# Rodar API-Terraform

**Objetivos**

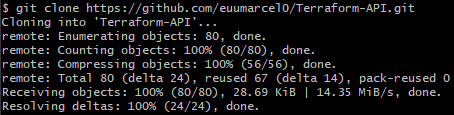
Rodar a API em Flask pelo VScode :

– Subir projeto do git

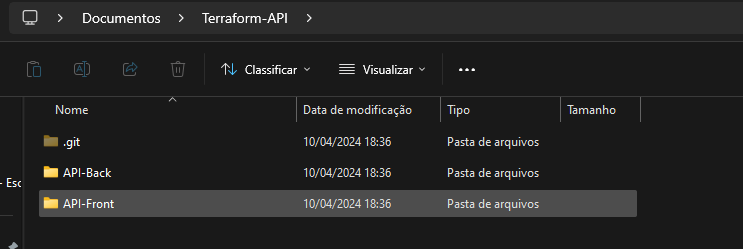
– Tratar erros

– Rodar no Postman

**Tarefa 1: Subir projeto do git**

1. No Explorador de Arquivos abra Documentos
2. Clique com o botão direito do mouse, Mostrar mais opções, Open Git Bash hare
3. Com o terminal do git aberto digite :   
   git clone <https://github.com/euumarcel0/Terraform-API.git>  
   e pressione Enter  
     
   

Vai ser criado uma pasta chamada “Terraform-API” acesse ela



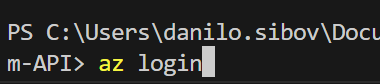
Clique com o botão direito em qualquer lugar vazio da pasta, Mostrar mais opções e clique Abrir com o code

Instalar bibliotecas:

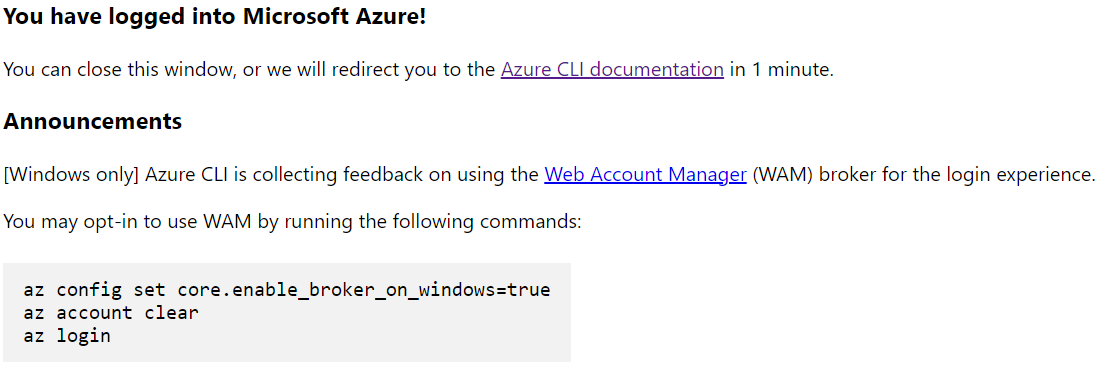
pip install flask\_cors

**Tarefa 2: Configurar conta Azure**

No terminal executar: az login preencher seu do @senaisp.edu.br

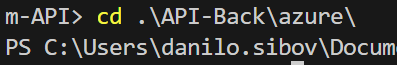


Login realizado com sucesso:

