Fundamentos de DevOps

Aula 09 - Introdução ao CI/CD - Laboratório

Prof. Esp. Guilherme Jorge Aragão da Cruz

guilherme.cruz@alumni.usp.br

linkedin.com/in/guijac

Laboratório

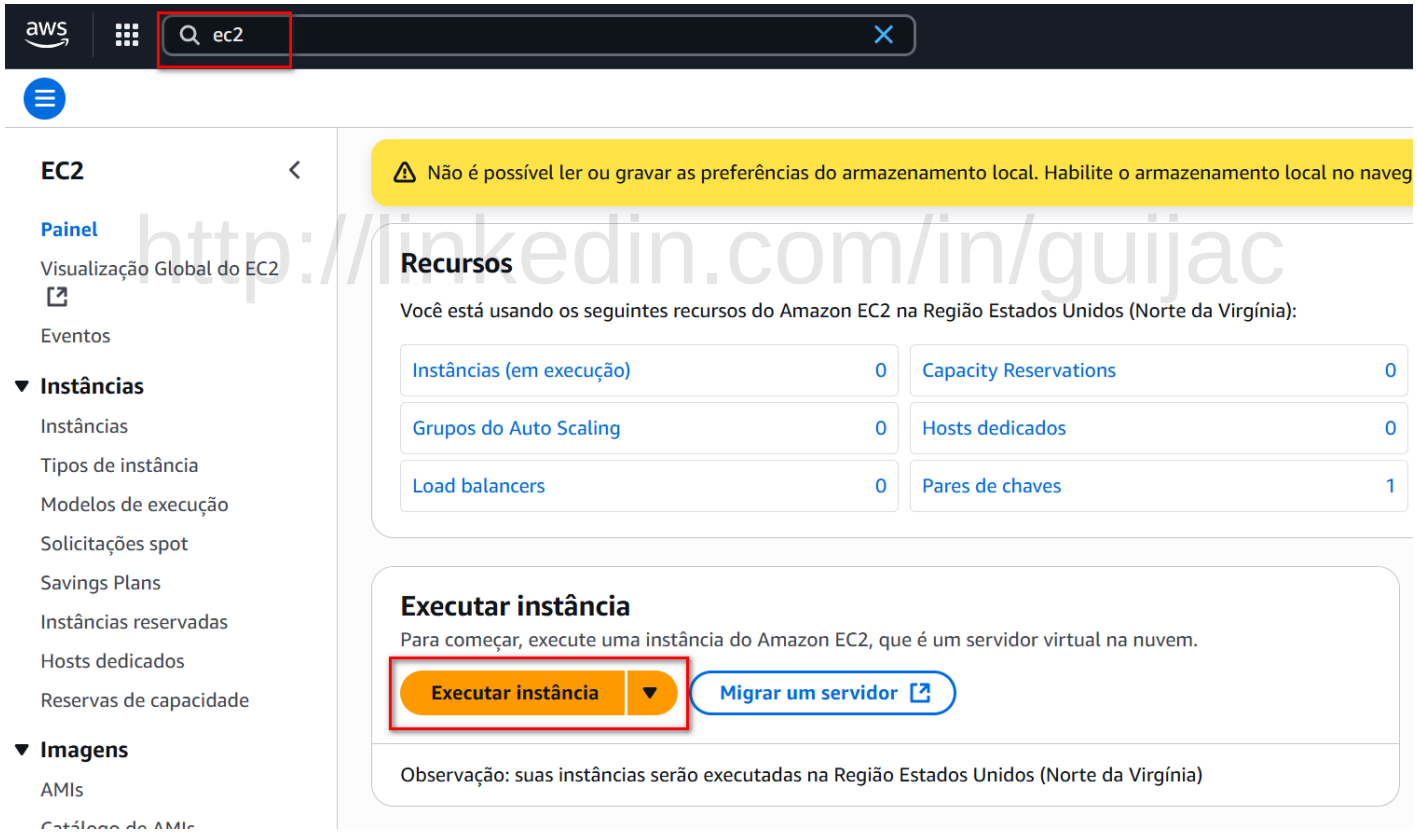
1. Criar um novo projeto em seu Gitlab (ou utilizar o projeto da ADO-02) para desenvolvimento do seu primeiro pipeline de CI;
2. Criar um arquivo .gitlab-ci.yml (**sugestão:** utilizar o arquivo e as extensões apresentadas em aula);
3. Realizar o “push” e acompanhar o pipeline;
4. Alterar o arquivo .gitlab-ci.yml, contemplando uma execução de testes unitários com JUnit;
5. **Desafio: Criar e configurar seu próprio *runner* na AWS.**

Desafio

- **Acesse o console da AWS**. Nesta atividade iremos preparar um ambiente para execução do laboratório (ex.: conta própria ou ambiente educacional, se disponível);
- Iremos criar uma instância “Amazon Elastic Compute Cloud” (EC2), uma VM da AWS com mais de 500 instâncias e opções do processadores, armazenamento, redes e sistemas operacionais.

