Installing Jenkins in CentOS/RHEL-7

To install Jenkins on your CentOS system, follow the steps below:

1. Jenkins is a Java application, so the first step is to install Java. Run the following command to install the OpenJDK 8 package:

sudo yum install java-1.8.0-openjdk-devel

The current version of Jenkins does not support Java 10 (and Java 11) yet. If you have multiple versions of Java installed on your machine make sure Java 8 is the default Java version.

2. The next step is to enable the Jenkins repository. To do that, import the GPG key using the following curl command:

curl --silent --location http://pkg.jenkins-ci.org/redhat-stable/jenkins.repo | sudo tee /etc/yum.repos.d/jenkins.repo

wget -O /etc/yum.repos.d/jenkins.repo http://pkg.jenkins.io/redhat-stable/jenkins.repo

3. And add the repository to your system with:

sudo rpm --import http://pkg.jenkins.io/redhat-stable/jenkins.io.key

Note: If any error run the following command

amazon-linux-extras install epel -y

4. Once the repository is enabled, install the latest stable version of Jenkins by typing:

sudo yum install jenkins

5. After the installation process is completed, start the Jenkins service with:

sudo systemctl start jenkins

6. To check whether it started successfully run:

systemctl status jenkins

7. Finally enable the Jenkins service to start on system boot.

sudo systemctl enable jenkins

Adjust the Firewall

If you are installing Jenkins on a remote CentOS server that is protected by a firewall you need to port 8080.

Use the following commands to open the necessary port:

sudo firewall-cmd --permanent --zone=public --add-port=8080/tcp

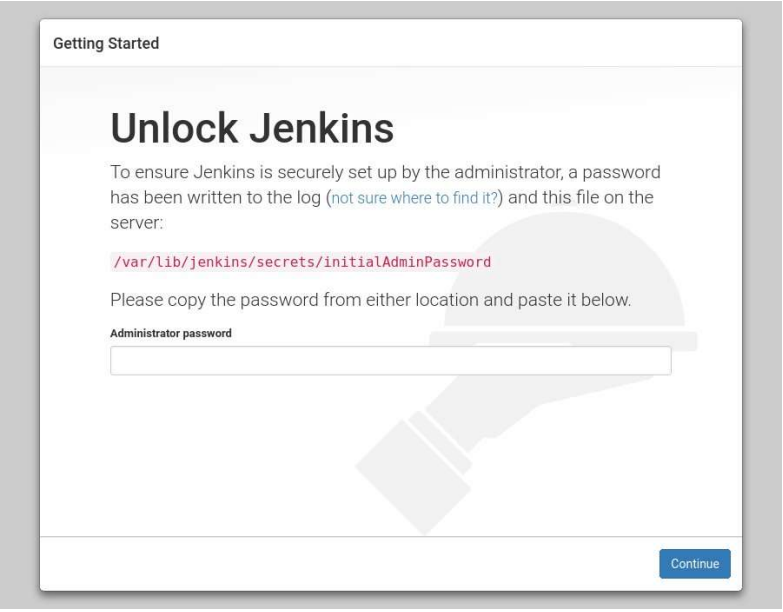
sudo firewall-cmd --reload

Setting Up Jenkins

1. To setup your new Jenkins installation, open your browser and type your domain or IP address followed by port 8080:

http://your_ip_or_domain:8080

A screen similar to the following will appear, prompting you to enter the Administrator password that is created during the installation:



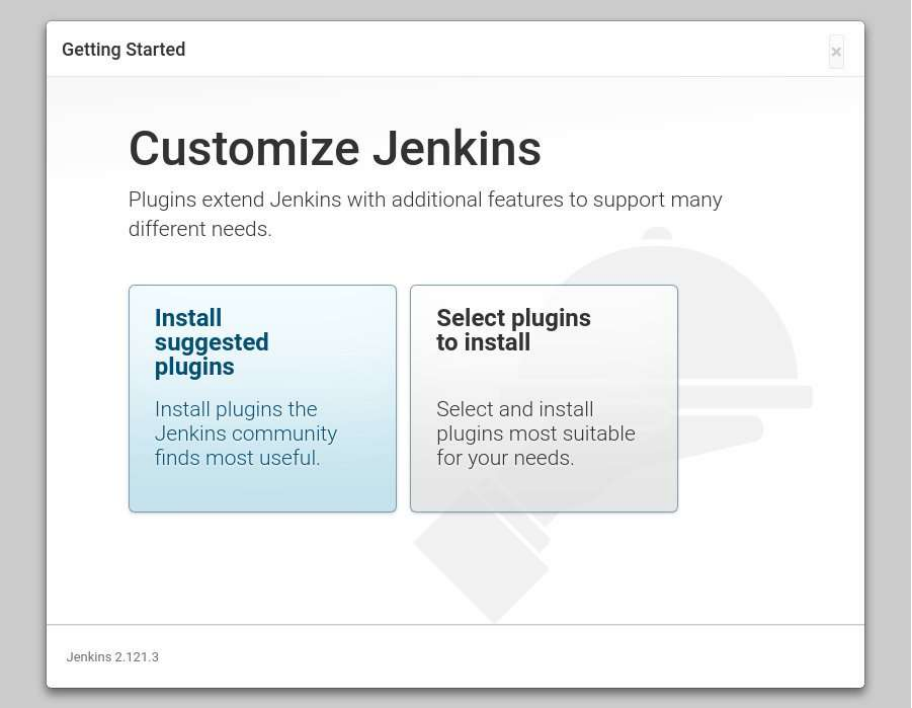
2. Use the following command to print the password on your terminal:

sudo cat /var/lib/jenkins/secrets/initialAdminPassword

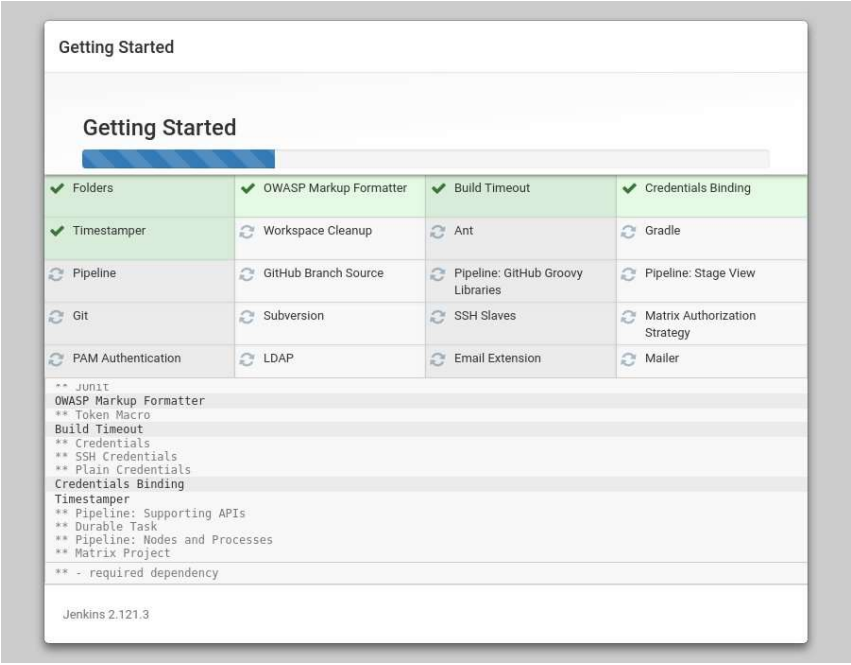
You should see a 32-character long alphanumeric password as shown bellow:

