

## **Processo de configuração do ambiente para instalação do Jenkins com Docker.**

O Jenkins e o Docker se dão muito bem juntos. Containers são ferramentas poderosas para construir software em diferentes ambientes e o Jenkins em si é fácil de rodar em um container. Mas como você pode gerenciar agents e controllers juntos no Docker?

### **O que é o Docker Compose?**

As vezes você precisa rodar mais do que um container, mas você não quer gerenciar vários scripts no shell, especialmente quando os containers precisam se comunicar uns com os outros.

O Docker Compose é uma ferramenta para definir como rodar múltiplos container em um arquivo de configuração simples, além de começar, parar e reiniciar eles com um único comando.

---

### **O que é o Jenkins?**

O Jenkins é uma ferramenta de automação. Enquanto você pode usá-lo para automatizar qualquer tipo tarefa, ele é geralmente mais associado à criação do código fonte e implantação dos resultados. O Jenkins para muitos é sinônimo de integração continua e entrega continua (CI / CD)

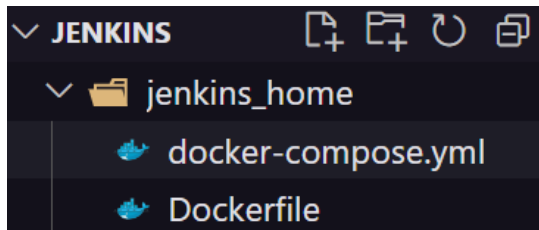
---

## **Configuração do Jenkins com o Docker Compose**

Obs. Certifique-se de ter Docker instalado na sua máquina.

Agora que já sabemos o que são o Docker Compose e o Jenkins chegou a hora de configurar o ambiente.

Primeiramente vamos criar um diretório Jenkins vazio onde será guardado nossos dois arquivos “Dockerfile” e “docker-compose.yml”. Ainda no diretório Jenkins criemos um diretório “Jenkins\_home” que será armazenada nossa aplicação Jenkins, conforme o exemplo a seguir:



---

Primeiros vamos criar nossa imagem personalizada do Jenkins no arquivo Dockerfile

```
1 FROM jenkins/jenkins:latest-jdk11
2 USER root
3 RUN apt-get update && apt-get install -y lsb-release
4 RUN curl -fsSL /usr/share/keyrings/docker-archive-keyring.asc \
5     https://download.docker.com/linux/debian/gpg
6 RUN echo "deb [arch=$(dpkg --print-architecture) \
7     signed-by=/usr/share/keyrings/docker-archive-keyring.asc] \
8     https://download.docker.com/linux/debian \
9     $(lsb_release -cs) stable" > /etc/apt/sources.list.d/docker.list
10 RUN apt-get update && apt-get install -y docker-ce-cli
11 RUN groupadd docker && usermod -aG docker jenkins
12 RUN usermod -aG root jenkins
13 USER jenkins
14 RUN jenkins-plugin-cli --plugins "blueocean:1.25.8 docker-workflow:521.
    v1a_a_dd2073b_2e"
```

Linha 1: Aqui estamos usando a imagem oficial do Jenkins que inclui o JDK 11 mais recente

Linha 2: Altera para o usuário root para poder executar os próximos comandos como super usuário dentro do contêiner.

Linha 14: Instala os plugins do Jenkins especificados.

Após a execução dessas etapas, teremos uma imagem do Jenkins personalizada com suporte ao Docker, ela será usada para a criação de um contêiner Jenkins utilizando o nosso próximo arquivo "docker-compose.yml".

---

A seguir a configuração do nosso docker-compose.yml

```
1  version: '3'
2  services:
3    jenkins:
4      container_name: jenkins
5      build:
6        context: .
7      ports:
8        - 8080:8080
9      volumes:
10       - /home/hacjesse/Documents/jenkins/jenkins_home:/var/jenkins_home
11       - /var/run/docker.sock:/var/run/docker.sock
```

Linha 1: Especifica a versão da sintaxe do Docker compose utilizada no arquivo

Linha 2: Define os serviços que serão executados no ambiente Docker.

Linha 3: Nome do serviço do Jenkins

Linha 5: Está especificando como construir o contêiner do Jenkins

Linha 6: Irá procurar o arquivo Dockerfile no diretório atual para a construção da imagem

Linha 8: Mapeia a porta 8080 do host para a porta 8080 do contêiner.

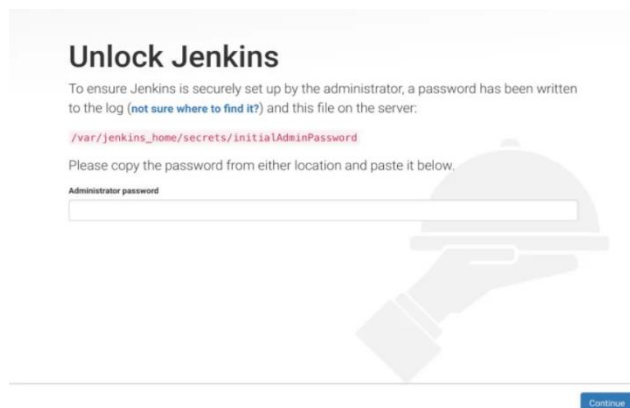
Linha 9: Mapeia volumes entre o host e o container

Após a criação do nosso arquivo .yml vamos executar o comando “Docker compose up -d” para a execução do contêiner Jenkins.