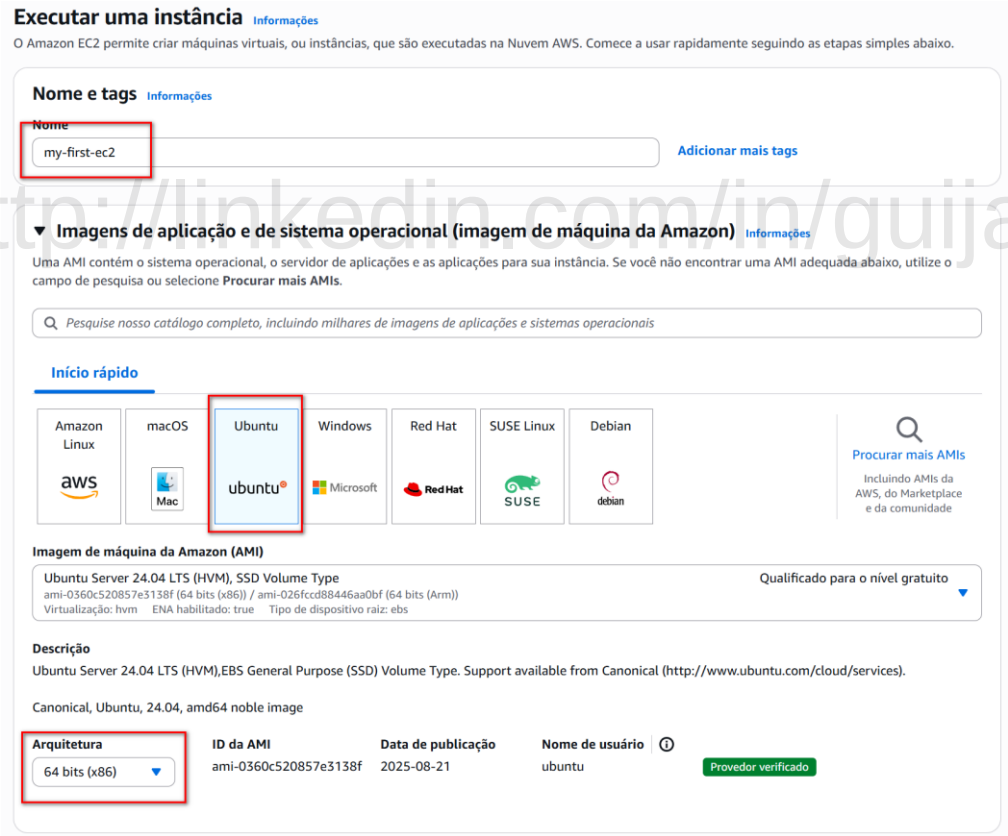
Desafio

- Dentro do menu de pesquisa do Console AWS digite “EC2”, familiarize-se com o ambiente e clique em “Executar instância”:



Desafio

- Dê um nome para seu servidor, vamos criar uma instância Ubuntu, Arquitetura 64 bits (x86) – esta informação será relevante para a instalação do Gitlab Runner:

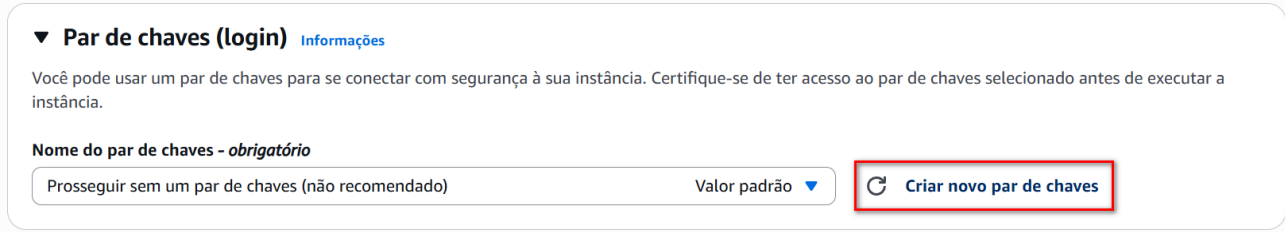


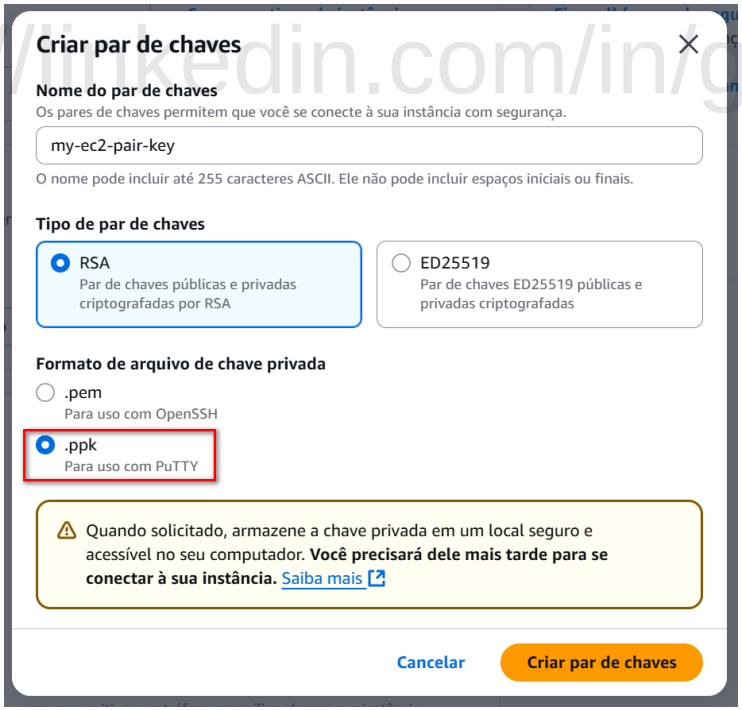
Desafio

- Dada a característica de processamento de um servidor de CI, iremos criar uma instância do tipo “t2.medium”, que contém 2 vCPUs e 4GiB RAM, **opcionalmente**, você pode obter conselhos para nosso caso de uso, para fins de estudo:

Desafio

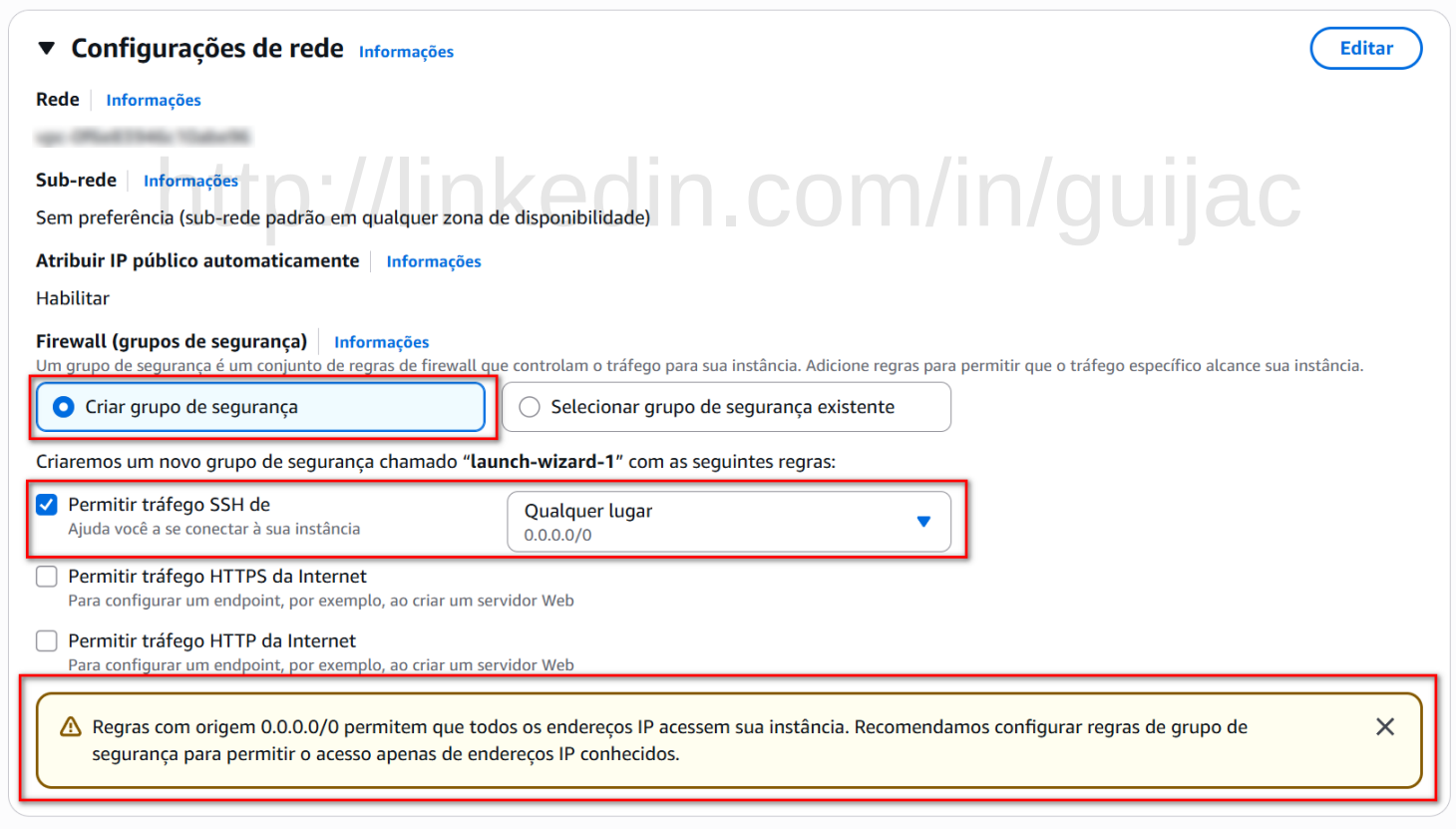
- Crie um par de chaves no formato PuTTY Private Key (.ppk) e realizar seu download (**GUARDE ESTE ARQUIVO COM SUA VIDA!**).





Desafio

- Iremos criar um novo grupo de segurança (*firewall/security group*);
- Notem que uma boa prática é limitar o acesso à IPs específicos, para fins de laboratório seguiremos diferente.



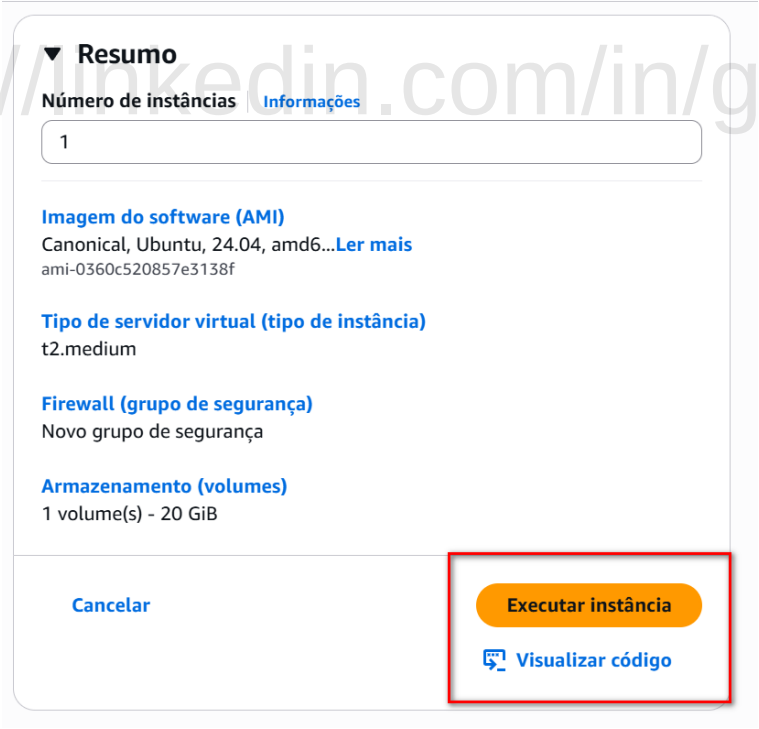
Desafio

- Dada a característica de armazenamento de um Servidor de CI, iremos configurar um armazenamento SSD com 20 GiB:

- Para ambientes de produção faça a reflexão: quanto de armazenamento irei precisar?

