

Guia Completo: Do Código ao Deploy - Lista de Tarefas na AWS

Visão Geral

Este guia te levará desde o código já criado até a aplicação funcionando na AWS com CI/CD completo. Siga cada etapa na ordem apresentada.

FASE 1: Preparação Local e Teste

Passo 1.1: Verificar Estrutura do Projeto

Confirme que sua estrutura está assim:

```
computacaoemnuvem20251/
├── .github/workflows/deploy.yml
├── backend/
│   ├── main.py
│   ├── requirements.txt
│   └── .env
├── frontend/
│   ├── public/index.html
│   ├── src/
│   │   ├── App.js
│   │   ├── index.js
│   │   └── index.css
│   ├── package.json
│   └── .env.production
├── deploy.sh
├── test-local.sh
├── .gitignore
├── README.md
└── projeto.txt
```

Passo 1.2: Testar Backend Localmente

1. **Abra o PowerShell** na pasta do projeto
2. **Navegue para o backend:**

```
cd backend
```

3. **Crie ambiente virtual:**

```
python -m venv venv
```

4. Ative o ambiente virtual:

```
venv\Scripts\activate
```

5. Instale dependências:

```
pip install -r requirements.txt
```

6. Execute o backend:

```
python main.py
```

7. Teste no navegador:

- Acesse: <http://localhost:8000>
- Deve mostrar: `{"message": "Lista de Tarefas API"}`
- Acesse: <http://localhost:8000/docs>
- Deve mostrar a documentação da API

8. Pare o servidor (Ctrl+C)

Passo 1.3: Testar Frontend Localmente

1. Abra NOVO PowerShell na pasta do projeto

2. Navegue para o frontend:

```
cd frontend
```

3. Instale dependências:

```
npm install
```

4. Execute o frontend:

```
npm start
```

5. Teste no navegador:

- Acesse: `http://localhost:3000`
- Deve mostrar a interface da lista de tarefas
- **IMPORTANTE:** Neste momento, a comunicação com a API pode falhar (normal)

6. Pare o servidor (Ctrl+C)

Passo 1.4: Teste Integrado Local

1. Execute backend (PowerShell 1):

```
cd backend
venv\Scripts\activate
python main.py
```

2. Execute frontend (PowerShell 2):

```
cd frontend
npm start
```

3. Teste completo:

- Acesse: `http://localhost:3000`
- Tente criar uma tarefa
- Deve funcionar completamente

FASE 2: Configuração da AWS

Passo 2.1: Acessar AWS Academy

1. Entre no AWS Academy Learner Lab
2. Clique em "Start Lab"
3. Aguarde o indicador ficar verde
4. Clique em "AWS" para acessar o console

Passo 2.2: Criar Instância EC2

1. No console AWS, acesse EC2
2. Clique em "Launch Instance"
3. Configure:
 - **Name:** `todo-app-server`
 - **AMI:** Amazon Linux 2 AMI (HVM)
 - **Instance type:** `t2.micro`

- **Key pair:** Criar nova ou usar existente
 - Se criar nova: **BAIXE o arquivo .pem**
- **Network settings:**
 - Criar security group novo
 - Allow SSH (22) from: My IP
 - Allow HTTP (80) from: Anywhere
 - Allow HTTPS (443) from: Anywhere

4. Launch Instance

Passo 2.3: Configurar Acesso SSH

1. **Anote o IP público** da instância

2. **Configure a chave SSH:**

- Mova o arquivo **.pem** para uma pasta segura
- **No PowerShell:**

```
# Navegue até onde está o arquivo .pem
icacls "seu-arquivo.pem" /inheritance:r
icacls "seu-arquivo.pem" /grant:r "%username%:R"
```

3. **Teste conexão SSH:**

```
ssh -i "seu-arquivo.pem" ec2-user@SEU-IP-PUBLICO
```

FASE 3: Configuração do GitHub

Passo 3.1: Criar Repositório GitHub

1. **Acesse GitHub.com**
2. **Crie novo repositório:**
 - Nome: **computacao-nuvem-2025** (ou similar)
 - **PRIVADO**
 - Não inicialize com README (já temos)

Passo 3.2: Conectar Repositório Local

No PowerShell, na pasta do projeto:

1. **Inicializar Git** (se não feito):

```
git init
git branch -M main
```

2. Adicionar remote:

```
git remote add origin https://github.com/SEU-USUARIO/SEU-REPOSITORIO.git
```

3. Primeiro commit:

```
git add .  
git commit -m "Initial commit: Todo app with FastAPI and React"  
git push -u origin main
```

Passo 3.3: Configurar Secrets do GitHub

1. No GitHub, vá para o repositório
2. **Settings > Secrets and variables > Actions**
3. **New repository secret** para cada:

EC2_SSH_KEY:

- Abra o arquivo `.pem` no bloco de notas
- Copie TODO o conteúdo (incluindo as linhas BEGIN/END)
- Cole como valor do secret

EC2_HOST:

- O IP público da sua instância EC2

EC2_USER:

- Valor: `ec2-user`

FASE 4: Preparação do Servidor

Passo 4.1: Configuração Manual Inicial

Conecte via SSH e execute:

```
# Atualizar sistema  
sudo yum update -y  
  
# Instalar Python 3 e pip  
sudo yum install -y python3 python3-pip  
  
# Instalar Node.js  
curl -fsSL https://rpm.nodesource.com/setup_18.x | sudo bash -  
sudo yum install -y nodejs
```

```
# Instalar PM2
sudo npm install -g pm2

# Instalar Git
sudo yum install -y git

# Criar diretório da aplicação
mkdir -p /home/ec2-user/app

# Testar instalações
python3 --version
node --version
npm --version
pm2 --version
```