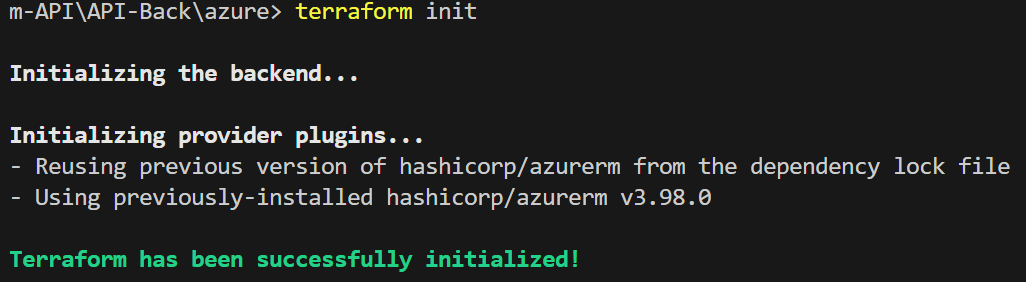


**Tarefa 3: Por provedor acessar o diretório ex.:**

**API-Back/azure:**

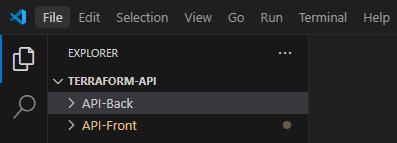


**Digitar o comando: terraform init**

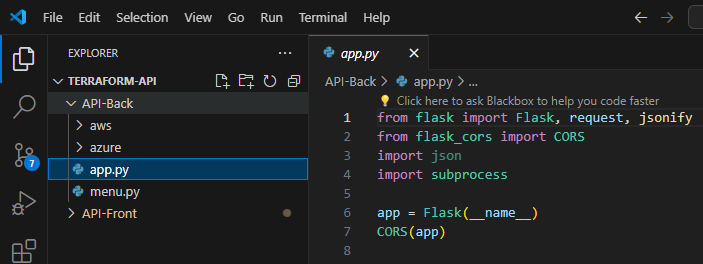


**Tarefa 4: Subir projeto do git**

1. Assim que abrir o **VSCODE** vamos tratar de erros

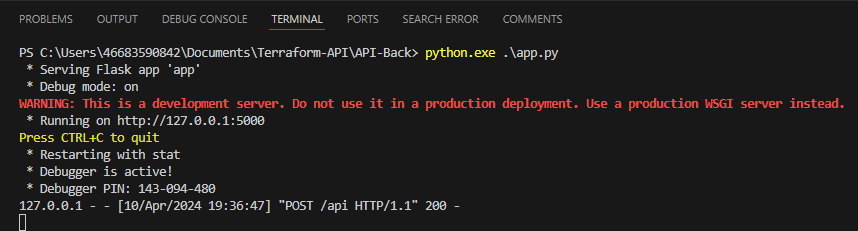


1. Vamos selecionar API-Back e abrir o app.py



1. No terminal vamos dar o comando : cd .\API-Back\   
   Vamos rodar a API

Comando: python.exe .\app.py



Vamos abrir o POSTMAN e colocar a url com o endpoint:

<http://localhost:5000/menu>, selecionar o método GET, Clicar em Body, Clicar em raw e definir os parâmetros:   
Botão: SEND

{

    "aws": {

        "create": [

            "vpc",

            "subnet\_publica",

            "subnet\_privada",

            "internet\_gateway",

            "route\_table",

            "route\_table\_association",

            "security\_group\_linux",

            "security\_group\_windows",

            "ec2\_linux",

            "ec2\_windows"

        ],

        "destroy": "Destruir todos os recursos AWS"

    },

    "azure": {

        "create": [

            "resource\_group",

            "storage\_account",

            "virtual\_network",

            "subnet\_publica",

            "subnet\_privada",

            "network\_security\_linux\_az",

"network\_security\_windows\_az",

            "public\_ip\_linux",

            "public\_ip\_windows",

            "interface\_de\_rede\_linux",

            "interface\_de\_rede\_windows",

            "linux\_virtual\_machine",

            "windows\_virtual\_machine"

        ],

        "destroy": "Destruir todos os recursos Azure"

