2020

LAB AUTOMAÇÃO DE BUILD E DEPLOY COM JENKINS E GITLAB

PROF MESTRE DANIEL LEMESZENSKI

FIAP | MBA

LAB DE CI E CD – GITLAB E JENKINS

SUMÁRIO

LAB DE CI E CD – GITLAB E JENKINS 1

1. Introdução 2
2. Docker Compose 2
3. Subindo o ambiente 2
4. Iniciando os serviços 3
5. criando chave rsa entre jenkins e host 3
6. Configurando Jenkins 4
7. Configurando GitLab 6
8. Criação do repositório no gitlab 7
9. Criando access token no gitlab 8
10. JENKINS – INSTALANDO GITLAB PLUGIN 9
11. JENKINS - Configurando a conexão COM Gitlab 11
12. Configurar Maven em Jenkins>Global tool Configuration 11
13. Criação do Job no Jenkins 12
14. Criando Webhook NO gitlab 16
15. Testando o pipeline (selecione test e push) 17

1. Introdução

Nesse lab vamos aprender a instalar e configurar o gitlab e o Jenkins. Além disso vamos criar um job de build no Jenkins disparado quando temos um evento em um repositorio do gitlab.

O deploy será um pacote jar do spring boot e que será construído com maven de publicado com scp.

2. Docker Compose

Para isso, usar o docker-compose.yml existente no projeto do git abaixo:

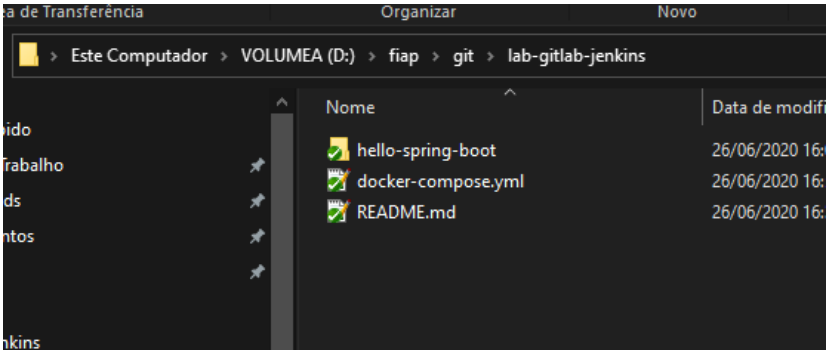
https://github.com/daniboy82/lab-gitlab-jenkins.git

```
version: '2'
services:
 jenkins:
  image: jenkins/jenkins
  container_name: jenkins
  hostname: jenkins
  network_mode: bridge
  ports:
   - "85:8080"
    #volumes:
    #  - ~/jenkins_home:/var/jenkins_home
 gitlab:
      image: gitlab/gitlab-ce
      container_name: gitlab
      hostname: gitlab
      restart: always
      network_mode: bridge
      ports:
        - "80:80"
      # volumes:
      #   - ~/gitlab/config:/etc/gitlab
      #   - ~/gitlab/logs:/var/log/gitlab
      #   - ~/gitlab/data:/var/opt/gitlab
```

3. Subindo o ambiente

Crie um fork do projeto:

https://github.com/daniboy82/lab-gitlab-jenkins



Clone o branch criado através do fork

entre no diretório lab-gitlab-jenkins

4. INICIANDO OS SERVIÇOS

No diretório em que você criou o arquivo docker-compose.yml, execute o comando:

```
docker-compose up -d
```

5. CRIANDO CHAVE RSA ENTRE JENKINS E HOST

A - Entrar no container Jenkins e criar chave rsa:

```
docker exec -it jenkins bash
```

(criar chave sem pass frase, enter, enter, enter...)

```
ssh-keygen -t rsa
```

B - Criando usuário Jenkins na máquina EC2

```
useradd -m jenkins
passwd Jenkins
ssh-keygen -t rsa
```

(enter, enter, enter….)

```
sudo docker cp jenkins:/var/jenkins_home/.ssh/id_rsa.pub /home/jenkins/
cat /home/jenkins/id_rsa.pub >> /home/jenkins/.ssh/authorized_keys
chmod 644 /home/jenkins/.ssh/authorized_keys
```

C- testar chave

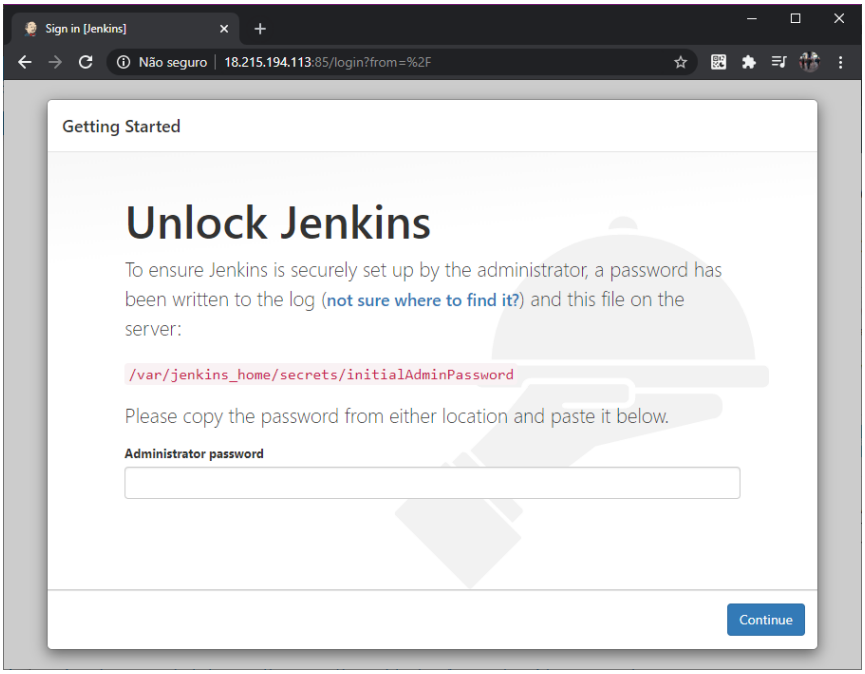
Entrar no container jenkins e executar

ssh jenkins@ip-elastico

Se conectar sem pedir senha significa que funcionou!

6. CONFIGURANDO JENKINS

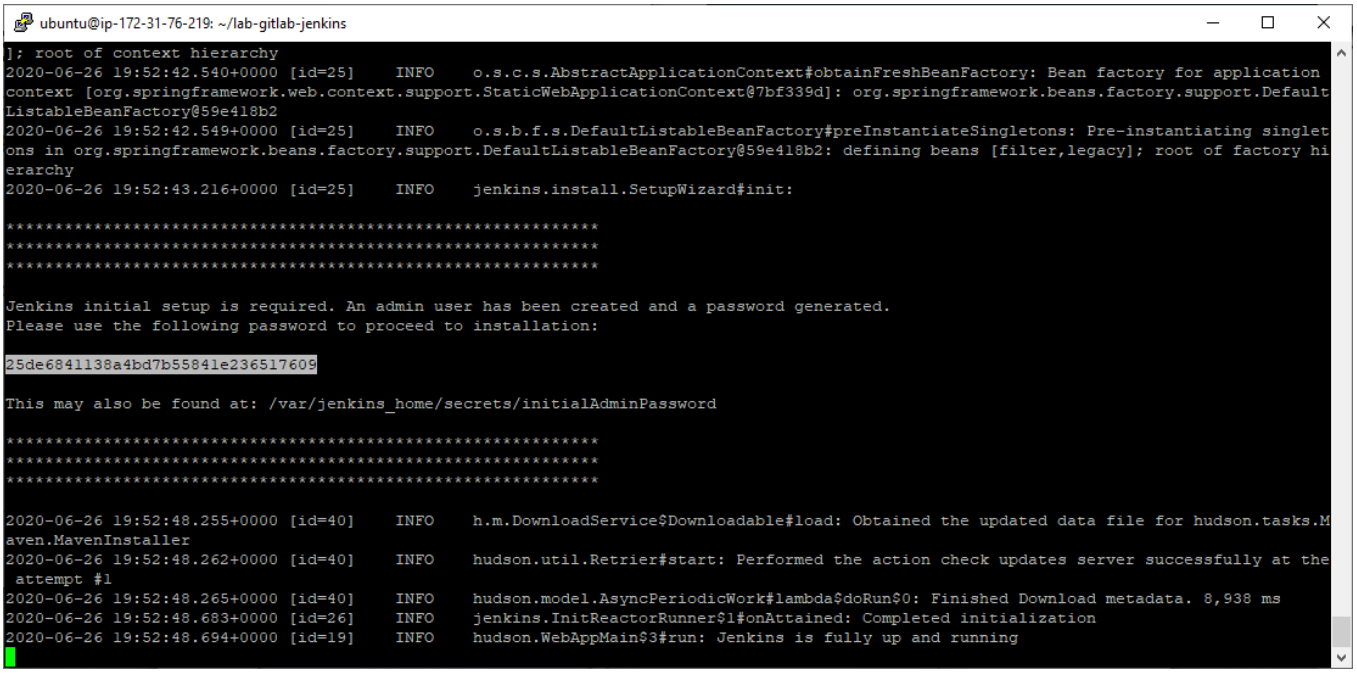
Após a inicialização dos serviços com o docker-compose, vamos fazer a configuração inicial do Jenkins. Acesse a url http://ip-externo:85/ e você será direcionado para a página inicial da ferramenta.

