings'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.
```

Figura 9 - Rodando o projeto na VM

O próximo passo será executar as funções da calculadora via AWS em um navegador, digitando o endereço de DNS IPv4 público disponibilizado na página anteriormente seguido da porta de execução (no caso será a porta 8000), o nome do projeto, o nome da função a ser executada e os dois números inteiros para realizar a

equação. Portanto, se quisermos fazer uma adição dos números 2 e 4, o endereço será: ec2-54-82-62-224.compute-1.amazonaws.com:8000/calculadora/adicao/2/4/ (Figura 10).

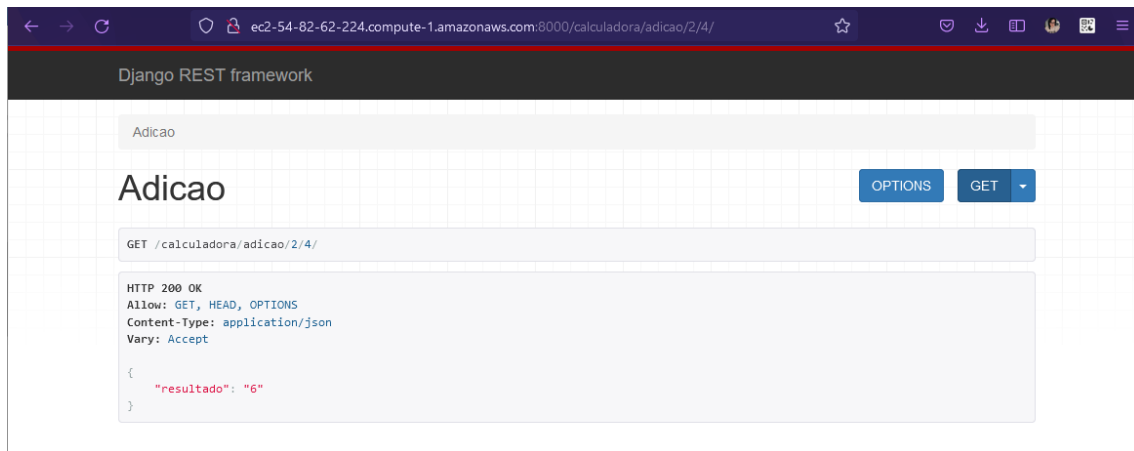


Figura 10 - Executando a função de Adição via AWS

O mesmo será feito para subtração (Figura 11), multiplicação (Figura 12) e divisão (Figura 13).

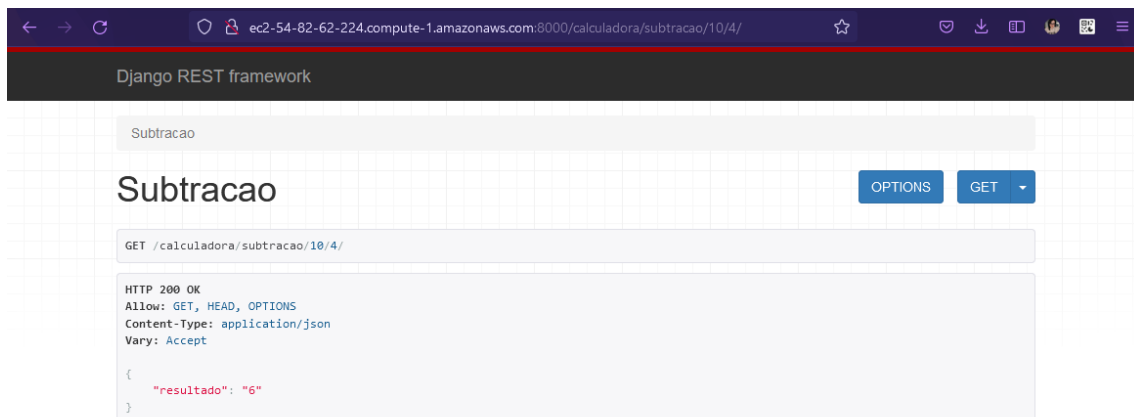


Figura 11 - Executando a função de Subtração via AWS

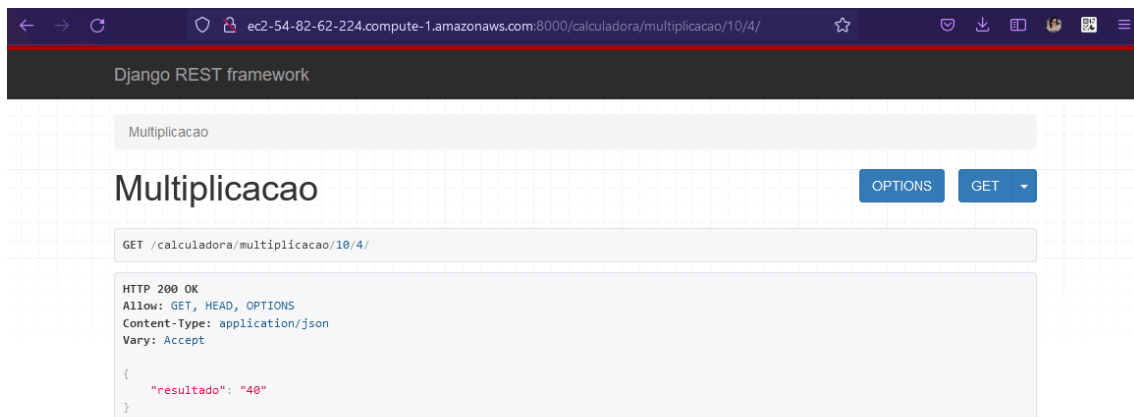