    }

}

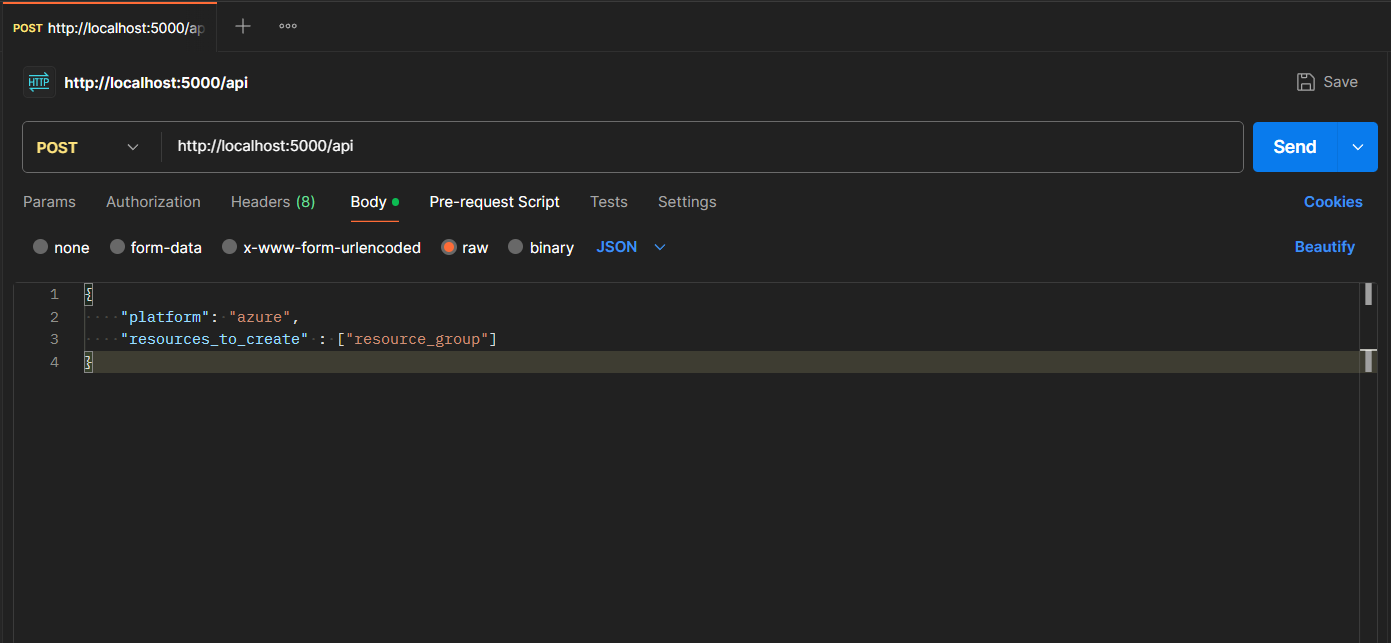
1. Vamos abrir o POSTMAN e colocar a URL com o ENDPOINT: <http://localhost:5000/api>, selecionar o método POST, Clicar em Body, Clicar em raw e definir os parâmetros:

**{**

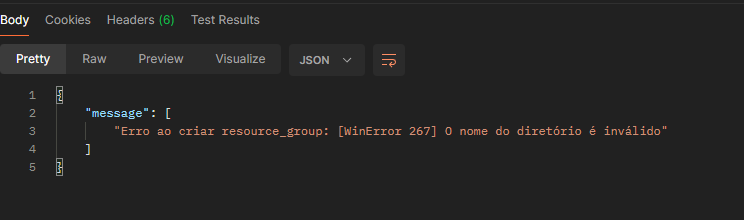
**"platform": "azure",**

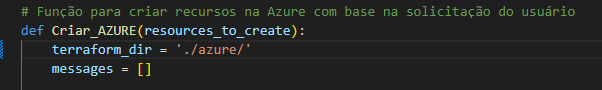
**"resources\_to\_create" : ["storage\_account"]**