2115173b548f4e99a203ee99a8732a32

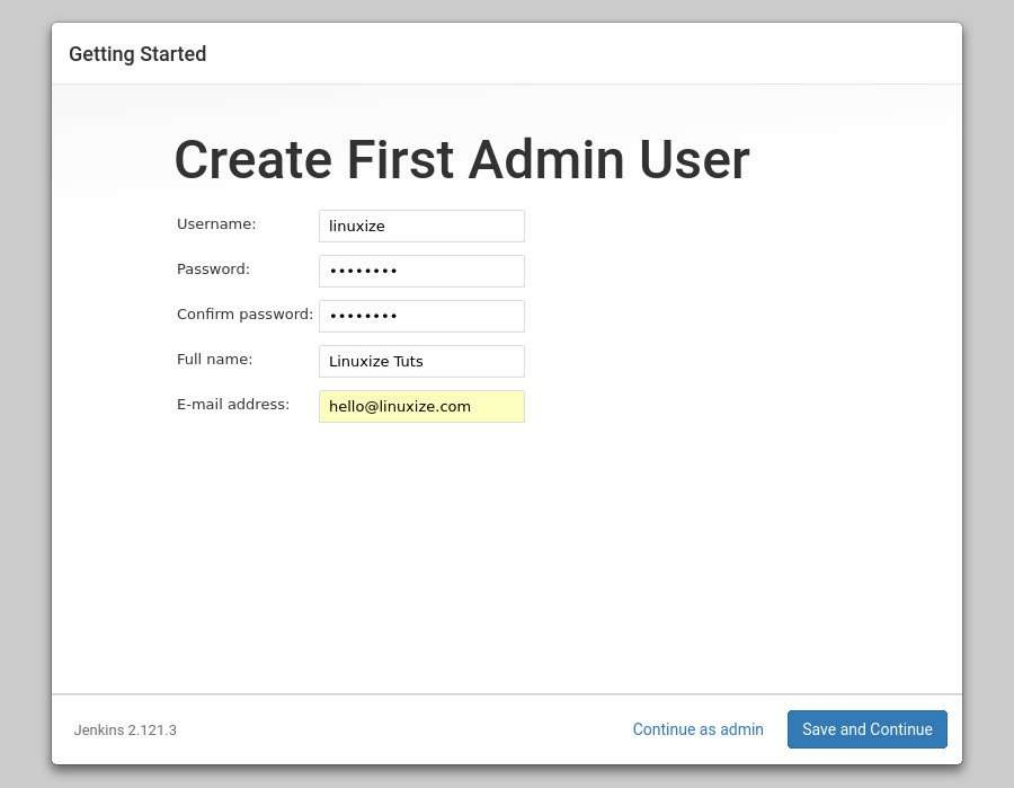
Copy the password from your terminal, paste it into the Administrator password field and click Continue.



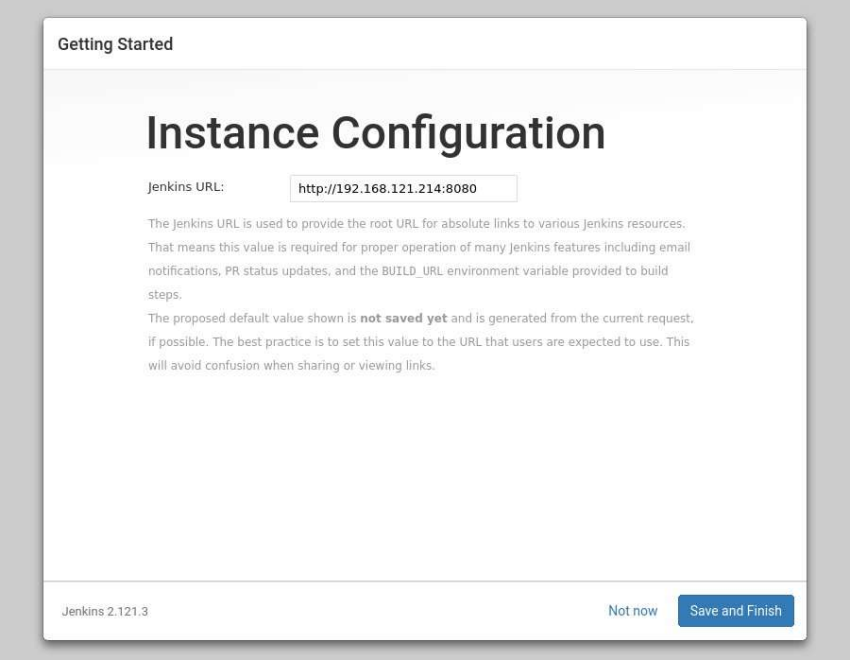
On the next screen you will be asked whether you want to install the suggested plugins or to select specific plugins. Click on the Install suggested plugins box, and the installation process will start immediately.