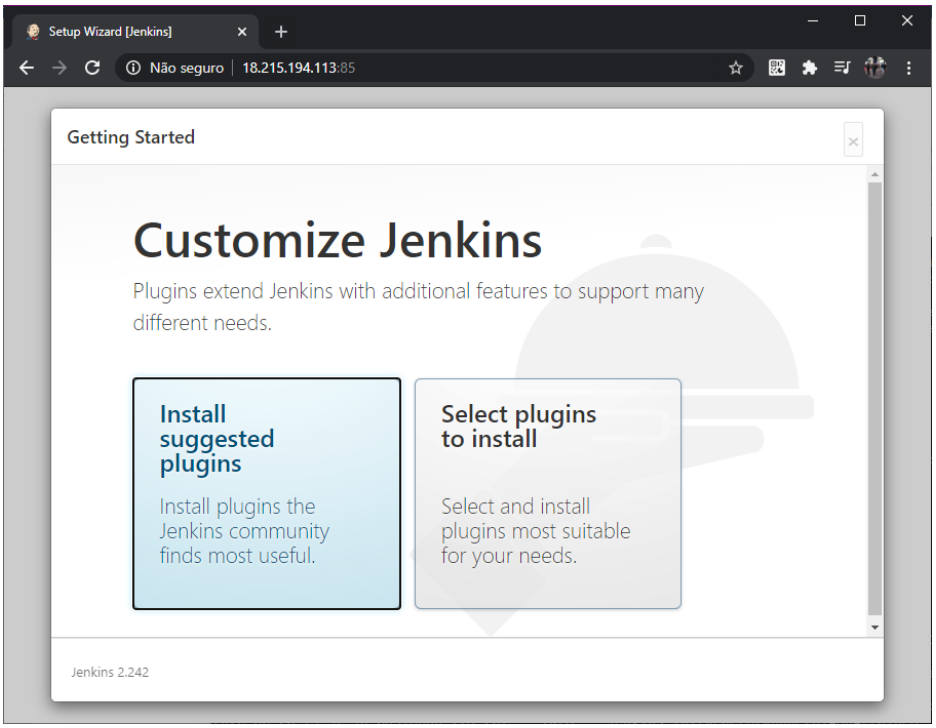


Para configurá-lo, vamos inserir a chave que ele gerou no momento da instalação. O jeito mais simples é digitar no terminal:

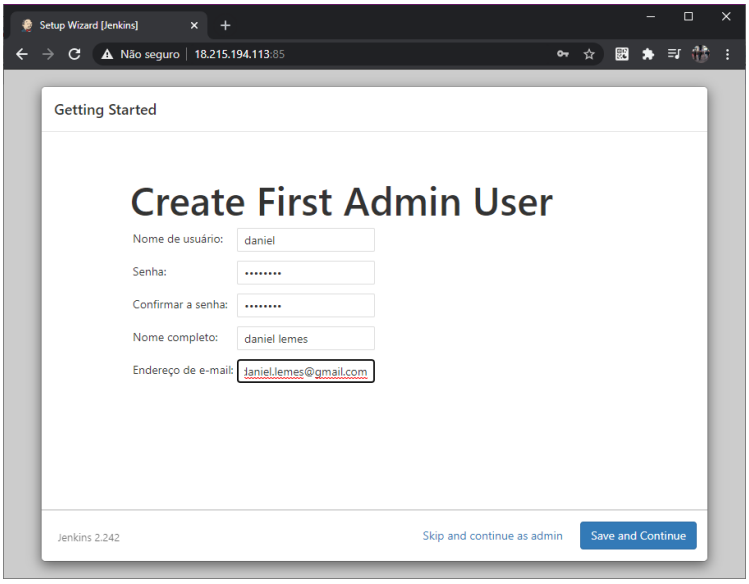
docker logs -f Jenkins

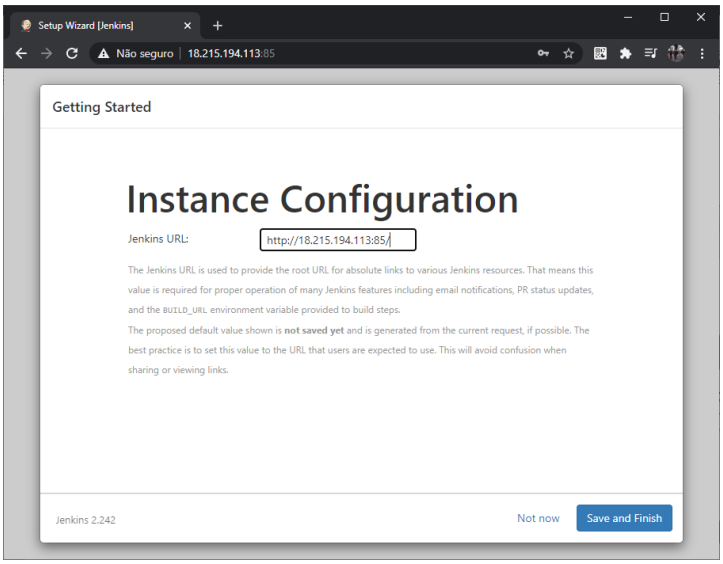
Ele vai exibir assim:





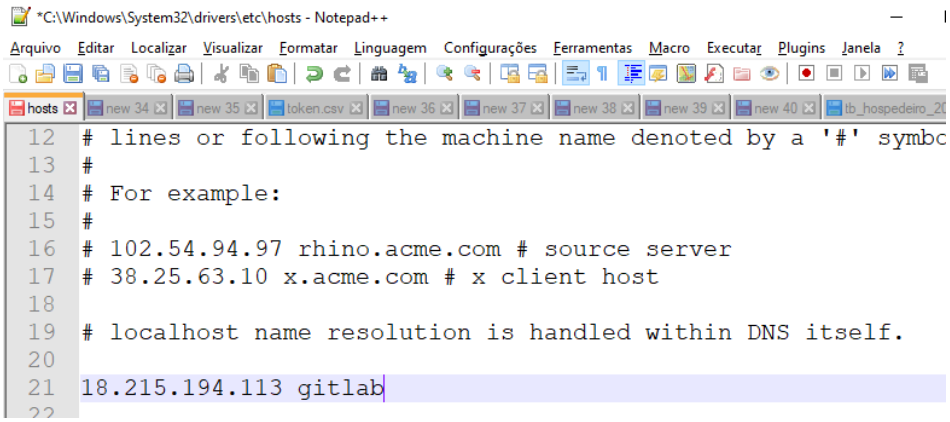
Pronto, o Jenkins já está pronto para ser utilizado!



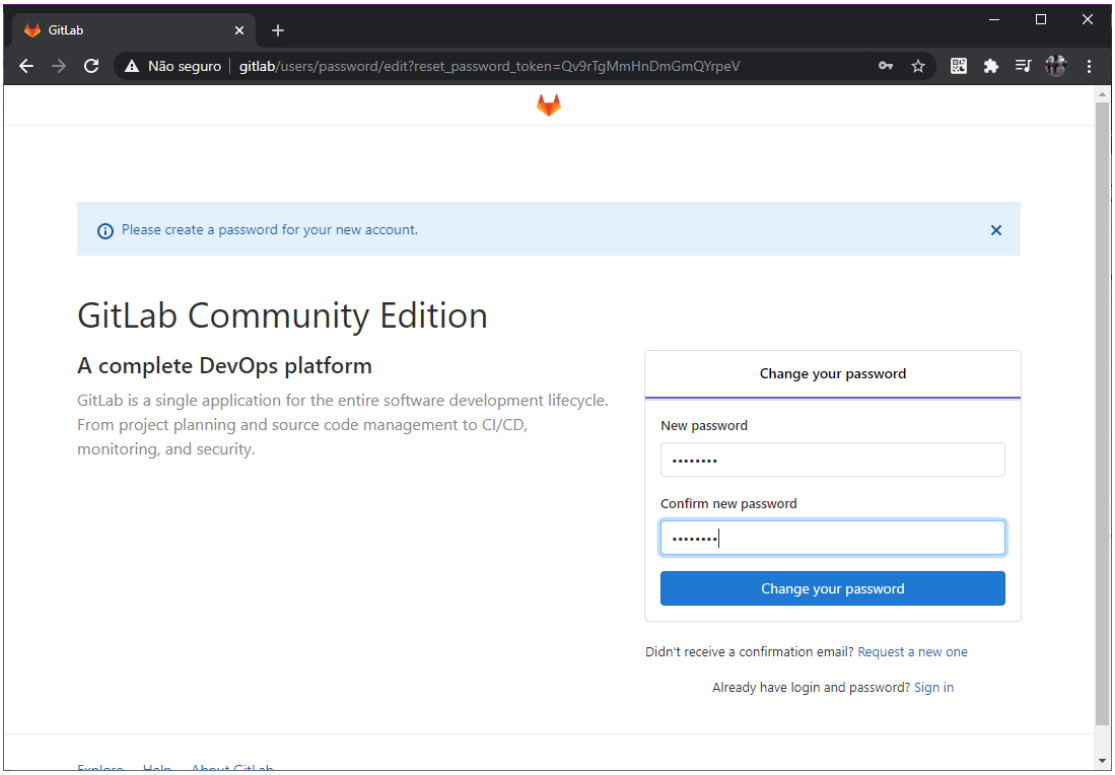


7. CONFIGURANDO GITLAB

Configurar o etc/hosts com o ip-externo da máquina ec2:



Com o GitLab, o processo é bem mais simples. É só acessar a página inicial dele http://gitlab/ e colocar uma senha com no mínimo oito caracteres.