**}**

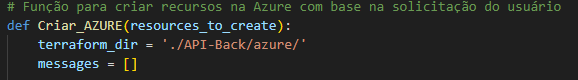


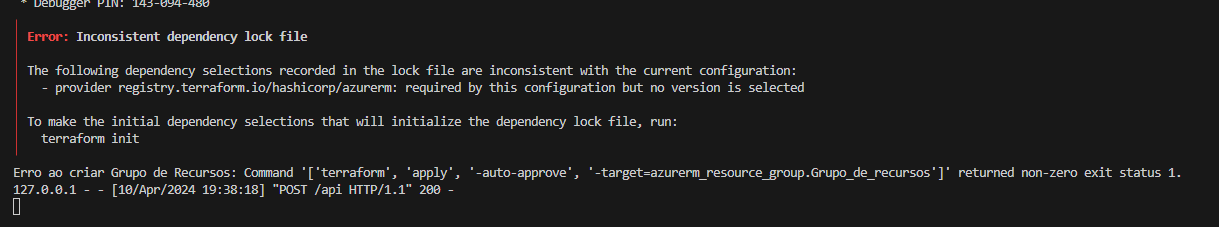
Vamos rodar a requisição de criar um Grupo de recursos

  
Vai apresentar um erro, agora vamos começar o tratamento de erros