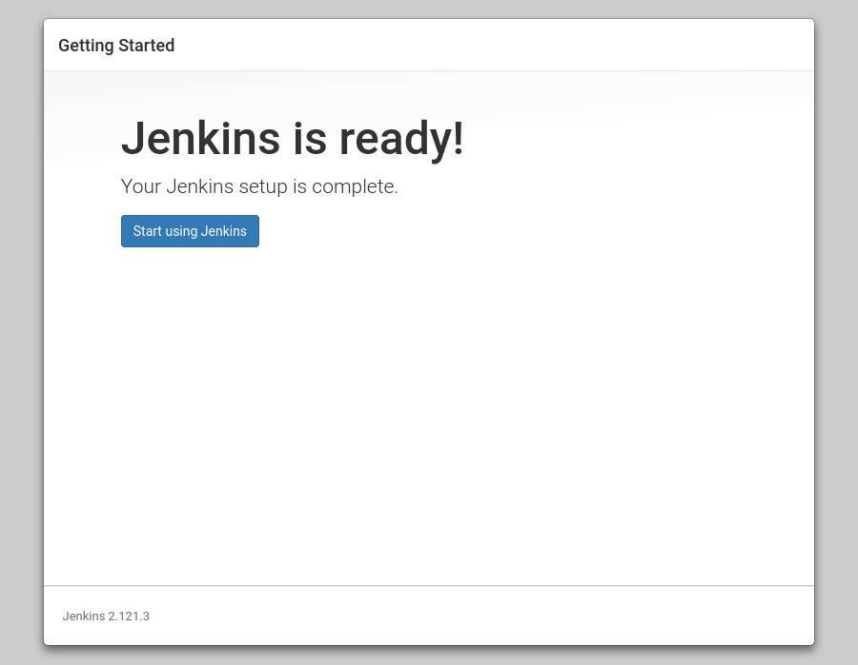
Once the installation is complete, you will be prompted to set up the first administrative user. Fill out all required information and click Save and Continue.



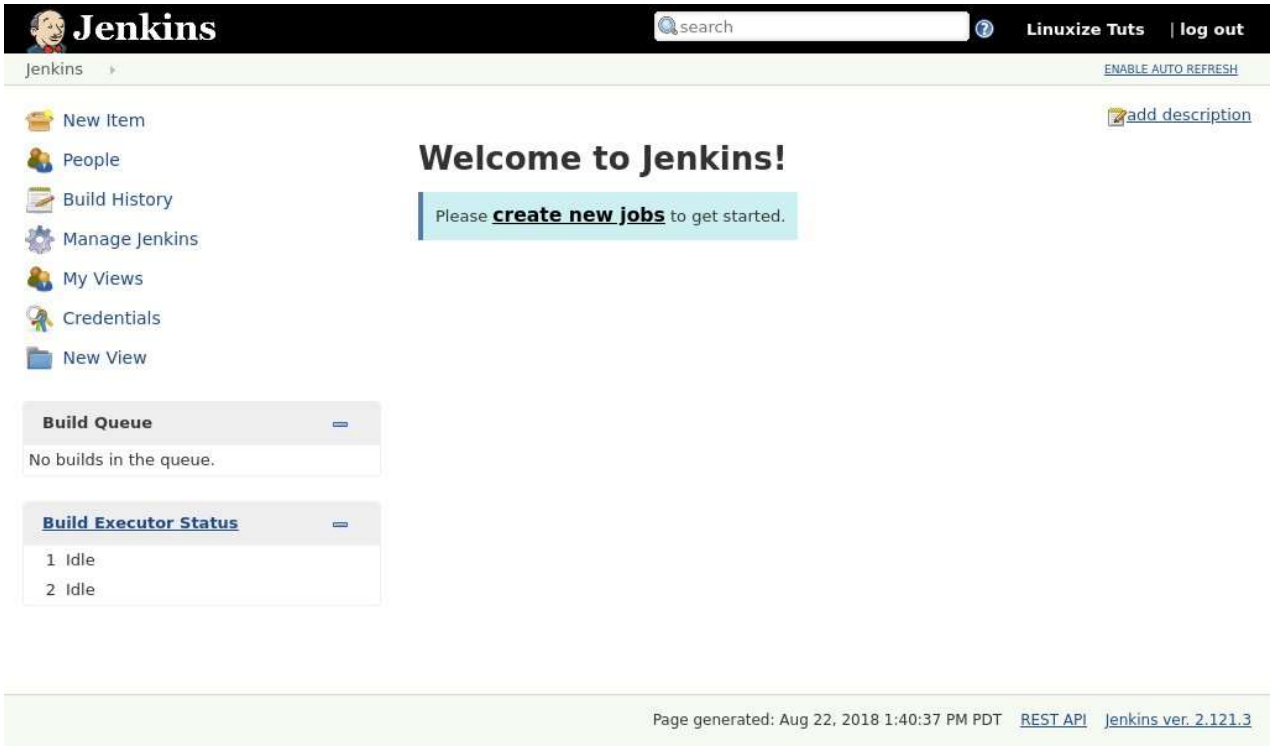
On the next page you will be asked to set the URL for the Jenkins instance. The URL filed will be populated with an automatically generated URL.



