

DOCUMENTAÇÃO TÉCNICA - Emprega Mais – CI/CD com GitHub Actions e Azure

1. Título do Projeto e Integrantes

Projeto: Cidades ESGInteligentes

Integrantes:

Felipe de Oliveira – RM556069

Pedro Barberini Rodrigues Carvalho – RM558432

Robert Ferreira dos Santos – RM559199

Gabriel Sá de Souza – RM554706

Henrique Bueno Rizzo – RM556723

2. Descrição do Pipeline (CI/CD)

O pipeline de CI/CD foi desenvolvido utilizando a ferramenta GitHub Actions, que automatiza as etapas de build, teste e deploy do projeto. Essa automação garante um fluxo de entrega contínua eficiente e confiável.

Etapas do Pipeline:

1. Checkout do repositório: Recupera o código-fonte mais recente do GitHub.
2. Configuração do ambiente: Instalação do JDK 21 e setup do Maven para o build.
3. Build e Testes Automatizados: Execução dos testes via mvn clean test, garantindo integridade do sistema.
4. Empacotamento: Geração do arquivo .jar do projeto para deploy.
5. Deploy Automatizado no Azure Web App: Publicação da aplicação através da ação azure/webapps-deploy@v3.

Secrets Configurados:

AZURE_WEBAPP_PUBLISH_PROFILE – Credenciais do Azure para deploy seguro.

SPRING_DATASOURCE_URL – URL de conexão do banco PostgreSQL.

SPRING_DATASOURCE_USERNAME – Usuário do banco.

SPRING_DATASOURCE_PASSWORD – Senha do banco.

Essas variáveis garantem segurança e automação durante todo o processo de deploy.

3. Docker: Arquitetura, Comandos e Imagem Criada

A containerização do projeto foi feita com Docker e Docker Compose, permitindo reproduzir o ambiente de produção localmente.

Dockerfile:

```
FROM maven:3.9-eclipse-temurin-21 AS build
```

```
WORKDIR /app
```

```
COPY pom.xml .
RUN mvn dependency:go-offline -B
COPY src ./src
RUN mvn clean package -DskipTests

FROM eclipse-temurin:21-jre
WORKDIR /app
COPY --from=build /app/target/*.jar app.jar
EXPOSE 8080
ENTRYPOINT ["java", "-jar", "app.jar"]
```

Comandos Utilizados:

```
docker compose up -d --build
docker compose ps
docker compose down
```

A arquitetura é composta por dois containers principais:

- Aplicativo Java Spring Boot (porta 8080)
- Banco de Dados PostgreSQL

Com isso, o ambiente pode ser iniciado rapidamente, garantindo escalabilidade e consistência.

4. Prints do Pipeline Rodando

As imagens a seguir comprovam o sucesso do pipeline CI/CD:

- Execução completa do GitHub Actions com as etapas de build, test e deploy.
- Azure Web App em funcionamento, com a aplicação publicada em produção.
(As imagens estão incluídas ao final deste documento como evidências visuais.)

5. Prints dos Ambientes Staging e Produção

Foram realizados testes de implantação em dois ambientes distintos:

- Staging: ambiente de validação interna e testes de integração.
- Produção (Azure Web App): ambiente final hospedado com sucesso no Azure, disponível via URL pública.

6. Desafios Encontrados e Soluções

Desafio 1: Integração do pipeline com o Azure.

Resolução: Configuração correta do AZURE_WEBAPP_PUBLISH_PROFILE como secret no GitHub.

Desafio 2: Conflito de portas locais (8080).

Resolução: Ajuste no docker-compose.yml para mapear a porta 8081 no host local.

Desafio 3: Testes automatizados falhando na inicialização do contexto Spring.

Resolução: Atualização das dependências e ajustes de contexto no application.properties.

7. Checklist de Entrega

- | Item | OK |
 - |-----|----|
 - | Projeto compactado em .ZIP com estrutura organizada |
 - | Dockerfile funcional | |
 - | docker-compose.yml ou arquivos Kubernetes | |
 - | Pipeline com etapas de build, teste e deploy | |
 - | README.md com instruções e prints | |
 - | Documentação técnica com evidências (PDF ou PPT) | |
 - | Deploy realizado nos ambientes staging e produção | |

8. Evidências Visuais

Summary

deploy succeeded in 1 minute in 30s

Search logs

Jobs

build

deploy

Run details

Usage

Workflow file

deploy

Download artifact from build job.