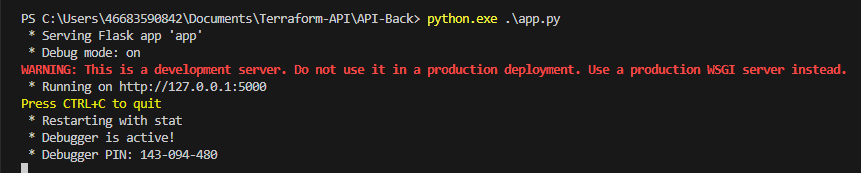
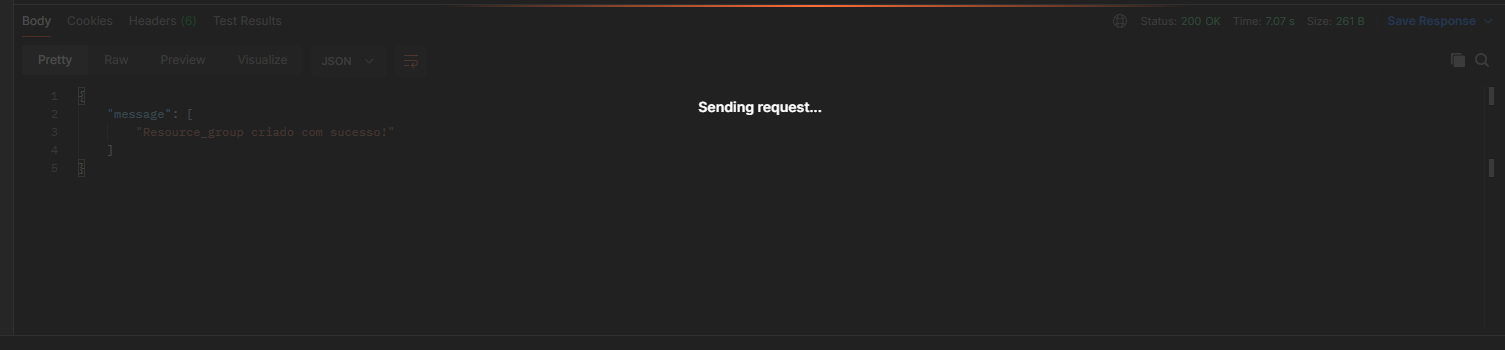
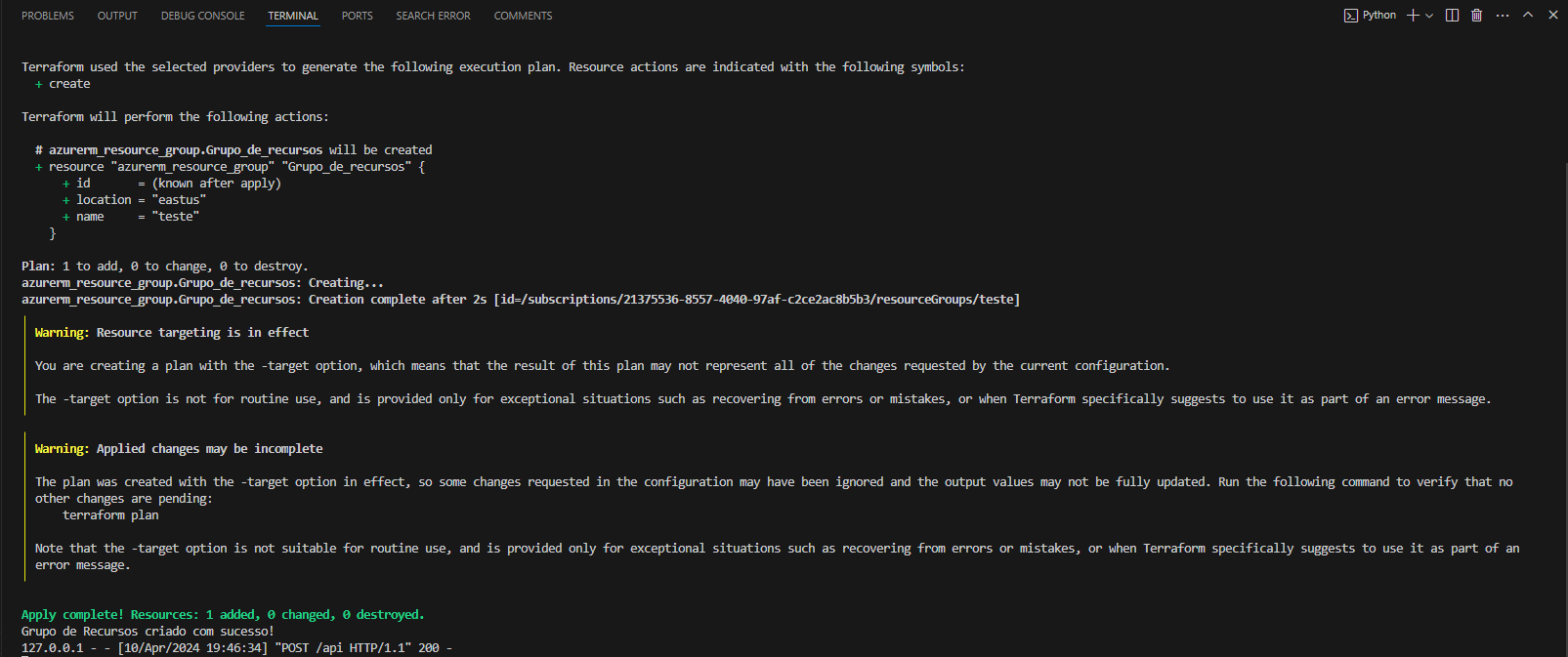
1. Possível erro rotas:  
no seu app.py vamos na linha 101 e 226  
  