To complete the setup confirm the URL by clicking on the Save and Finish button.



Finally, click on the Start using Jenkins button and you will be redirected to the Jenkins dashboard logged in as the admin user you have created in one of the previous steps.



If you've reached this point, you've successfully installed Jenkins on your CentOS system.

```
readlink -f $(which java) - to check the installation location
```

Master slave configuration:

Slave:

yum install java-1.8.0-openjdk-devel -y

ssh-keygen -t rsa -N "" -f /root/.ssh/id_rsa

cd /root/.ssh

cat id_rsa.pub > authorized_keys

chmod 777 authorized_keys

Master:

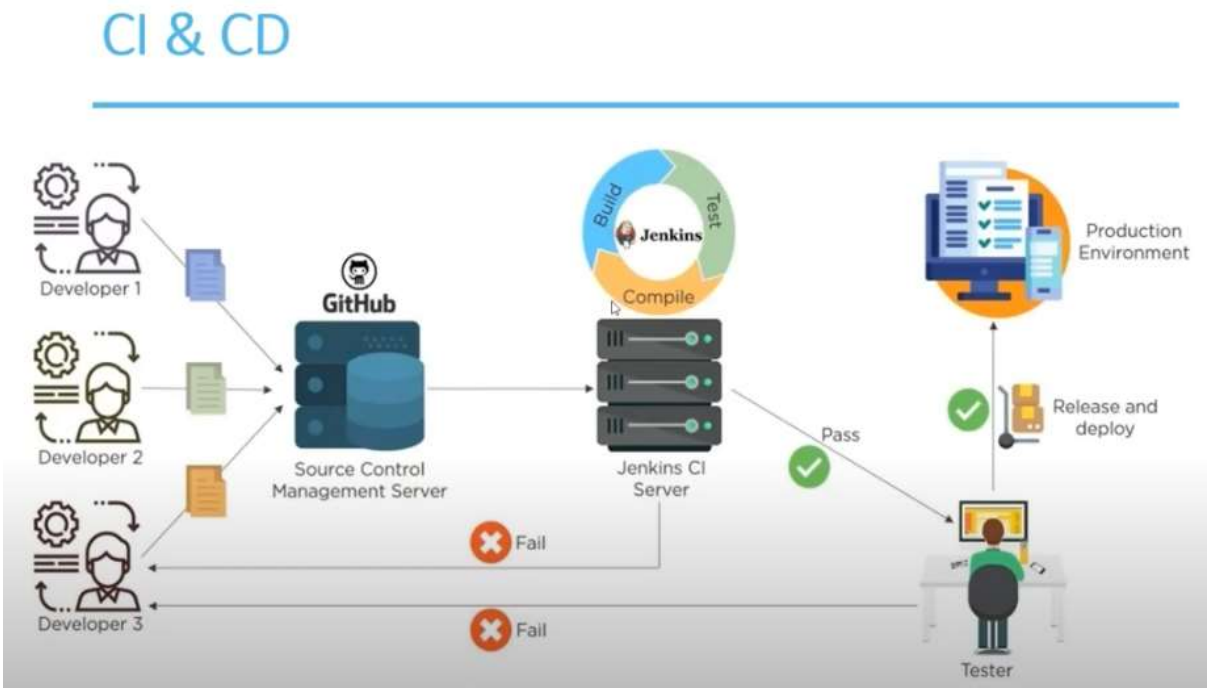
mkdir -p /var/lib/jenkins/.ssh

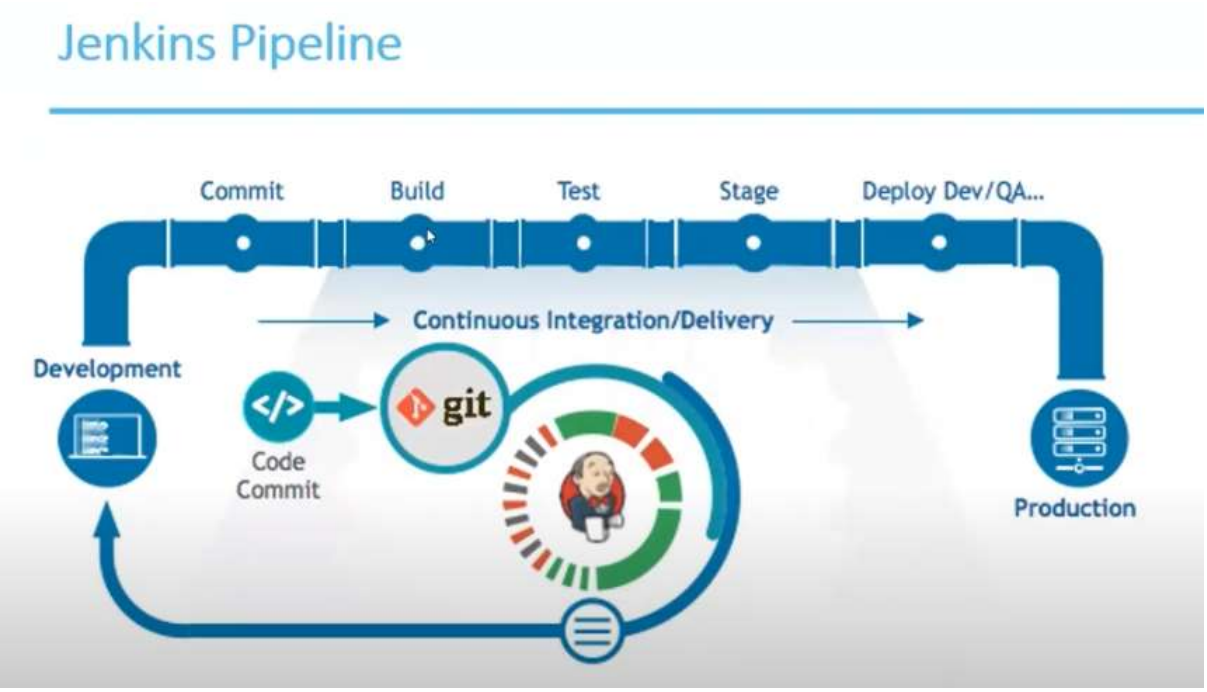
cd /var/lib/jenkins/.ssh

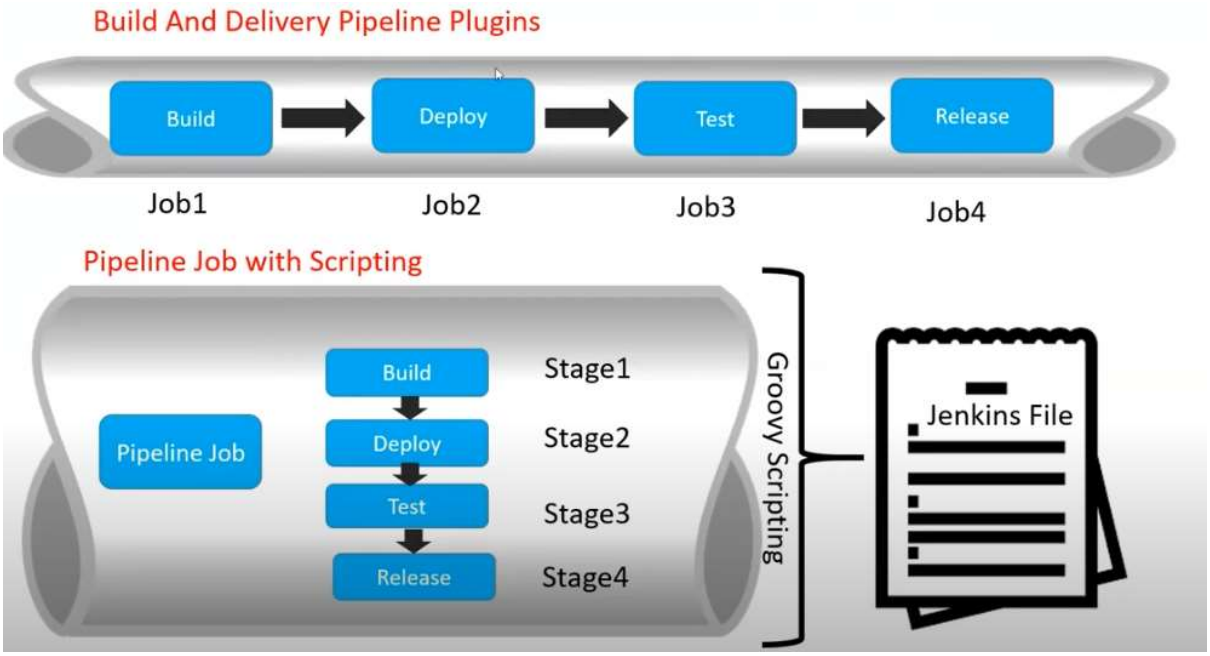
ssh-keyscan -H 10.110.2.148 >>/var/lib/jenkins/.ssh/known_hosts

chown root known_hosts

chmod 777 known_hosts




Jenkins Pipeline
Commit
Build
Test
Stage
Deploy Dev/QA...
Continuous Integration/Delivery
Development
Code
Commit
git
Production