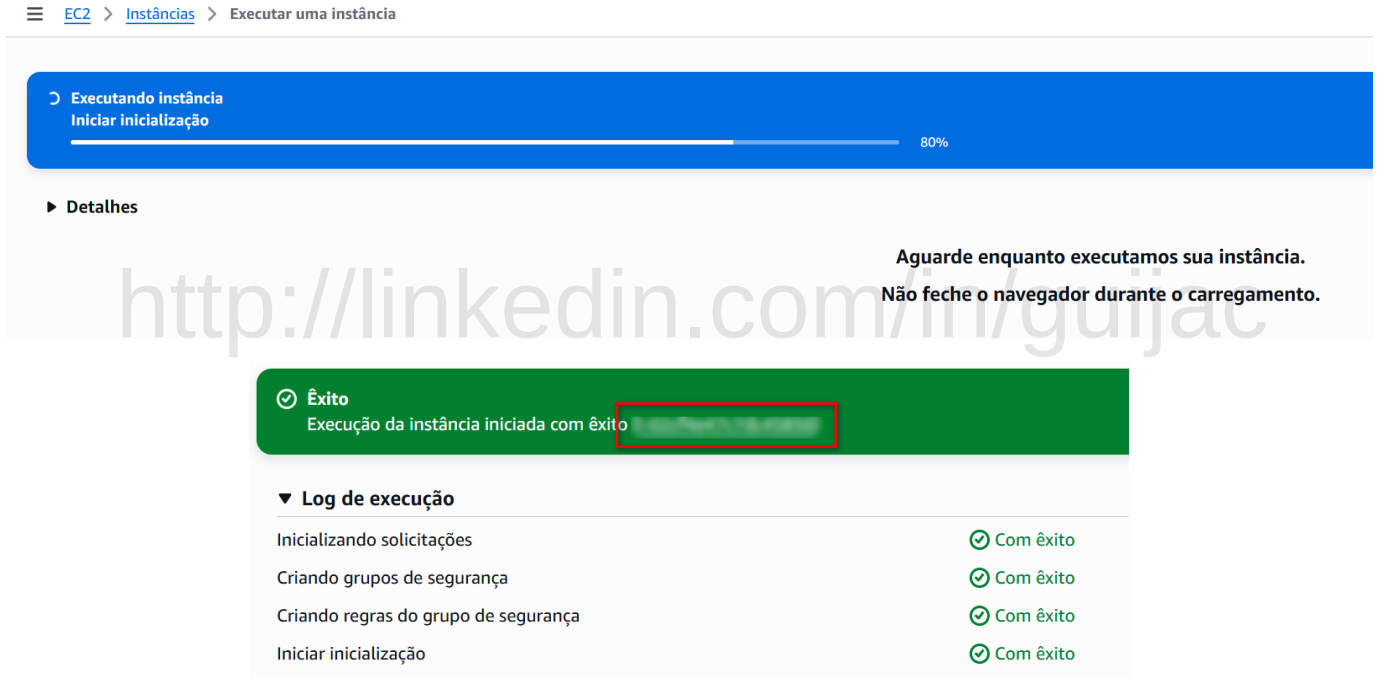
Desafio

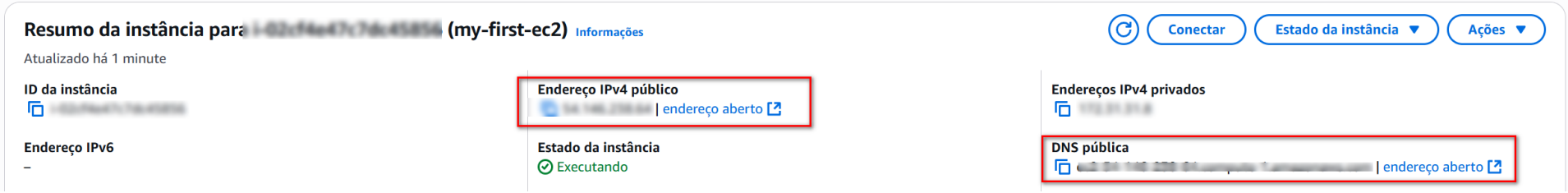
- Revise todas as configurações tomadas, **com atenção especial à chave PPK**, e clique em “Executar Instância”;
- Note que é possível visualizar o código (infraestrutura como código) que irá provisionar o seu EC2, explore este código.