Linhas 108: vamos ajustar a rota adicionando API-Back  
  


Resultado:



Mesmo rodando no POSTMAN pode dar esse erro:   
  
  
  
2. Possível erro de INIT:

vamos parar a API no terminal com Ctrl + C  
  
AZURE:   
vamos entrar no diretório da Azure   
Comando : cd .\azure\  
e executar o Comando: terraform init  
em seguida vamos executar: cd .. ( dois pontos )   
   
AWS  
vamos entrar no diretório da Aws  
Comando : cd .\aws\  
e executar o Comando: Terraform init ( Não esqueça de trocar as informações do aws detalhes no arquivo credentials da pasta .aws)   
em seguida vamos executar: cd .. ( dois pontos )

Agora Vamos executar a API novamente:   
Comando: python.exe .\app.py  
  
  
Vamos voltar no POSTMAN para executar novamente**:  
  
  
  
**Vemos que já está funcionando

**Tarefa 5: Rodar no Postman**

Temos um arquivo chamado menu.py no nosso projeto, podemos usar como base para rodar no POSTMAN  
  
exemplo de requisição:  
 **{**

**"platform": "azure",**

**"resources\_to\_create" : ["resource\_group"]**

**}**

Primeiro definimos a plataforma com o parâmetro “platform”, em seguida definimos o recurso que desejamos criar com o parâmetro “resources\_to\_create”  
  
no arquivo menu.py temos as criações   
( lembre-se que o recurso desejado deve estrar em “(aspas duplas)” e [Colchetes]

**Tarefa 6: Testes com Postman**

TESTES DE API – 11-04-2024

Testando a criação de conta de armazenamento:

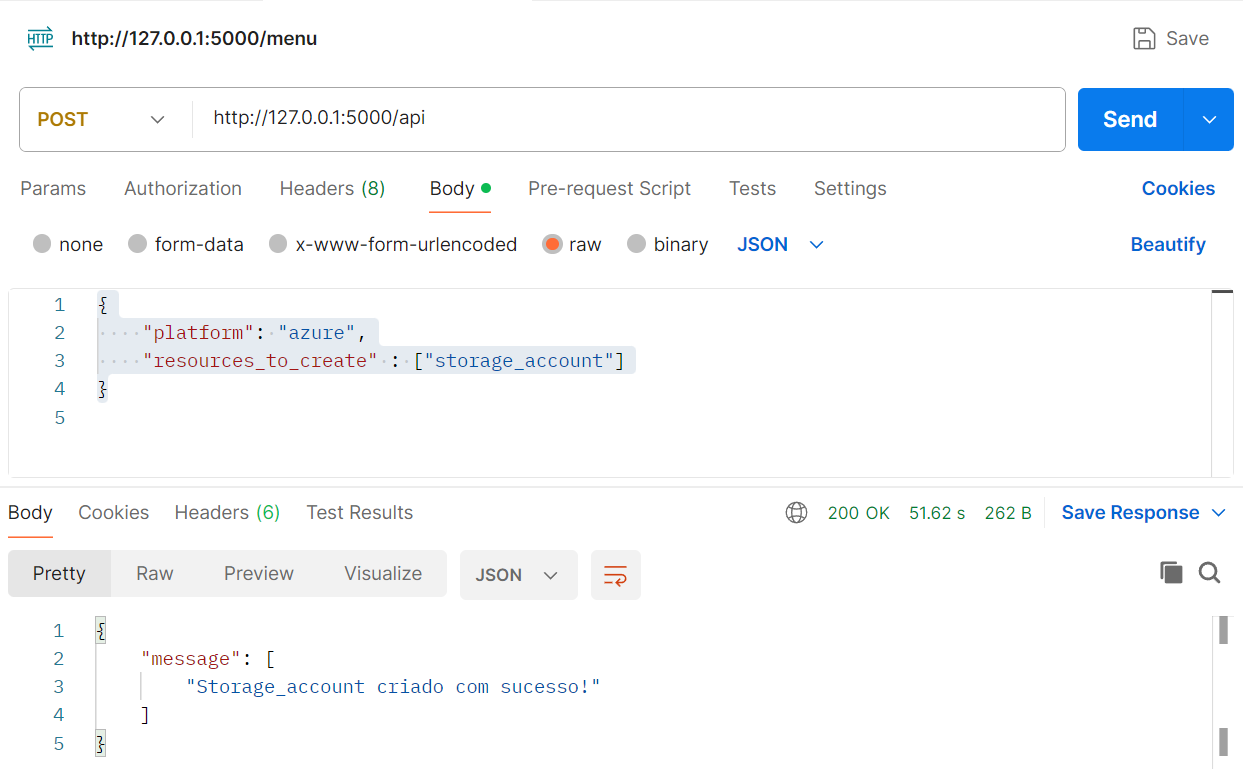
JSON que foi inserido no POSTMAN:  
**{**

**"platform": "azure",**

**"resources\_to\_create" : ["storage\_account"]**

**}**

Endpoint: <http://127.0.0.1:5000/api> – Método: POST



Sucesso no console: REST HTTP 200 e mensagem:

Conta de Armazenamento criada com sucesso!