---

Na url do seu Browser acesse “localhost:8080” para acessar o Jenkins

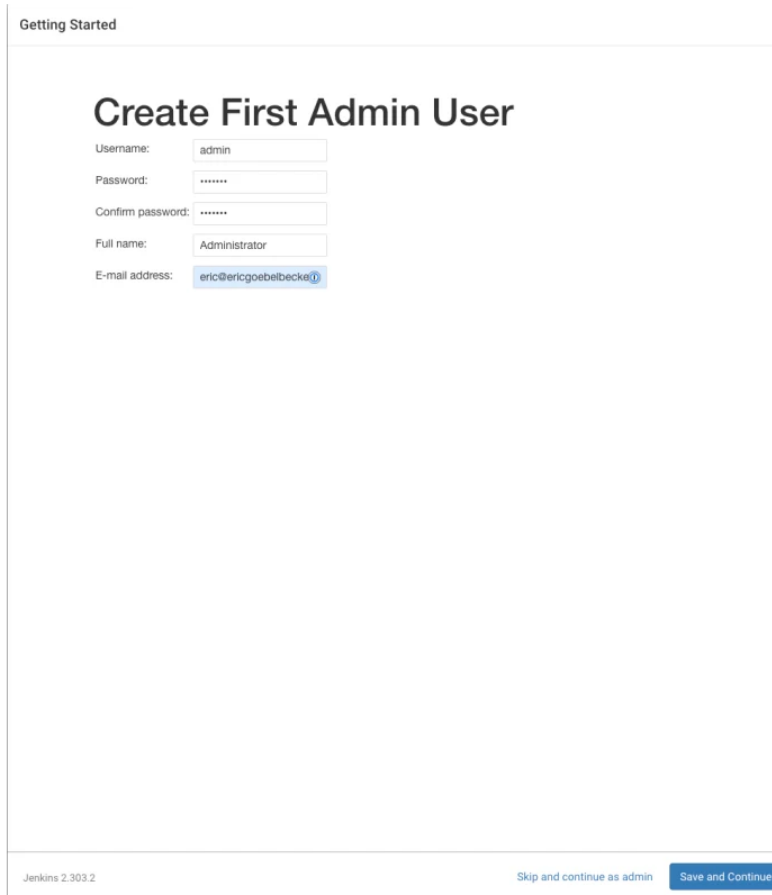
Primeiramente você será direcionado para a página do administrador que pedirá uma senha para poder prosseguir.



A senha será mostrada no terminal, porém se não aparecer ela poderá ser encontrada no seguinte caminho do projeto:

`/var/jenkins_home/secrets/inicialAdminPassword`

Após inserir a chave, escolha a opção para instalar os “plugins sugeridos”. Quando terminar a instalação irá aparecer uma nova tela para a criação do administrador e a senha.



The screenshot shows the 'Getting Started' page of Jenkins. The main heading is 'Create First Admin User'. Below it, there are five input fields: 'Username' (filled with 'admin'), 'Password' (filled with '\*\*\*\*\*'), 'Confirm password' (filled with '\*\*\*\*\*'), 'Full name' (filled with 'Administrator'), and 'E-mail address' (filled with 'eric@ericgoebelbecke@'). At the bottom right, there are two buttons: 'Skip and continue as admin' and 'Save and Continue'. The bottom left corner shows 'Jenkins 2.303.2'.

preencha os campos, depois clique na opção “salvar e continuar”.

Na página seguinte deixe a configuração padrão e clique em “salvar e finalizar”.

Tudo certo! Você está pronto para configurar um agente.

---

## **Configuração de uma pipeline em uma aplicação ReactJs utilizando um agente em node com Docker**

No painel de controle do Jenkins procure a opção para criar uma “Nova Tarefa” e preencha o campo com o nome da Pipeline e logo abaixo selecione a opção “Pipeline”. Em seguida na opção “Definition” selecione a opção “Pipeline Script from SCM”, logo abaixo em “SCM” selecione “Git”, em seguida na opção de “repositórios” cole a url de um repositório do GitHub que apresente a aplicação React. Em “branches to build” verifique no projeto se é main ou master e em seguida salve a configuração.

Segue o exemplo a seguir:

Enter an item name


nome\_da\_pipeline\_aqui

» Required field



**Freestyle project**

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.



**Pipeline**

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Configure

- General
- Advanced Project Options
- Pipeline**

Pipeline

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

https://github.com/rhavymala/acomidadobebe-landingpage.git

Credentials ?

- none -

Add

Advanced

Add Repository

Configure

- General
- Advanced Project Options
- Pipeline**

Branches to build ?

Branch Specifier (blank for 'any') ?

\*/main

Add Branch

Repository browser ?

(Auto)

Additional Behaviours

Add

Script Path ?

Jenkinsfile

☒ Lightweight checkout ?

Pipeline Syntax

Save Apply