Desafio

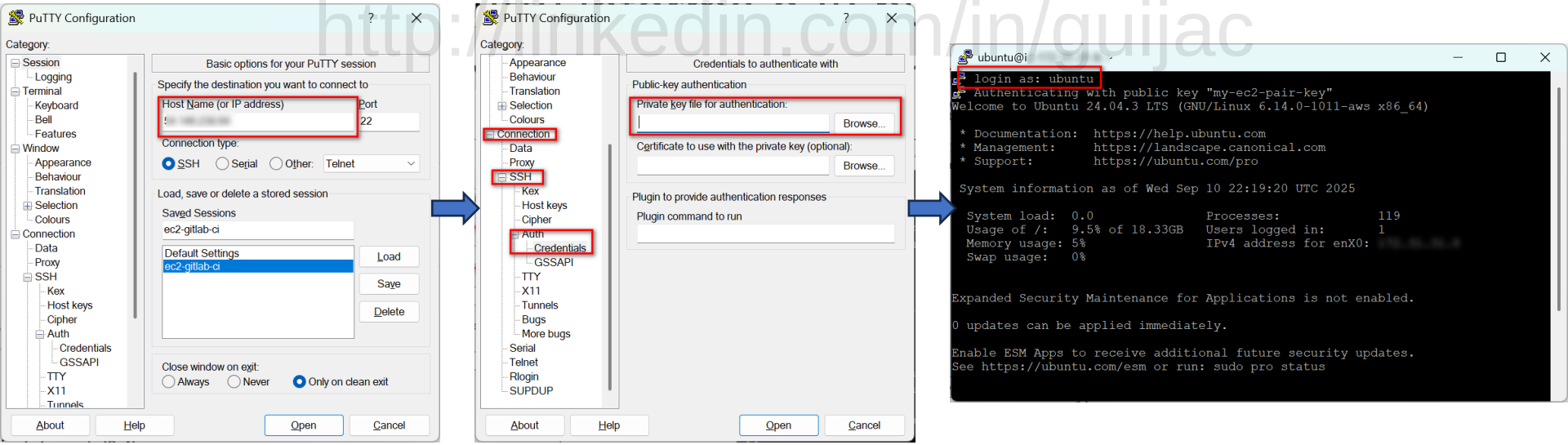
- Aguarde a conclusão e acesse os detalhes da instância recém criada, anotando o seu IP ou DNS público:





Desafio

- Também será possível conectar-se à instância através do AWS Console, mas para fins didáticos utilizaremos o PuTTY;
- Informe o IP ou DNS público e a chave privada para conexão;
- O usuário padrão será “ubuntu”:



Desafio

- Uma vez conectados à instância iremos realizar a instalação do Docker, pré-requisito para a instalação do Gitlab Runner;
- Execute os seguintes comandos:

```
sudo apt update
sudo apt install docker.io
sudo docker login
```


- Valide a instalação do Docker através do comando “docker --version”:

```
$ docker --version
Docker version 27.5.1, build 27.5.1-0ubuntu3~24.04.2
$
```

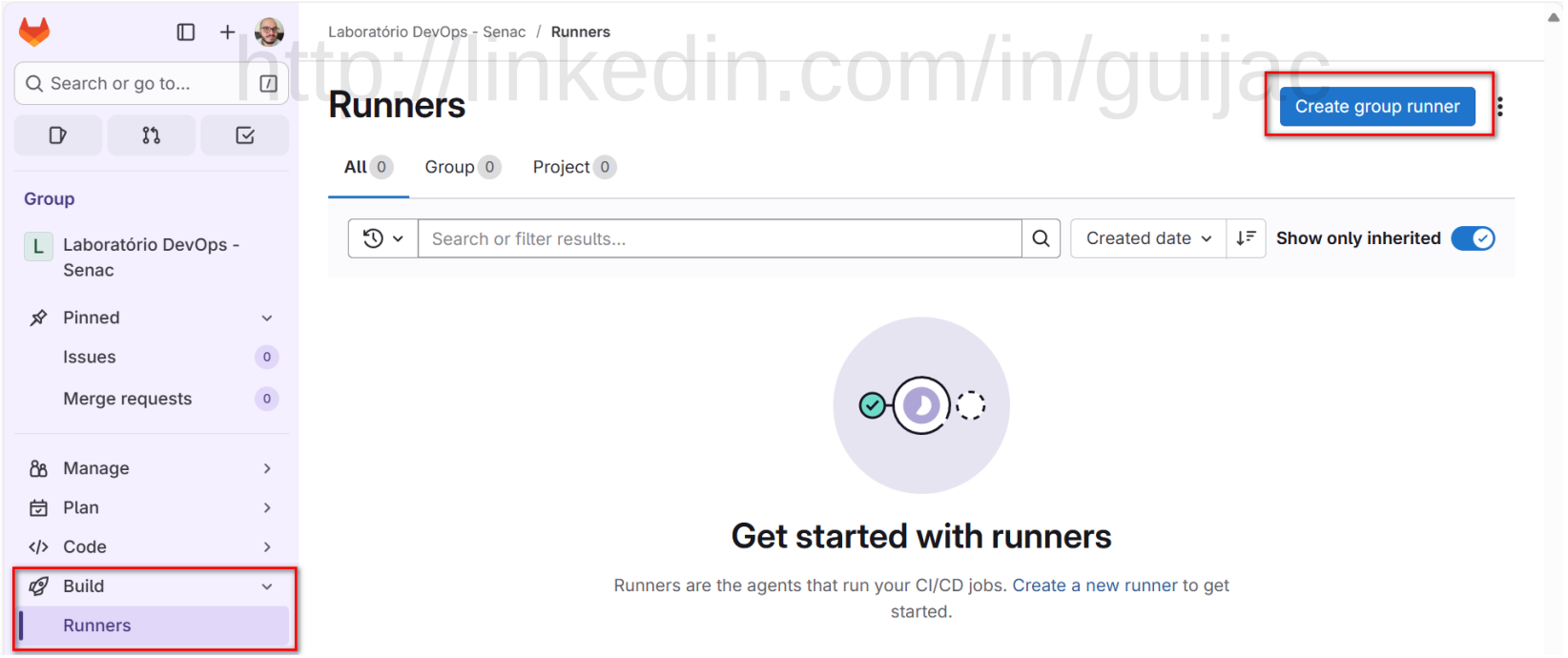
Desafio

- Para a instalação do Gitlab Runner será necessário acompanhar a documentação oficial: Install GitLab Runner using the official GitLab repositories;
- Leia com atenção, usando o tradutor, se necessário;
- Valide a instalação através do comando “gitlab-runner -v”;
- Em caso de dúvidas, peça meu apoio.


```
$ gitlab-runner -v
Version:      18.3.1
Git revision: 5a021a1c
Git branch:   18-3-stable
GO version:   go1.24.4 X:cacheprog
Built:        2025-09-04T15:24:16Z
OS/Arch:      linux/amd64
$
```