Build And Delivery Pipeline Plugins
Build
Deploy
Test
Release
Job1
Job2
Job3
Job4
Pipeline Job with Scripting
Pipeline Job
Build
Deploy
Test
Release
Stage1
Stage2
Stage3
Stage4
Groovy Scripting
Jenkins File

Pipeline concepts

- Pipeline
- A Pipeline is a user-defined model of a CD pipeline. A Pipeline's code defines your entire build process, which typically includes stages for building an application, testing it and then delivering it.
- Also, ***a pipeline block is a key part of Declarative Pipeline syntax.***
- Node
- A node is a machine which is part of the Jenkins environment and is capable of executing a Pipeline.
- Also, ***a node block is a key part of Scripted Pipeline syntax.***
- Stage
- A stage block defines a conceptually distinct subset of tasks performed through the entire Pipeline (e.g. "Build", "Test" and "Deploy" stages), which is used by many plugins to visualize or present Jenkins Pipeline status/progress.
- Step
- A single task. Fundamentally, a step tells Jenkins what to do at a particular point in time (or "step" in the process). For example, to execute the shell command make use the sh step: sh 'make'. When a plugin extends the Pipeline DSL, that typically means the plugin has implemented a new step.

How many Ways we can create Pipeline

- We can Create Jenkins Pipeline in 2 Ways
- 1) Using **Build And Delivery Pipeline Plugins**
- **2) Using Groovy Script on the Fly(Here we use Jenkins file)**
  - Scripted
  - Declarative

Scripted Pipeline

Jenkinsfile (Scripted Pipeline)

```
node { 1
    stage('Build') { 2
        // 3
    }
    stage('Test') { 4
        // 5
    }
    stage('Deploy') { 6
        // 7
    }
}
```

1. Execute this Pipeline or any of its stages, on any available agent.
2. Defines the "Build" stage. `stage` blocks are optional in Scripted Pipeline syntax. However, implementing `stage` blocks in a Scripted Pipeline provides clearer visualization of each `stage's subset of tasks/steps in the Jenkins UI.
3. Perform some steps related to the "Build" stage.
4. Defines the "Test" stage.
5. Perform some steps related to the "Test" stage.
6. Defines the "Deploy" stage.
7. Perform some steps related to the "Deploy" stage.

```
node {
        stage('Build')
        {
           echo "Building the Project....."
        }

        stage('Test')
        {

           echo "Testing the Project....."

        }

        stage('Deploy')
        {
           echo "Deploying the Project....."
        }
}
```

Declarative Pipeline

Jenkinsfile (Declarative Pipeline)

```
pipeline{
    agent any ❶
    stages {
        stage('Build') { ❷
            steps {
                // ❸
            }
        }
        stage('Test') { ❹
            steps {
                // ❺
            }
        }
        stage('Deploy') { ❻
            steps {
                // ❼
            }
        }
    }
}
```

❶ Execute this Pipeline or any of its stages, on any available agent.
❷ Defines the "Build" stage.
❸ Perform some steps related to the "Build" stage.
❹ Defines the "Test" stage.
❺ Perform some steps related to the "Test" stage.
❻ Defines the "Deploy" stage.
❼ Perform some steps related to the "Deploy" stage.

Git hub location : https://github.com/pavanoltraining/jenkinspipeline/blob/master/jenkinsfile

Demo project link https://github.com/pavanoltraining/JenkinsPipelineDemoProject

Jenkinsfile (Declarative Pipeline)

```
pipeline {
    agent any

    stages {
        stage('Build') {
            steps {
                echo 'Building..'
            }
        }
        stage('Test') {
            steps {
                echo 'Testing..'
            }
        }
        stage('Deploy') {
            steps {
                echo 'Deploying....'
            }
        }
    }
}
```