Passo 4.2: Tornar Script Executável

No PowerShell local, atualize o script de deploy:

```
git add .
git commit -m "Update deploy script permissions"
git push
```

FASE 5: Primeiro Deploy Manual

Passo 5.1: Deploy Manual para Teste

Na sua máquina local, no PowerShell:

1. Clone para o servidor:

```
scp -i "seu-arquivo.pem" -r . ec2-user@SEU-IP-PUBLICO:/home/ec2-user/app/
```

2. Execute deploy manual:

```
ssh -i "seu-arquivo.pem" ec2-user@SEU-IP-PUBLICO
```

No servidor:

```
cd /home/ec2-user/app
chmod +x deploy.sh
./deploy.sh
```

Passo 5.2: Verificar Deploy

1. Teste os serviços:

```
# Verificar PM2
pm2 status

# Verificar Nginx
sudo systemctl status nginx

# Teste API
curl http://localhost:8000/health
```

2. Teste no navegador:

- Acesse: <http://SEU-IP-PUBLICO>
- Deve mostrar a aplicação funcionando

FASE 6: Ativação do CI/CD

Passo 6.1: Testar Pipeline

1. Faça uma alteração simples (ex: no README.md):

```
# Edite o README.md e adicione uma linha
git add .
git commit -m "Test CI/CD pipeline"
git push
```

2. Monitore no GitHub:

- Vá para **Actions** no repositório
- Veja o workflow executando
- Verifique se ambos os jobs passaram

Passo 6.2: Verificar Deploy Automático

1. Se o pipeline passou:

- Acesse sua aplicação no navegador
- Verifique se a alteração apareceu

2. Se houver erros:

- Verifique os logs no GitHub Actions
- Verifique conexão SSH
- Verifique se os secrets estão corretos

🔗 FASE 7: Demonstração e Entrega

Passo 7.1: Preparar Demonstração

1. Funcionalidades para demonstrar:

- Criar nova tarefa
- Marcar como concluída
- Deletar tarefa
- Mostrar persistência (refresh da página)

2. CI/CD para demonstrar:

- Fazer uma alteração visual simples
- Commit e push
- Mostrar pipeline executando
- Mostrar mudança no ar automaticamente

Passo 7.2: Preparar Video/Apresentação

Roteiro sugerido (5-10 minutos):

1. Mostrar aplicação funcionando (2 min)

- Demonstrar CRUD de tarefas
- Mostrar responsividade

2. Mostrar código no GitHub (2 min)

- Estrutura do projeto
- Frontend e Backend separados
- Arquivos de CI/CD

3. Demonstrar CI/CD (3 min)

- Fazer alteração no código
- Commit e push
- Mostrar Actions executando
- Mostrar atualização automática

4. Mostrar infraestrutura AWS (2 min)

- Instância EC2
- Security Groups
- Logs da aplicação

Passo 7.3: Entrega Final

Submeter:

1. **Link do repositório GitHub** (com acesso de leitura para o professor)
2. **URL da aplicação funcionando**

3. Vídeo de demonstração (se solicitado)

TROUBLESHOOTING

Problemas Comuns e Soluções:

1. SSH Connection Failed

```
# Verificar permissões da chave
chmod 400 seu-arquivo.pem

# Verificar security group
# Porta 22 deve estar aberta para seu IP
```

2. GitHub Actions Falha

- Verificar se todos os 3 secrets estão configurados
- Verificar se a chave SSH está correta (incluindo quebras de linha)
- Verificar se o IP do EC2 está correto

3. Aplicação não carrega

```
# No servidor, verificar logs
pm2 logs todo-backend
sudo tail -f /var/log/nginx/error.log

# Reiniciar serviços
pm2 restart todo-backend
sudo systemctl restart nginx
```

4. Database errors

```
# Recriar banco de dados
cd /home/ec2-user/app/backend
python3 -c "from main import Base, engine; Base.metadata.create_all(bind=engine)"
```

CHECKLIST FINAL

Antes da Entrega:

- ☐ Aplicação funciona localmente
- ☐ Instância EC2 criada e configurada

- ☐ Repositório GitHub privado criado
- ☐ Secrets do GitHub configurados
- ☐ Deploy manual funciona
- ☐ CI/CD pipeline funciona
- ☐ Aplicação acessível via IP público
- ☐ Demonstração preparada
- ☐ README.md completo
- ☐ Video/apresentação pronta (se necessário)

URLs para Testar:

- ☐ <http://SEU-IP-PUBLICO> - Frontend funciona
- ☐ <http://SEU-IP-PUBLICO/docs> - API docs acessível
- ☐ GitHub Actions - Pipeline verde
- ☐ Funcionalidades CRUD funcionam



Conclusão

Seguindo este guia, você terá:

- ☒ Aplicação web completa (Frontend + Backend + BD)
- ☒ Deploy automático na AWS
- ☒ CI/CD funcionando
- ☒ Documentação completa
- ☒ Demonstração pronta

Tempo estimado total: 2-4 horas

Boa sorte com o projeto! 🚀