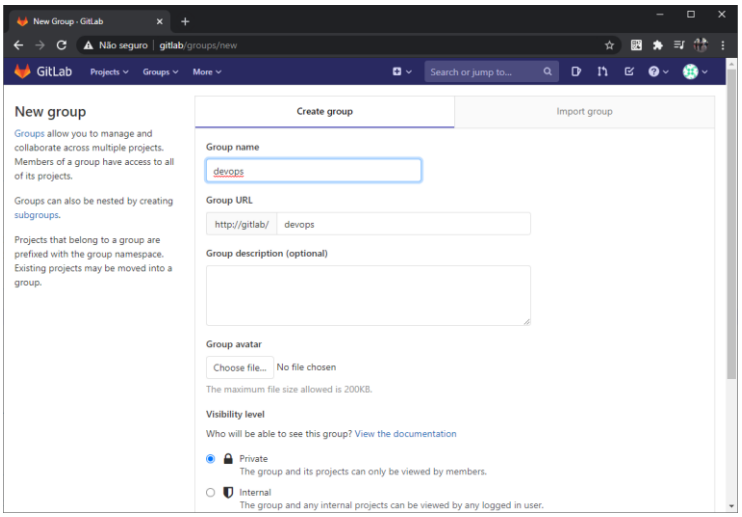


Agora estamos com o ambiente pronto e podemos começar a configuração da integração. Segura, que o filho é seu!

8. CRIAÇÃO DO REPOSITÓRIO NO GITLAB

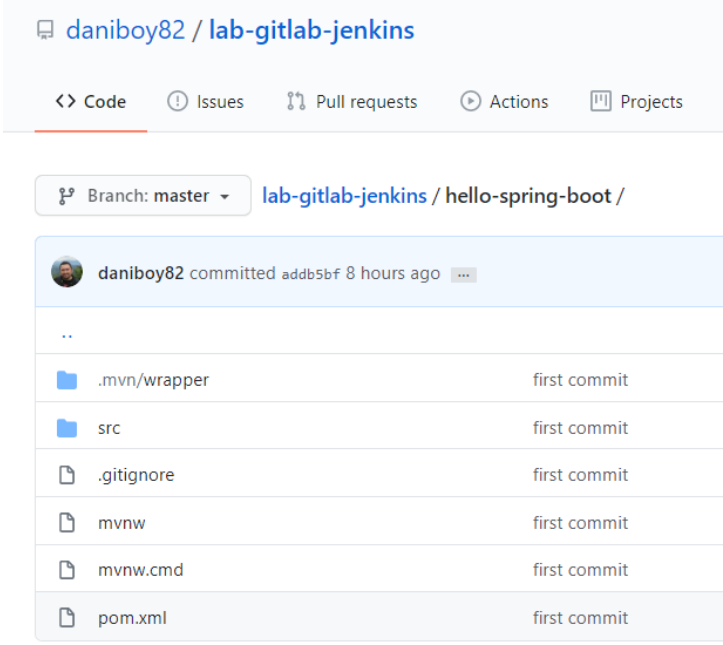
Vamos criar um repositório no Gitlab para armazenar nosso projeto.

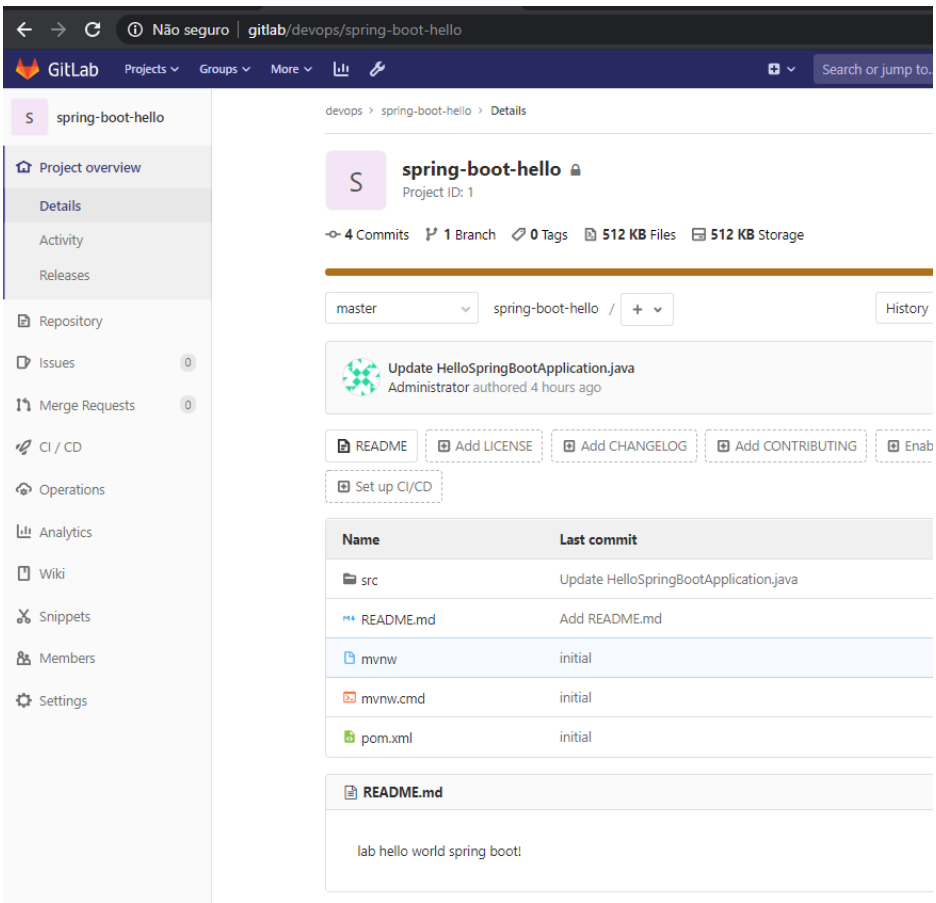
Crie o grupo devops:



a - Crie o projeto spring-boot-hello no gitlab
b - Copie os arquivos do github existentes
em https://github.com/daniboy82/lab-gitlab-jenkins/tree/master/hello-spring-boot nesse projeto;

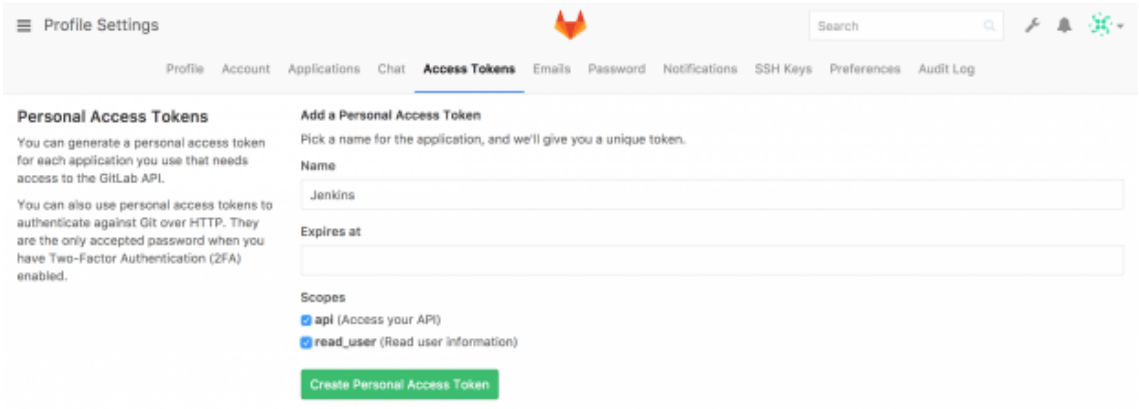
c-realize o commit e push desses fontes:



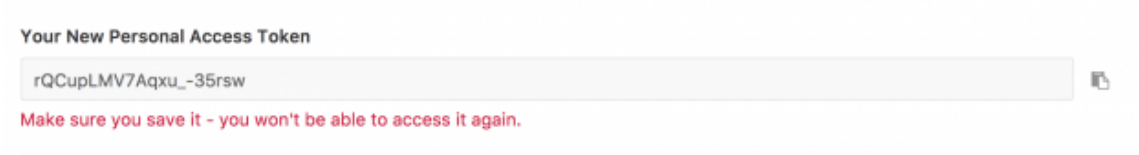


9. CRIANDO ACCESS TOKEN NO GITLAB

Clique no círculo com sua imagem de perfil, que fica no canto superior direito, e vá em *Profile Settings*.
Clique na aba *Access Tokens*. Digite um nome fácil de identificar para seu token e clique em *Create Personal Access Token*.