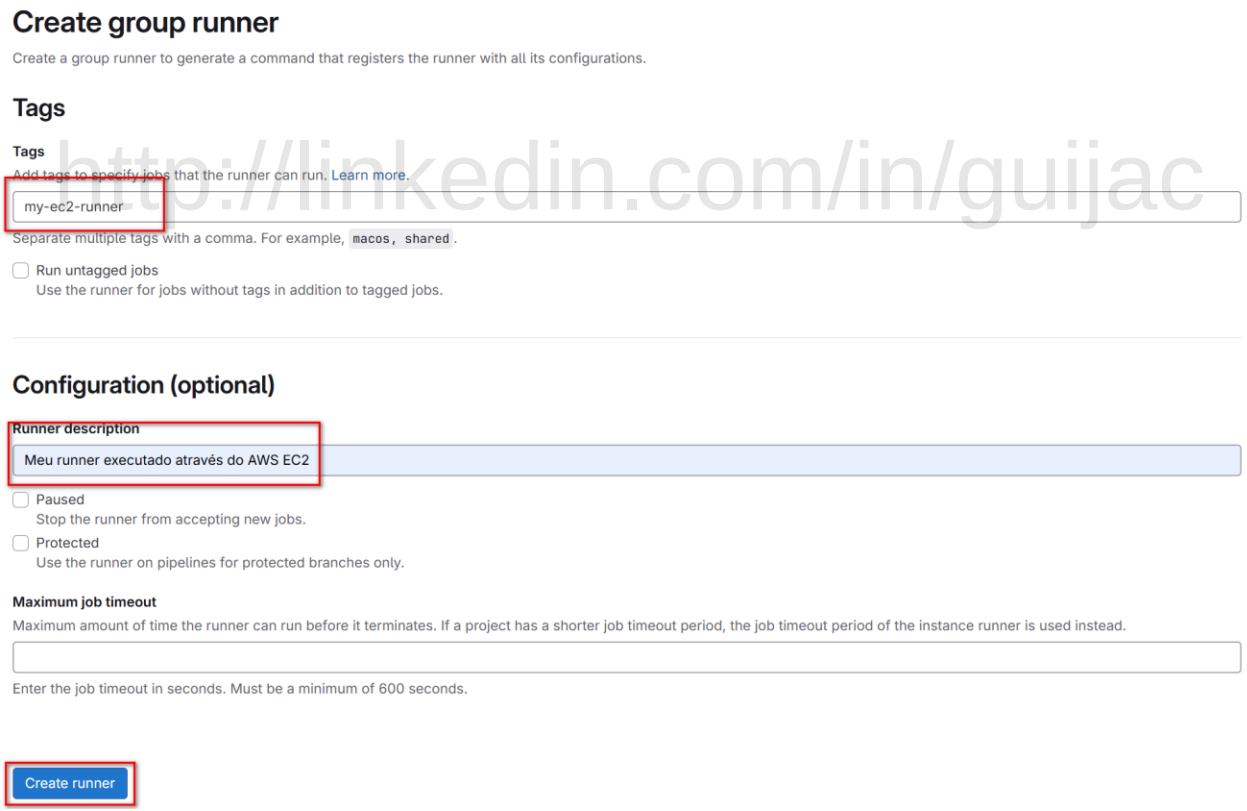
Desafio

- Concluída a instalação do *runner* no EC2, precisaremos registrá-lo, através de um token do Gitlab;
- Recomendo registrá-lo associado à um grupo, através do menu “Build”>”Runners”>”Create group runner”:



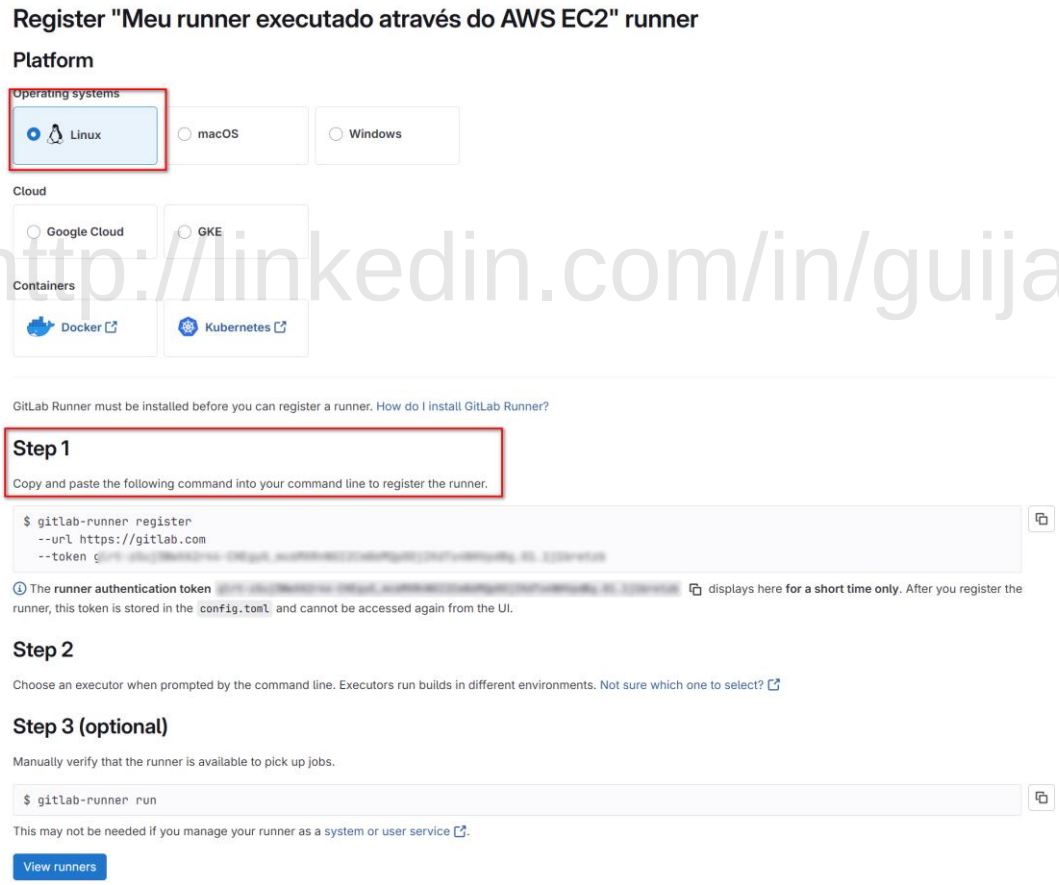
Desafio

- Dê um nome para a tag do *runner* (a tag é utilizada para o Gitlab escolher qual *runner* utilizar para um determinado *job*) e uma descrição para o mesmo, a seguir, clique em “Create runner”:



Desafio

- Selecione a plataforma do *runner* (Linux) e anote o token informado no “Step 1”, ele será informado no *bash* do seu EC2:



Desafio

- De volta à instância, leia a documentação Registering runners e execute o comando para registrar o runner, com o token copiado da etapa anterior;
- Informe os demais dados solicitados:

```
sudo gitlab-runner register  --url https://gitlab.com  --token <seu-token>

Enter the GitLab instance URL (for example, https://gitlab.com/):
[https://gitlab.com]: <informe https://gitlab.com/>

Enter a name for the runner. This is stored only in the local config.toml file:
[ip-172-31-31-8]: <dê um nome para seu runner>

Enter an executor: instance, shell, ssh, parallels, virtualbox, kubernetes, custom, docker, docker-windows,
docker+machine, docker-autoscaler: <docker>

Enter the default Docker image (for example, ruby:3.3): <alpine>
```

- Em caso de sucesso, a mensagem abaixo será exibida:

```
Runner registered successfully. Feel free to start it, but if it's running already the config should be
automatically reloaded!

Configuration (with the authentication token) was saved in "/home/ubuntu/.gitlab-runner/config.toml"
```

Desafio

- Concluída a configuração em seu EC2, clique em “View Runners”;
- Seu *runner* deverá aparecer normalmente, pronto para uso:

Desafio

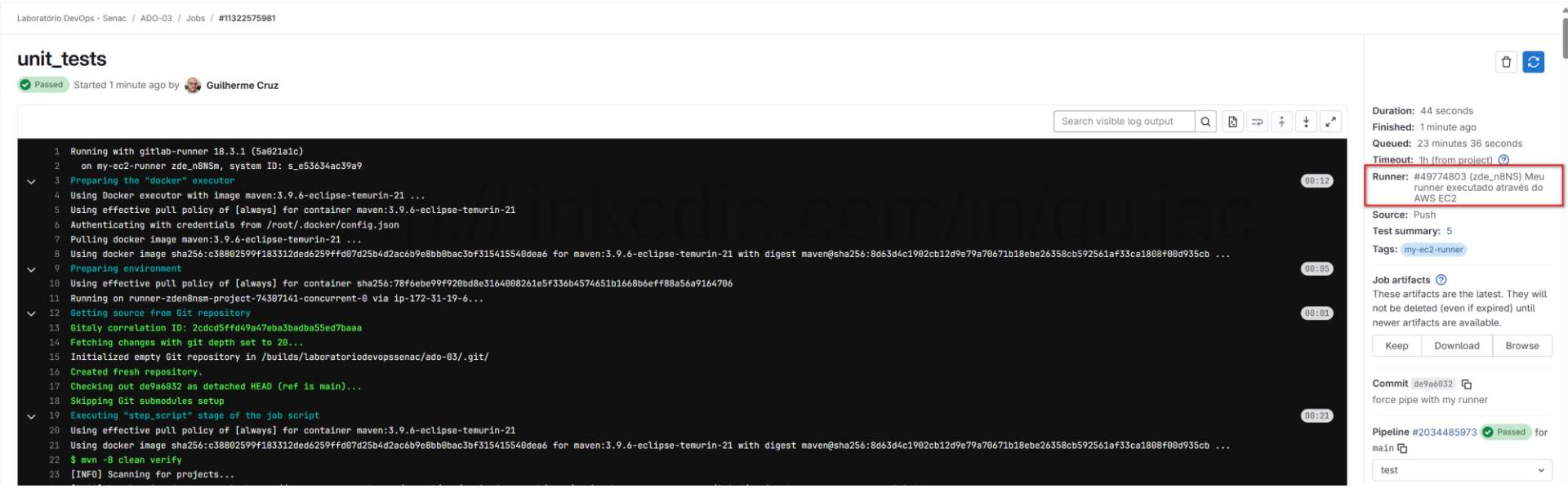
- Inclua a *tag* do *runner* em seu arquivo .gitlab-cy.yml, realize o *push* e acompanhe a execução do pipeline:

```yaml
stages:
  - test

# Job de Testes Unitários
unit_tests:
  stage: test
  tags:
    - my-ec2-runner
  image: maven:3.9.6-eclipse-temurin-21
  script:
    - mvn -B clean verify   # roda testes + gera relatório JaCoCo
  artifacts:
    when: always
    paths:
      - target/site/jacoco/
      - target/surefire-reports/
    reports:
      junit: target/surefire-reports/*.xml
```

