

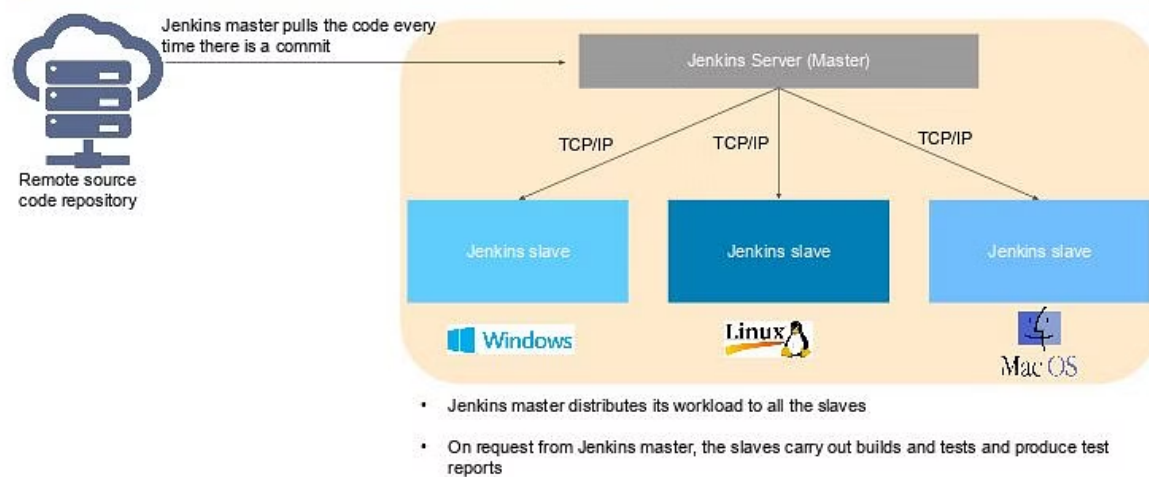
## Jenkins Architecture

Here's how Jenkins elements are put together and interact:

- Developers commit changes to the source code, found in the repository.
- The Jenkins CI server checks the repository at regular intervals and pulls any newly available code.
- The Build Server builds the code into an executable file. In case the build fails, feedback is sent to the developers.
- Jenkins deploys the build application to the test server. If the test fails, the developers are alerted.
- If the code is error-free, the tested application is deployed on the production server.

The files can contain different code and be very large, requiring multiple builds. However, a single Jenkins server cannot handle multiple files and builds simultaneously; for that, a distributed Jenkins architecture is necessary.