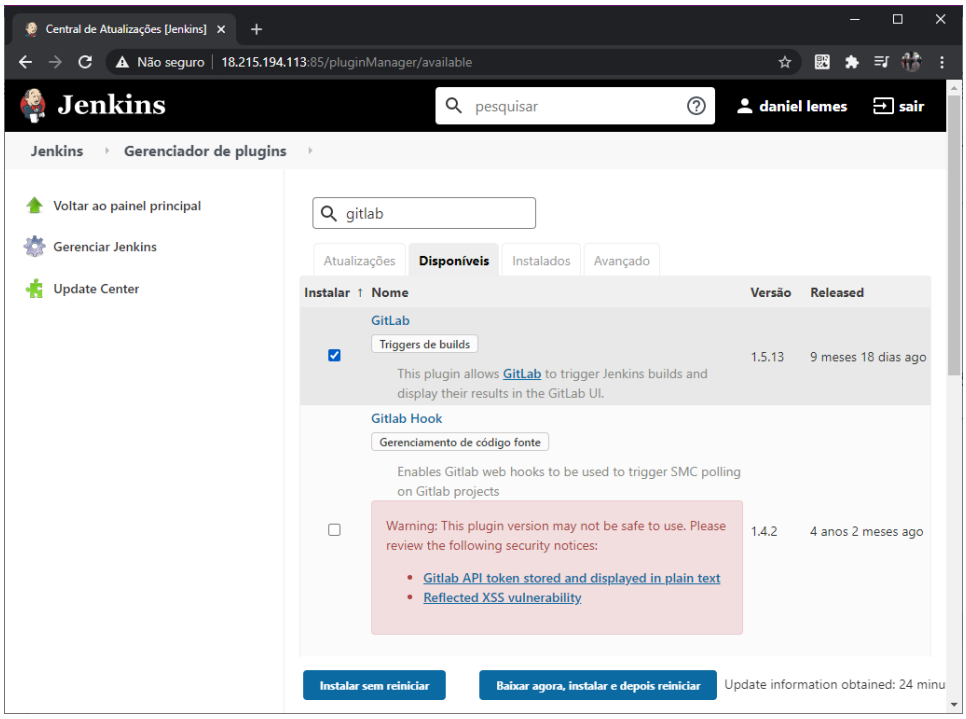
Após a criação, copie o hash exibido na tela, porque vamos usá-lo no próximo passo.



O token desaparece após atualizar a página, então, guarde em um lugar seguro.

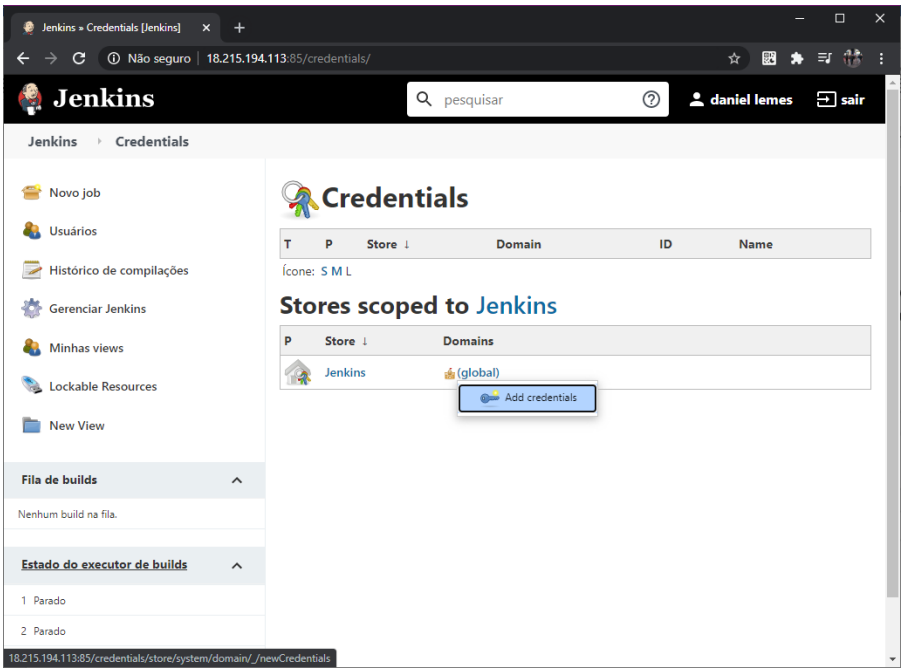
10.JENKINS – INSTALANDO GITLAB PLUGIN

Vamos precisar do GitLab Plugin para nos conectarmos ao repositório do projeto (lembre-se: são só dois arquivos, mas vamos manter a positividade). Para instalar, vá até [Gerenciar Jenkins]->[Gerenciar Plugins] e clique na aba *Disponíveis*.

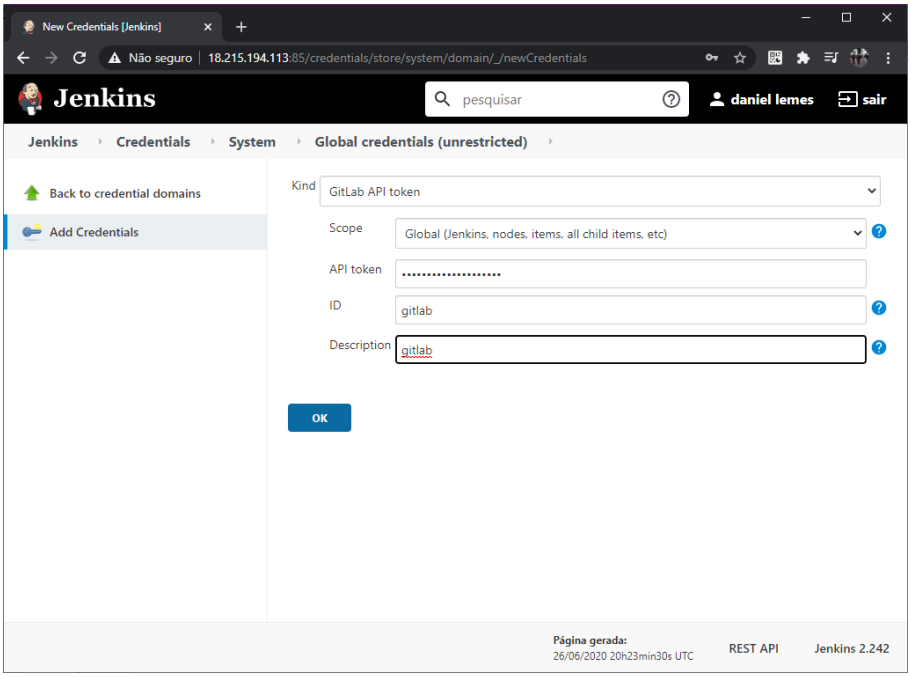


Com o plugin instalado, vamos adicionar uma conexão com o GitLab, seguir alguns passos simples.

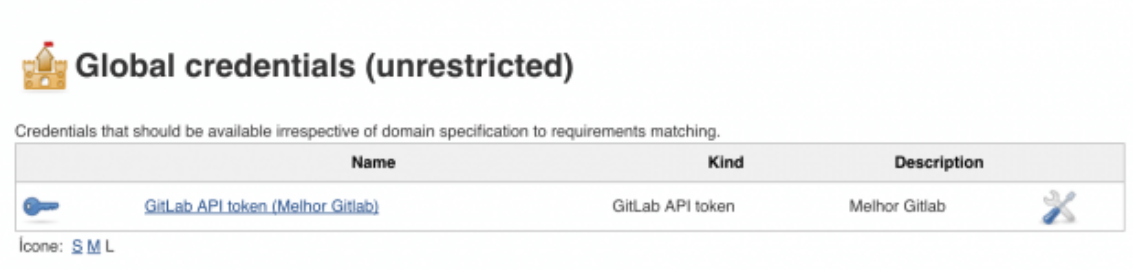
Na tela inicial do Jenkins, vá até Credentials. Clique na seta do lado de global e selecione *Add Credentials*.



Na janela a seguir, selecione *GitLab API Token* e coloque o hash gerado no GitLab no campo API token.



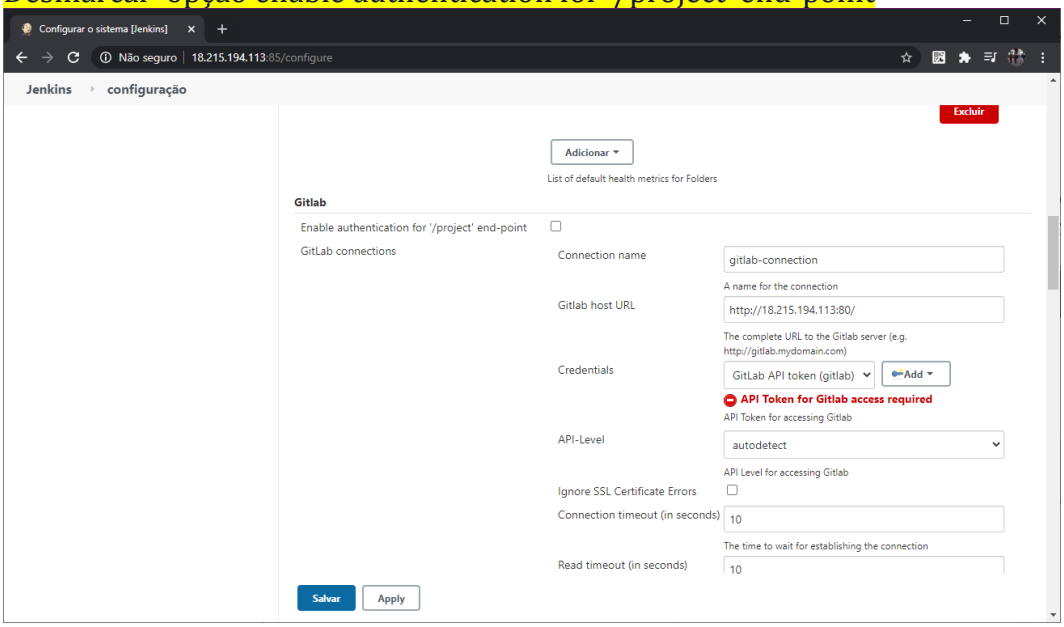
Após clicar em OK, você verá sua credencial criada.