Figura 12 - Executando a função de Multiplicação via AWS

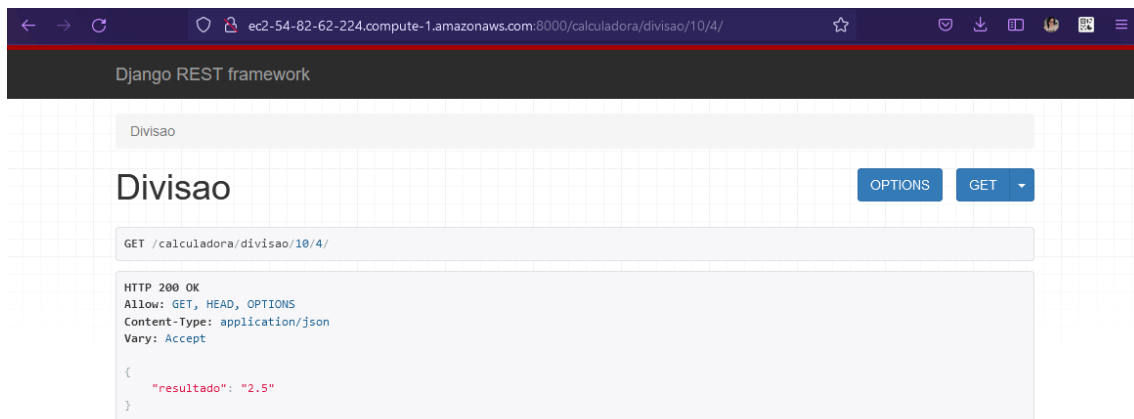


Figura 13 - Executando a função de Divisão via AWS



Figura 14 - Erro mostrado na tela caso seja divisão por zero

### 1.5. Excluir a máquina virtual

Para excluir uma instância, basta clicar em “Estado da Instância” e depois em “Encerrar instância” (Figura 15).

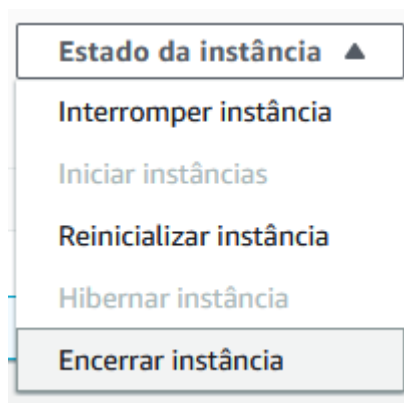


Figura 15 - Encerrando Instância

Após um instante, a página atualizará e o status da Instância aparecerá como “Encerrado” antes de desaparecer da página por completo (Figura 16).



Instâncias (2) <a href="#">Informações</a>				
<input type="text" value="Find instância by attribute or tag (case-sensitive)"/>				
<input type="checkbox"/>	Name ▾	ID de instância	Estado da inst... ▾	Tipo de inst... ▾
<input type="checkbox"/>	exemplo_unifil...	<a href="#">i-0b7a48fe161b5ca7d</a>	⊖ Encerrado 🔍	t2.micro

Figura 16 - Instância encerrada