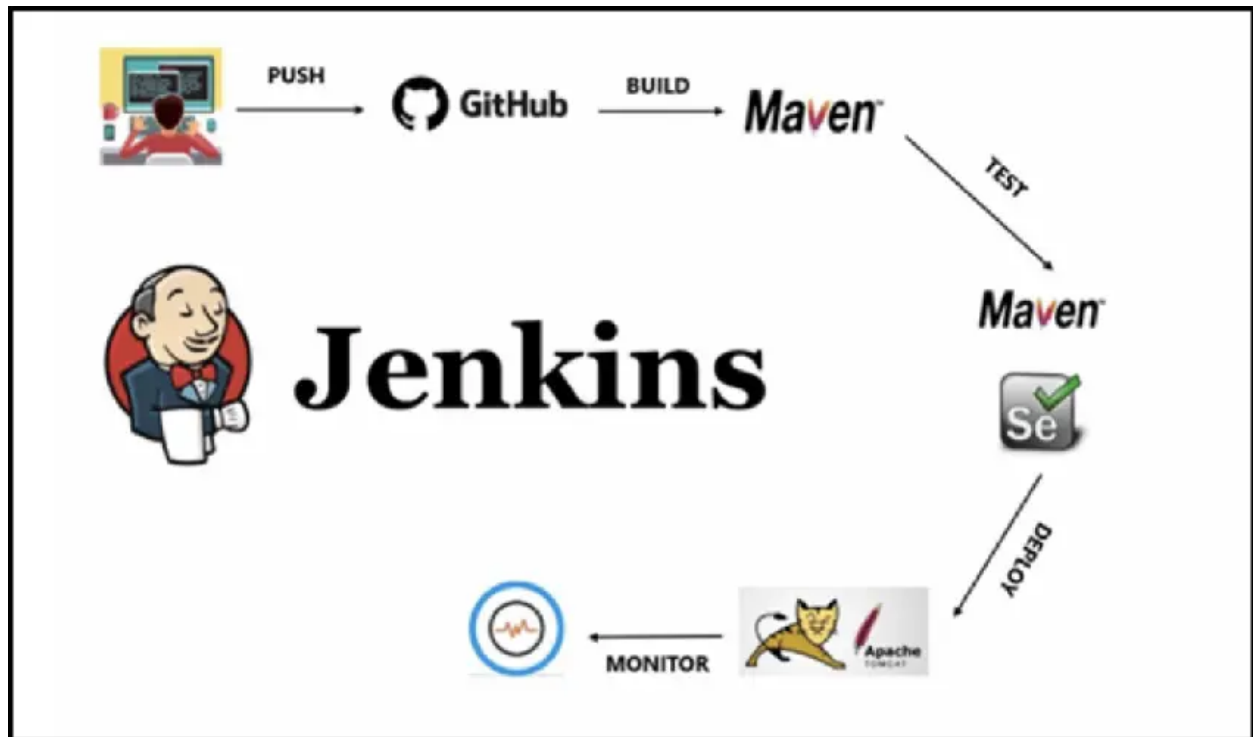
### Jenkins Master-Slave Architecture



As you can see in the diagram provided above, on the left is the Remote source code repository. The Jenkins server accesses the master environment on the left side and the master environment can push down to multiple other Jenkins Slave environments to distribute the workload.

That lets you run multiple builds, tests, and product environment across the entire architecture. Jenkins Slaves can be running different build versions of the code for different operating systems and the server Master controls how each of the builds operates.

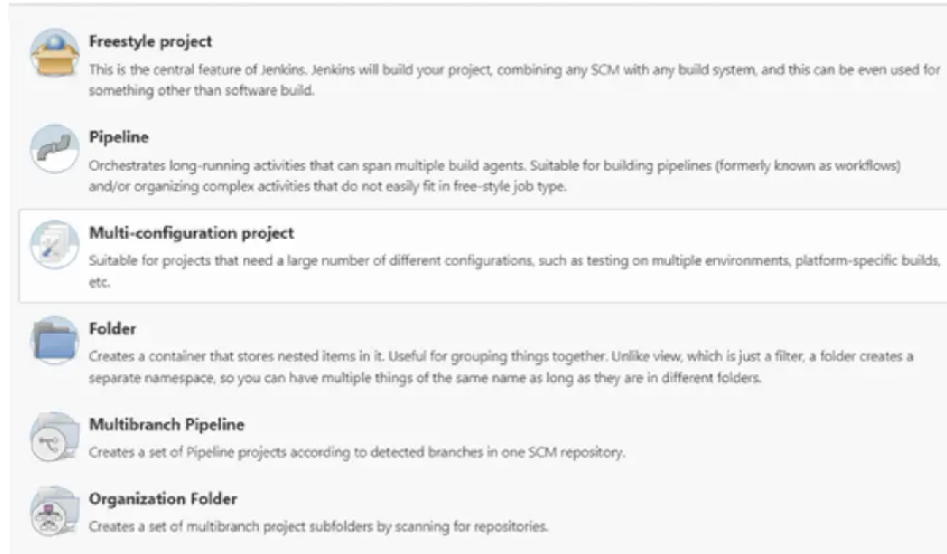
Supported on a master-slave architecture, Jenkins comprises many slaves working for a master. This architecture - the Jenkins Distributed Build - can run identical test cases in different environments. Results are collected and combined on the master node for monitoring.



## What are Jenkins Jobs?

Jenkins Jobs are a given set of tasks that runs sequentially as defined by the user. Any automation implemented in Jenkins is a Jenkins Job. These jobs are a significant part of Jenkins's build process. We can create and build Jenkins jobs to test our application or project.

When working with Jenkins, the term “Jenkins Job” and “Jenkins Project” are synonymous. With a Jenkins Job, you can clone source code from version control like Git, compile the code, and run unit tests based on your requirements. Also, Jenkins allows you to merge code using other code management tools like Supervernion, CVS, CVN, perforce, etc.



## Types Of Jenkins Jobs

There are different Jenkins Job types available intended for different purposes. Based on the complexity and nature of your project, you can choose the one that best suits your needs. Let us look at the different types of jobs in Jenkins briefly:

Job Type	Description
Freestyle Project	This is the central and the most widely used feature in Jenkins. It is an available Jenkins build job offering multiple operations. Using this option, you can build and run pipelines or scripts seamlessly.
Maven Project	If your work involves managing and building projects containing POM files, you prefer using Maven Project to build jobs in Jenkins. On choosing this option, Jenkins, by default, will pick the POM files, make configurations, and run builds.
Pipeline	Freestyle Project is often not a good option to create Jenkins Jobs. Therefore, Pipeline is the best option. Use the option Pipeline for creating Jenkins Jobs, especially when working on long-running activities.
Multi-configuration Project	If you are working on a project requiring multiple configurations, you prefer to use the Multi-configuration Project option. This option allows for making multiple configurations for testing in multiple environments.
GitHub Organization	If you click on this option, it scans the User's GitHub account for all repositories. And then, it matches markers as defined.