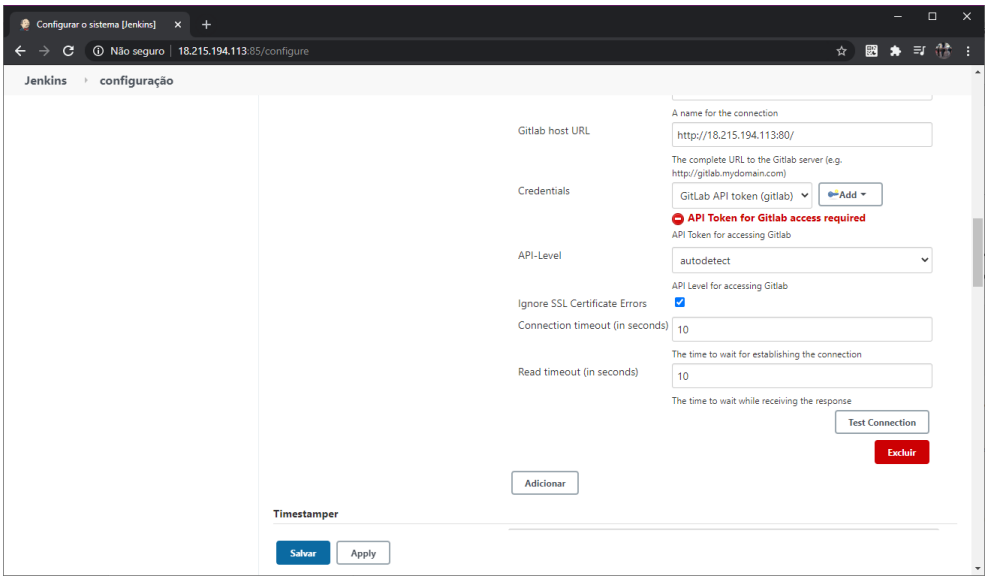
11.JENKINS - CONFIGURANDO A CONEXÃO COM GITLAB

Com a nossa credencial criada, vamos em [Gerenciar Jenkins]->[Configurar o sistema]. Desça até a Gitlab e preencha as informações conforme a imagem abaixo. Ao terminar, clique em *Test Connection*. Se tudo estiver certo, clique em *Salvar*.

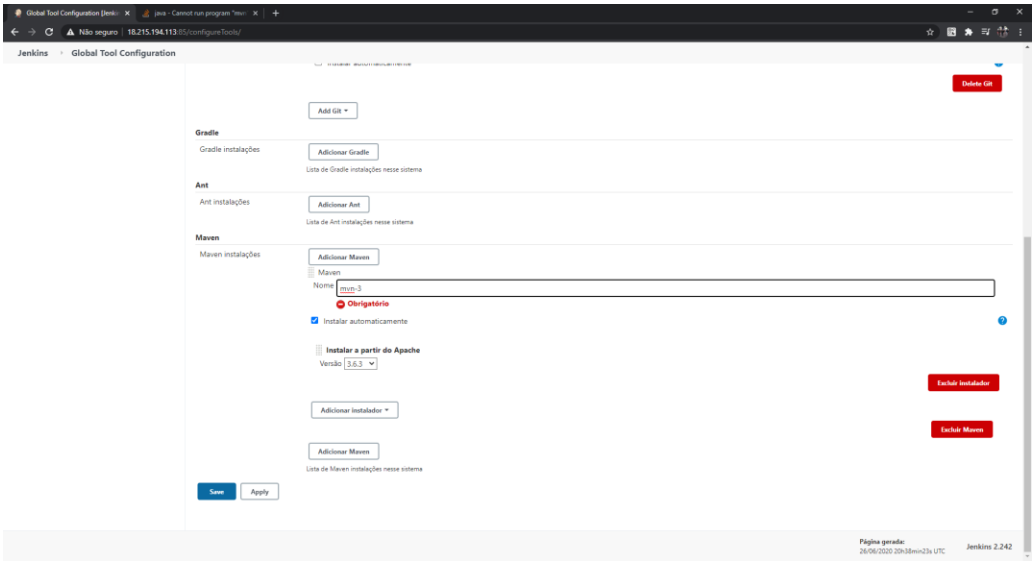
Desmarcar opção enable authentication for '/project' end-point



Marcar ignore ssl certifcate erros:

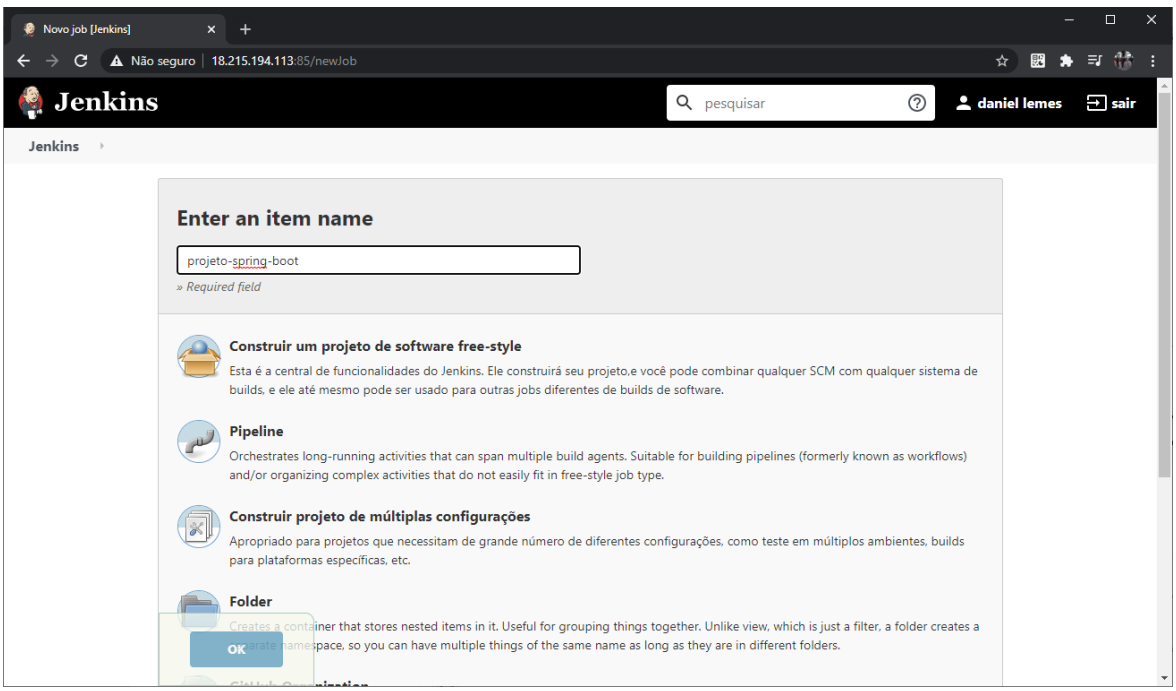


12. CONFIGURAR MAVEN EM JENKINS>GLOBAL TOOL CONFIGURATION



13.CRIAÇÃO DO JOB NO JENKINS

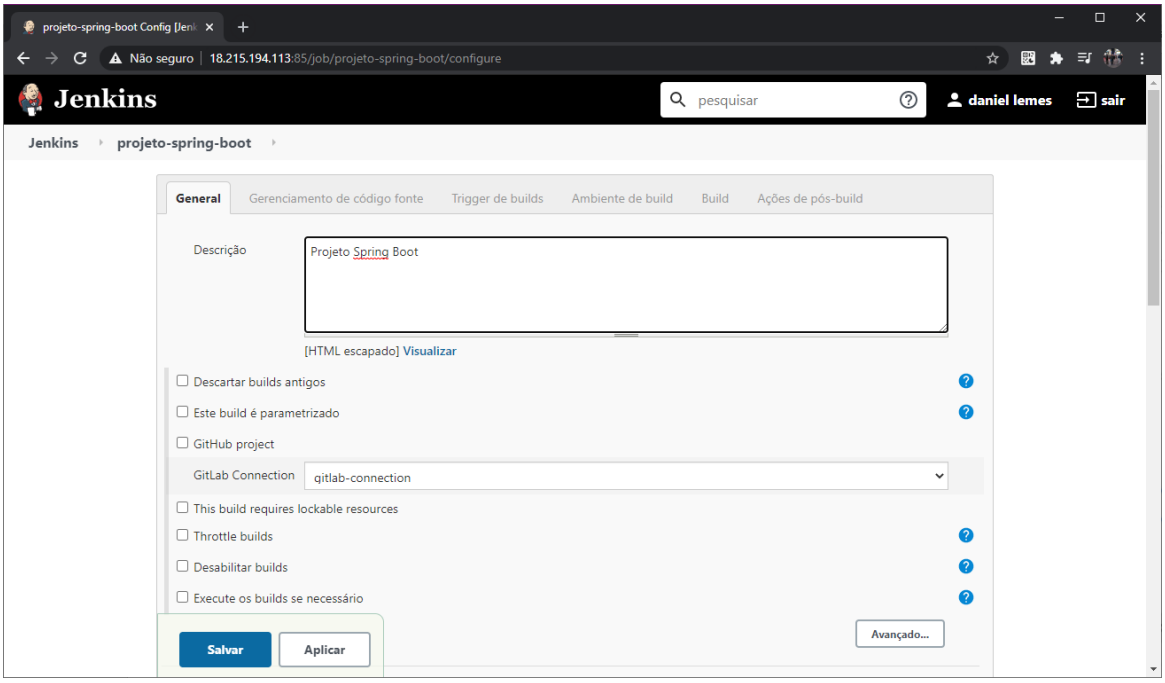
No canto esquerdo, clique em *Novo Job*, escolha *Projeto Freestyle*, digite um nome para o Job e clique em *OK*.



