**🚀 Projeto: App Web Flask com CI/CD no Azure DevOps**

**📌 Objetivo**

Criar uma aplicação Flask simples e configurar um pipeline que:

1. Testa o código (CI).
2. Faz deploy automático em uma VM Linux no Azure (CD).

# 🧱 ETAPA 1 — Criar a Aplicação Flask

1. Instalar dependências:

python -m venv venv

source venv/bin/activate # Linux/macOS

pip install flask

1. Criar estrutura do projeto:

flask-azure-devops/

├── app.py

├── requirements.txt

├── .gitignore

└── templates/

└── index.html

1. Código simples do app (app.py):

from flask import Flask, render\_template

app = Flask(\_\_name\_\_)

@app.route("/")

def home():

return render\_template("index.html")

if \_\_name\_\_ == "\_\_main\_\_":

app.run(host="0.0.0.0", port=5000)

1. HTML simples (templates/index.html):

<!DOCTYPE html>

<html>

<head>

<title>Flask CI/CD App</title>

</head>

<body>

<h1>Olá! Deploy via Azure DevOps funcionando!</h1>

</body>

</html>

1. Criar requirements.txt:

pip freeze > requirements.txt

1. Adicionar .gitignore:

venv/

\_\_pycache\_\_/

\*.pyc

# 🗂️ ETAPA 2 — Subir o Projeto para o GitHub

1. Crie um repositório no GitHub.
2. Inicialize o repositório local:

git init

git add .

git commit -m "Flask app inicial"

git remote add origin https://github.com/seu-usuario/seu-repo.git

git push -u origin master

# ⚙️ ETAPA 3 — Configurar Azure DevOps

1. Criar Projeto no Azure DevOps

Acesse: <https://dev.azure.com>

Crie um novo projeto.

1. Conectar com o GitHub

 Vá em **Pipelines > New Pipeline**

 Escolha **GitHub** como repositório.

 Autorize o Azure a acessar seu GitHub.

# 🏗️ ETAPA 4 — Criar Pipeline de CI/CD (YAML)

Crie o arquivo .azure-pipelines.yml no seu repositório com:

trigger:

- master

pool:

vmImage: 'ubuntu-latest'

steps:

- task: UsePythonVersion@0

inputs:

versionSpec: '3.x'

- script: |

python -m venv venv

source venv/bin/activate

pip install -r requirements.txt

displayName: 'Instalar dependências'

- script: |

echo "Testes placeholder - insira seus testes aqui"

displayName: 'Executar Testes'

- script: |

echo "Deploy automático seria configurado aqui"

displayName: 'Deploy'

**Nota:** Por enquanto, isso é apenas simulado. O próximo passo mostrará como de fato fazer o deploy.

# ☁️ ETAPA 5 — Fazer o Deploy em Azure VM (opcional)

Se quiser simular um ambiente real:

**1. Criar uma VM Linux no Azure (ex: Ubuntu 20.04).**

**2. Instalar Python, Git e Flask:**

sudo apt update

sudo apt install python3-pip git -y

pip3 install flask

**3. Gerar chave SSH e adicionar ao Azure DevOps como “Service Connection”.**

**4. Substituir o passo de Deploy no YAML:**

- task: CopyFilesOverSSH@0

inputs:

sshEndpoint: 'NomeDaConexaoSSH'

sourceFolder: '$(Build.SourcesDirectory)'

contents: '\*\*'

targetFolder: '/home/usuario/app'

- script: |

ssh usuario@ip-da-vm 'cd /home/usuario/app && nohup python3 app.py &'

displayName: 'Rodar app Flask na VM'

# ✅ ETAPA 6 — Validar

1. Acesse o endereço da VM no navegador: http://<ip-da-vm>:5000.
2. Edite o repositório no GitHub e veja o Azure DevOps fazer o build + deploy automaticamente.

# 📚 Dicas de Aprendizado

* Aprenda comandos básicos de **Flask**.
* Experimente com **testes unitários** (ex: pytest).
* Pratique boas práticas de CI/CD: versionamento, testes, rollback.
* Evolua para usar **Docker** e **Azure App Services** no lugar de VM futuramente.