9 - java-app (ID: 426e85a49, Size: 5708420, Expected Digest: sha256:f3c29f4d77db36e03b230bf7ada99063f073906af4509f71da12e91843ea)

10 Redirecting to Azure download url: https://devopsresultsstate.blob.core.windows.net/actions-results/15247195_7729_4ef1-807a-379aaef977d/workFlow-5ab-rvn-57000ep91ff-504b-41c8-8ff749c5c97/artifacts/ec4112f740bd4b9ab3e64545110da459f0f5e64272ad049d411a8

11 [INFO] [2025-10-14T10:58:22.766432Z] [System] [Information] [System] [System] [System]

12 [INFO] [2025-10-14T10:58:22.766432Z] [System] [Information] [System] [System] [System]

13 [INFO] [2025-10-14T10:58:22.766432Z] [System] [Information] [System] [System] [System]

14 [INFO] [2025-10-14T10:58:22.766432Z] [System] [Information] [System] [System] [System]

15 [INFO] [2025-10-14T10:58:22.766432Z] [System] [Information] [System] [System] [System]

16 Artifactory download completed successfully.

17 Total of 1 artifact(s) downloaded

18 Downloaded artifact has finished successfully

Deploy to Azure Web App

9 Predeployment Step Started

10 Deployment Step Started

11 Starting deployment for web app...

12 Package deployment using onedeploy initiated.

13 [

14 {

15 id: 'benn0027f-0835-4632-8859-0014947d9436'

16 status: 4,

17 statusText: '',

18 author: 'N/A',

19 authorEmail: 'N/A',

20 authorName: 'N/A',

21 deployer: 'onedeploy',

22 message: 'OneDeploy',

23 progress: 0,

24 receivedTime: '2025-10-14T00:58:16.138003Z',

25 startTime: '2025-10-14T00:58:17.527467Z',

26 endTime: '2025-10-14T00:58:22.766432Z',

27 lastSuccessEndTime: '2025-10-14T00:58:22.766432Z',

28 complete: true,

29 isTemporary: false,

30 isReadyOnly: true,

31 url: 'https://emrep-mais-fecnhobravcay2.scm.brazilsouth-01.azurewebsites.net/api/deployment/benn0027f-0835-4632-8859-0014947d9436/*',

32 logUrl: 'https://emrep-mais-fecnhobravcay2.scm.brazilsouth-01.azurewebsites.net/api/deployments/benn0027f-0835-4632-8859-0014947d9436/*log',

33 siteName: 'emrep-mais',

34 buildSummary: { errors: [], warnings: [] }

35]

36 Deploy logs can be viewed at https://emrep-mais-fecnhobravcay2.scm.brazilsouth-01.azurewebsites.net/api/deployment/benn0027f-0835-4632-8859-0014947d9436/*log