Com o Job criado, vamos às configurações.

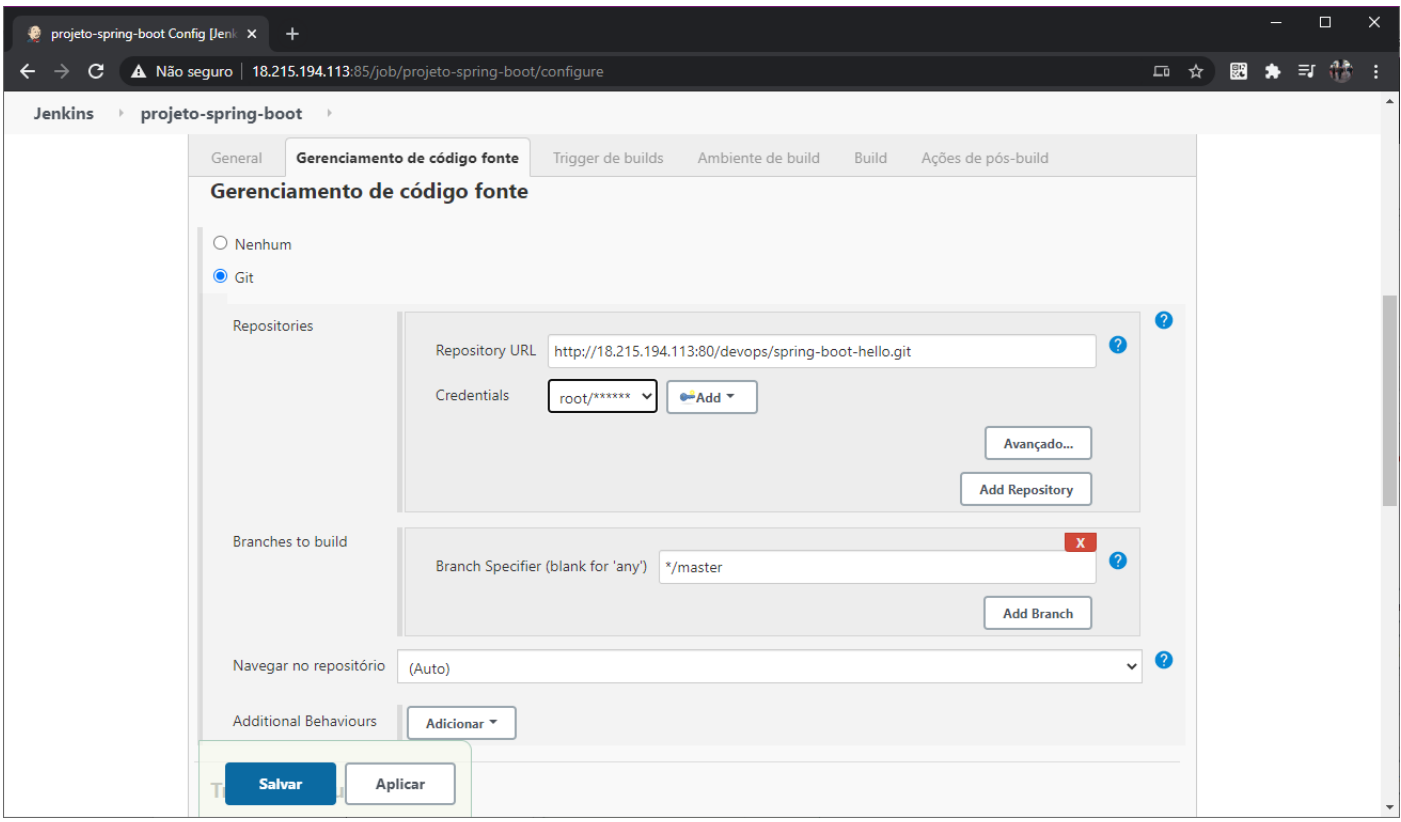
General

Adicione a conexão que criamos no campo *GitLab Connection*.

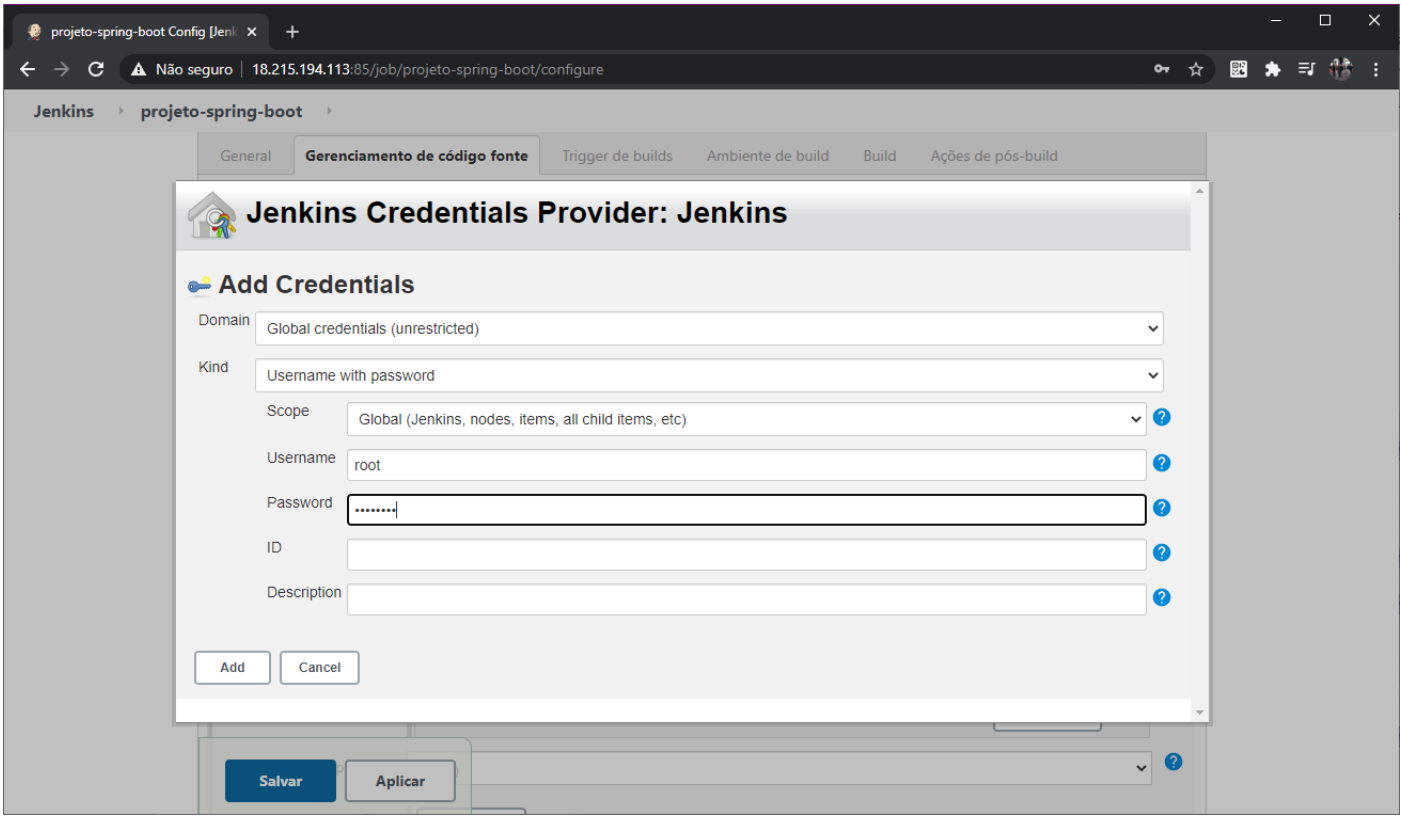


Gerenciamento de código fonte

Selecione a opção Git e adicione as informações do repositório criado no GitLab.



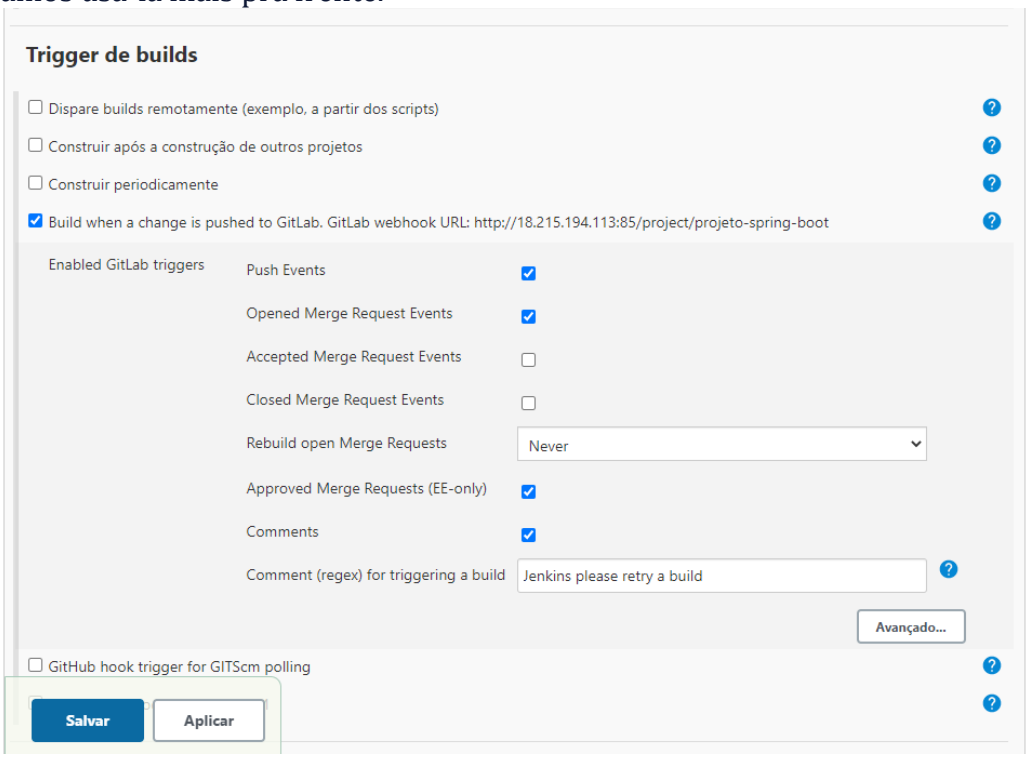
Para adicionar as credencias válidas para o repositório, clique em [Add]->[Jenkins] e adicione um usuário e senha válidos para conexão.