Desafio

- Note que o job especificado irá informar que foi executado através do runner recém configurado:



- Concluída esta etapa, o desafio foi finalizado, parabéns! ☺

Referências Bibliográficas

FOWLER, Martin. **Continuous Integration**. Disponível em https://martinfowler.com/articles/continuousIntegration.html. Acesso em 08 mar 2025;

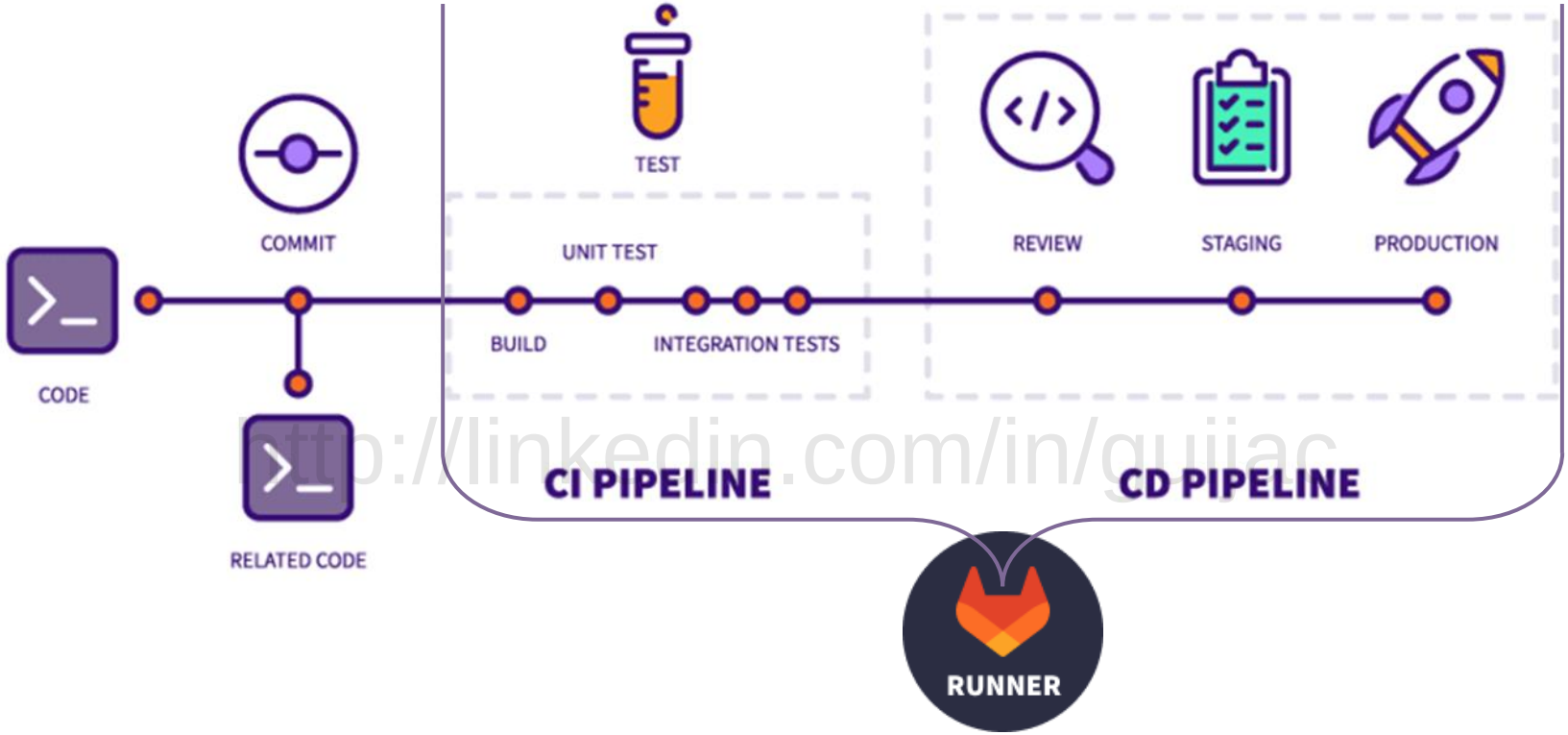
GITHUB. **Workflow syntax for GitHub Actions**. Disponível em https://docs.github.com/en/actions/using-workflows/workflow-syntax-for-github-actions. Acesso em 08 mar 2025;

GITLAB. **CI/CD concepts**. Disponível em https://docs.gitlab.com/ee/ci/introduction/. Acesso em 08 mar 2025;

HUMBLE, Jez; FARLEY, David. **Continuous delivery: reliable software releases through build, test, and deployment automation**. Pearson Education, 2010;

KIM, Gene et al. **The DevOps handbook: How to create world-class agility, reliability, & security in technology organizations**. IT Revolution, 2021.

Por hoje (agora sim!) é só!



Fonte: Adaptado de [Ultimate guide to CI/CD: Fundamentals to advanced implementation]

Prof. Esp. Guilherme Jorge Aragão da Cruz

guilherme.jacruz@sp.senac.br

linkedin.com/in/guijac

Licença

Prof. Esp. Guilherme Jorge Aragão da Cruz

guilherme.cruz@alumni.usp.br

linkedin.com/in/guijac