37 Successfully deployed web package to App Service.

38 App Service Application URL: <https://emrep-mais-fecnhobravcay2.brazilsouth-01.azurewebsites.net>

Complete job

Figura: Print sucesso 1

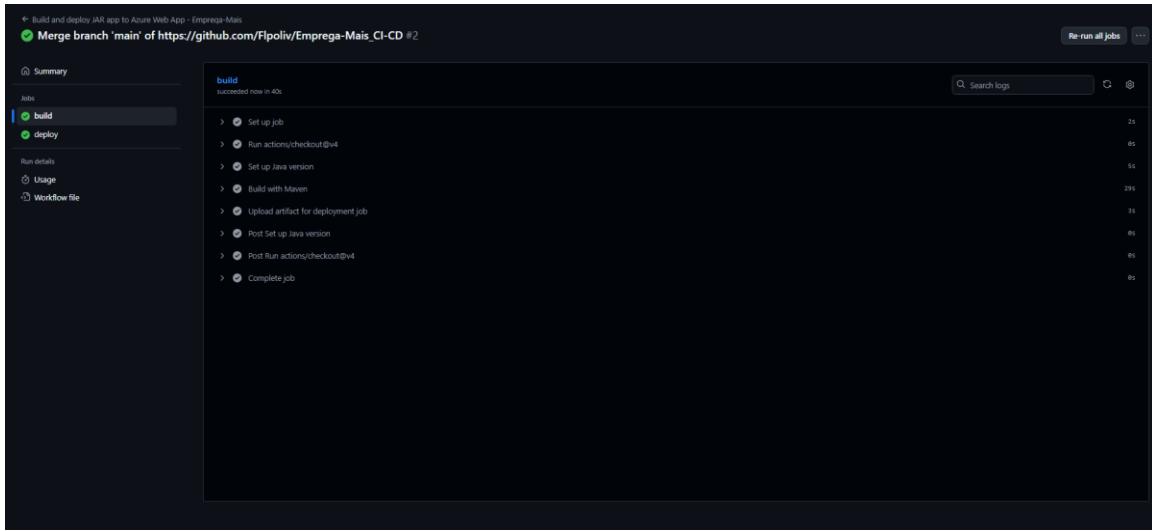


Figura: print sucesso 2

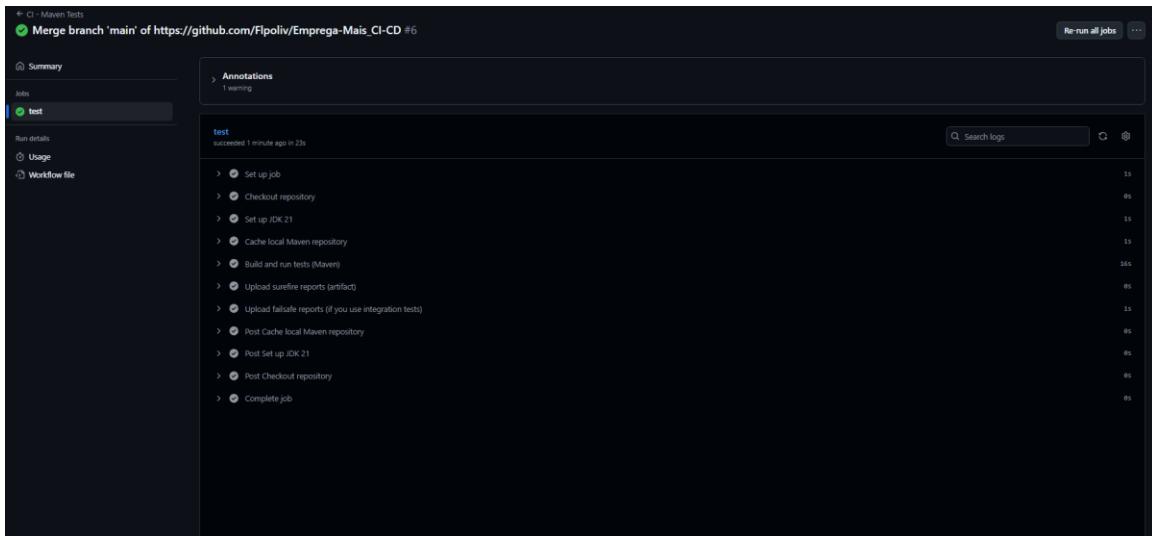


Figura: print sucesso 3

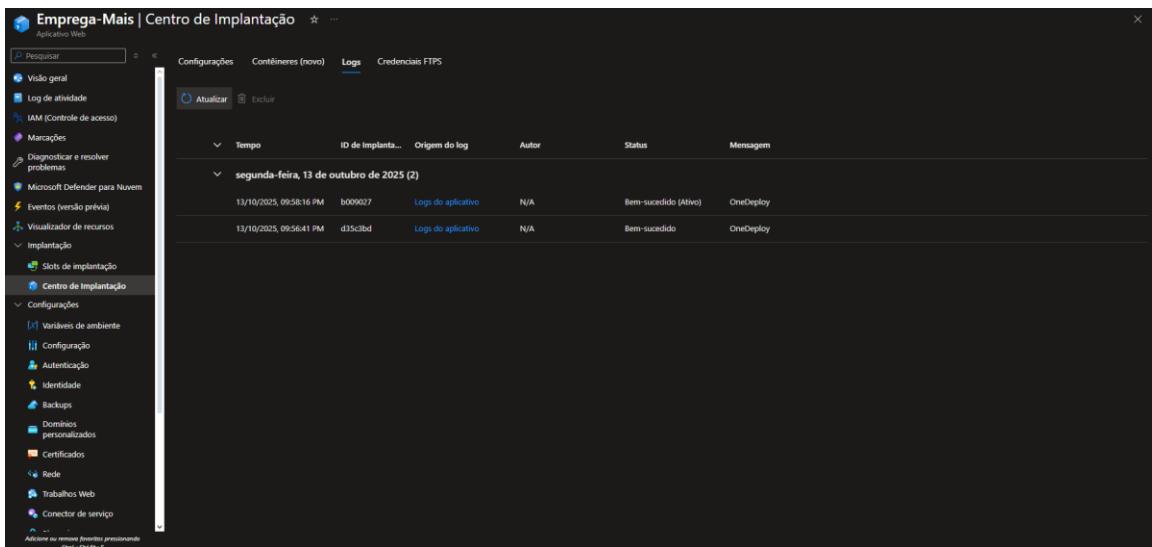


Figura: print azure sucesso

The screenshot shows the 'Repository secrets' section in GitHub. The left sidebar has options for Codespaces, Pages, Security (Advanced Security, Deploy keys), Secrets and variables (Actions, Codespaces, Dependabot). The main area lists four secrets:

Name	Last updated	Action
AZUREAPPSERVICE_PUBLISHPROFILE_98789EBF975340958857788642386F05	13 minutes ago	edit delete
SPRING_DATASOURCE_PASSWORD	1 hour ago	edit delete
SPRING_DATASOURCE_URL	1 hour ago	edit delete
SPRING_DATASOURCE_USERNAME	1 hour ago	edit delete

A green button labeled 'New repository secret' is visible at the top right.

Figura: Secrets e Variables