Trigger de Builds

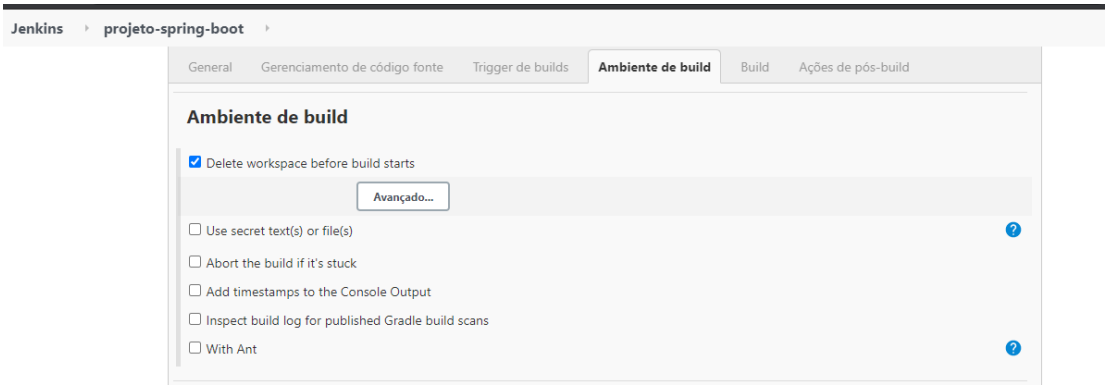
Esse é o primeiro passo para nossa integração entre os dois serviços. Estamos quase lá!

Marque a opção *Build when a change is pushed to GitLab* e selecione quais ações executadas no GitLab vão acionar o Job no Jenkins. Copie a URL logo após a frase *GitLab CI Service URL*, pois vamos usá-la mais pra frente.



Marque delete workspace before build starts:

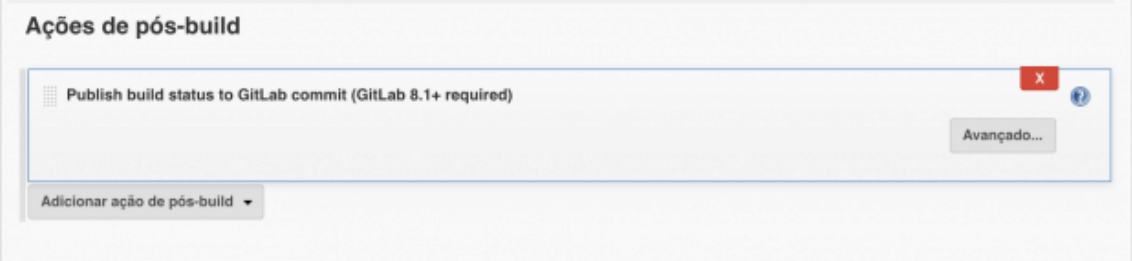
Ambiente de Build



Configurar build:

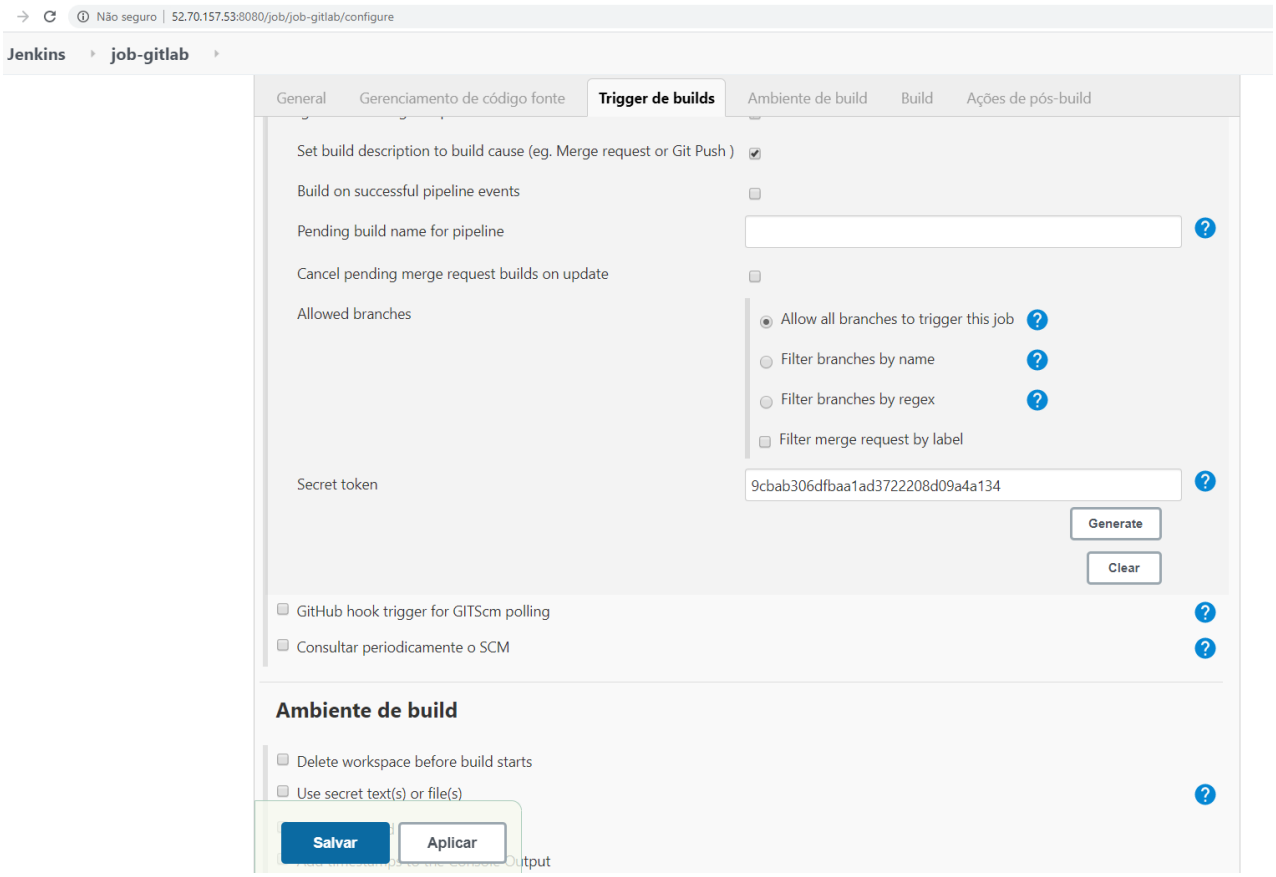
Ações de pós-build

Selecione a opção *Publish build status to GitLab commit*. Agora o Jenkins colocará o feedback de cada build nos commits/merges que acionaram o Job.



Clique em Salvar e o Job está finalizado.

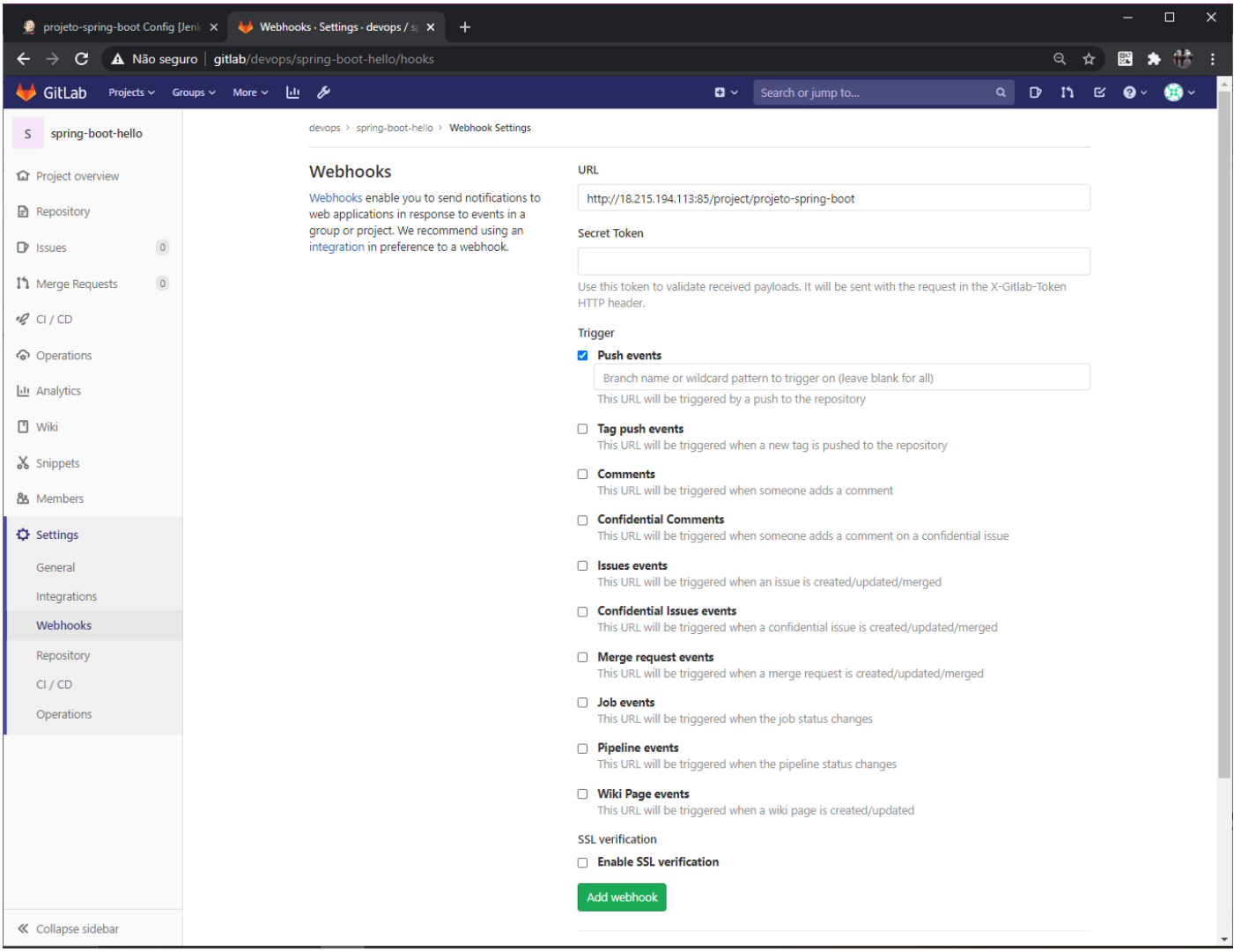
Depois de salvar o job, entrar novamente nas configurações do job, em trigger de builds e clicar em avançado, generate token (salve a hash gerada no notepad ou área de transferência) e clique em salvar.



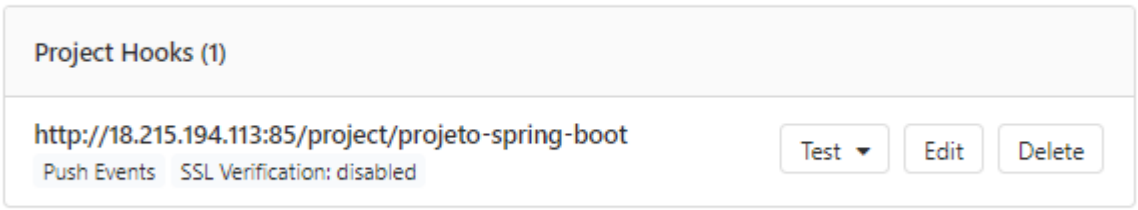
14.CRIANDO WEBHOOK NO GITLAB

Como último passo, vamos adicionar o Webhook em nosso repositório no GitLab. Ele vai fornecer as informações para o Jenkins quando houver alguma alteração ou ação no repositório no qual ele foi configurado.

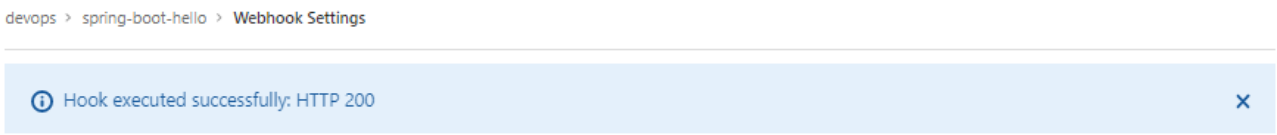
Na página inicial do repositório, clique na engrenagem no canto superior direito e selecione *Webhooks.*



15.TESTANDO O PIPELINE (SELECIONE TEST E PUSH)



Deve aparecer:



Após tantos passos e configurações, vamos ver nosso projeto rodando. Abra um merge request ou faça um commit/push para o repositório criado do gitlab e aguarde o Job ser iniciado. Ao finalizar, ele vai adicionar ao GitLab o feedback do build realizado com as alterações que foram feitas no código. Com isso seu projeto ganha muito mais agilidade e confiabilidade, com testes e feedbacks rápidos e automatizados.

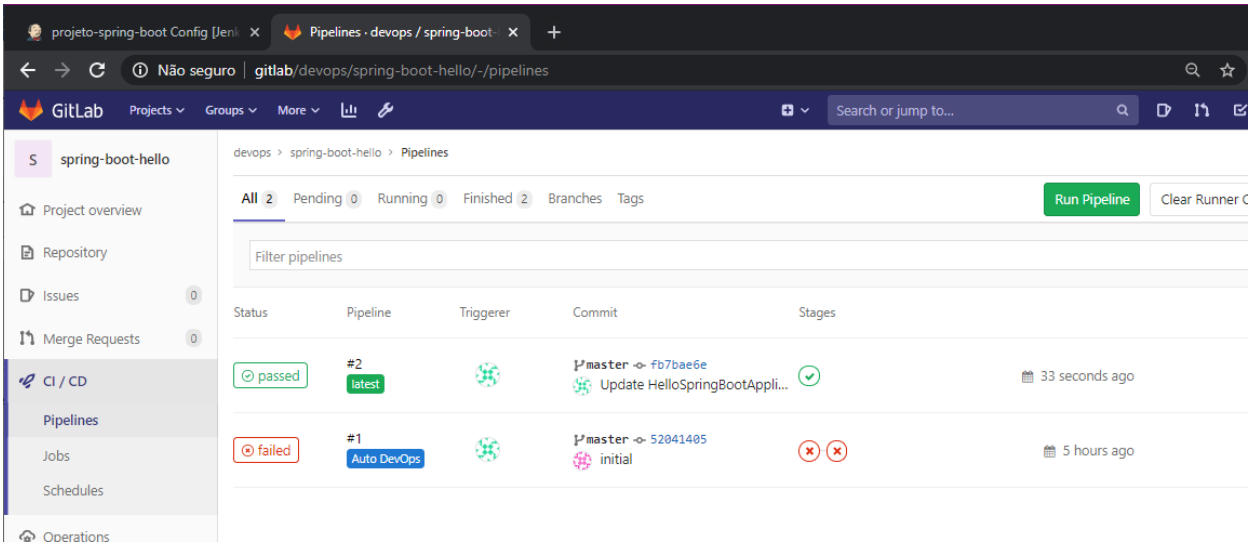
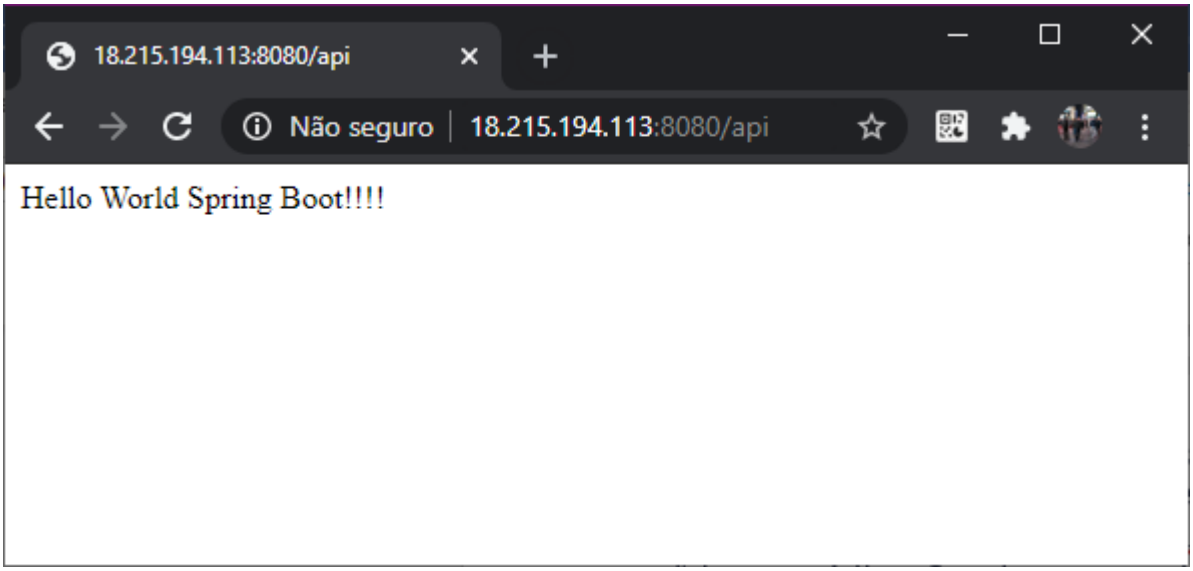


Figura 1 menu pipeline do repositório

Valide que o deploy foi realizado com sucesso acessando http://ip-ec2:8080



Referências:

https://imasters.com.br/devsecops/gitlab-jenkins-uma-integracao-poderosa