## Setup & Configure Jenkins on an AWS EC2 Virtual Machine:

Navigate to AWS EC2 service and click on “Launch instance” button on top right. Provide the basic details like Instance name as “Jenkins-server”, AMI as “Ubuntu 22.04” and make sure you select the correct Key Pair to authenticate while SSH into the launched EC2 instance.

Once all the EC2 instance is up and running, select the security group associated with the newly created EC2 instance and add Custom TCP Port number 8080 with 0.0.0.0/0 CIDR range as we will need to access Jenkins running on this port from the external world via public internet.

Now, select the created Jenkins-server and click on Connect button from the top menu options. Opt for SSH method and copy the SSH command from the bottom. Paste the SSH connection command on the local terminal from the location where the Key Pair file which you selected while creating the EC2 server exists.

To configure the server, start installing Jenkins server on this newly launched EC2 instance Ubuntu 22.04 VM by following the below listed steps:

As a prerequisite, we need to install Java on the VM where Jenkins is to be installed. To install OpenJava version on the EC2 VM follow the below steps:

1. To install the OpenJDK version of Java, first update your apt package index:

```
sudo apt update
```

2. Next, check if Java is already installed:

```
java -version
```

3. If Java is not currently installed, you'll get the following output:

### Output

```
Command 'java' not found, but can be installed with:
```

```
sudo apt install default-jre          # version 2:1.11-72build1, or
sudo apt install openjdk-11-jre-headless # version 11.0.14+9-0ubuntu2
sudo apt install openjdk-17-jre-headless # version 17.0.2+8-1
sudo apt install openjdk-18-jre-headless # version 18~36ea-1
sudo apt install openjdk-8-jre-headless  # version 8u312-b07-0ubuntu1
```

4. Execute the following command to install the JRE from OpenJDK 11:

```
sudo apt install default-jre -y
```

The JRE will allow you to run almost all Java software.

5. Verify the installation with:

```
java -version
```

6. You'll receive output similar to the following:

```
Output
openjdk version "11.0.14" 2022-01-18
OpenJDK Runtime Environment (build 11.0.14+9-Ubuntu-0ubuntu2)
OpenJDK 64-Bit Server VM (build 11.0.14+9-Ubuntu-0ubuntu2, mixed mode, sharing)
```

7. You may need the JDK in addition to the JRE in order to compile and run some specific Java-based software. To install the JDK, execute the following command, which will also install the JRE:

```
sudo apt install default-jdk -y
```

8. Verify that the JDK is installed by checking the version of javac, the Java compiler:

```
javac -version
```

9. You'll see the following output:

```
Output
javac 11.0.14
```

Now, you can start installing Jenkins server by following the below steps:

1. First, add the repository key to the system:

```
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo tee
/usr/share/keyrings/jenkins-keyring.asc > /dev/null
```

After the key is added the system will return with OK.

2. Next, let's append the Debian package repository address to the server's sources.list:

```
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]
https://pkg.jenkins.io/debian-stable binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list >
/dev/null
```

3. After both commands have been entered, we'll run update so that apt will use the new repository.

```
sudo apt update --allow-insecure-repositories
```

4. Finally, we'll install Jenkins and its dependencies.

```
sudo apt install jenkins -y --allow-unauthenticated
```

Now that Jenkins and its dependencies are in place, we'll start the Jenkins server. Let's start Jenkins by using systemctl:

```
sudo systemctl start jenkins
```

5. Since systemctl doesn't display status output, we'll use the status command to verify that Jenkins started successfully:

```
sudo systemctl status jenkins
```

If everything went well, the beginning of the status output shows that the service is active and configured to start at boot:

```
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enable>
   Active: active (running) since Fri 2023-05-19 00:54:01 UTC; 10min ago
     Main PID: 5961 (java)
        Tasks: 36 (limit: 1141)
       Memory: 312.0M
          CPU: 42.091s
      CGroup: /system.slice/jenkins.service
              └─5961 /usr/bin/java-Djava.awt.headless=true -jar /usr/share/java/jenkins.>

May 19 00:53:31 ip-172-31-12-114 jenkins[5961]: 6e8fd28c96af4ae59ed7e5e04a73f9c6
May 19 00:53:31 ip-172-31-12-114 jenkins[5961]: This may also be found at: /var/lib/jenk>
May 19 00:53:31 ip-172-31-12-114 jenkins[5961]: <
May 19 00:53:31 ip-172-31-12-114 jenkins[5961]: <
May 19 00:53:31 ip-172-31-12-114 jenkins[5961]: <
May 19 00:54:01 ip-172-31-12-114 jenkins[5961]: 2023-05-19 00:54:01.774+0000 [id=28] >
May 19 00:54:01 ip-172-31-12-114 jenkins[5961]: 2023-05-19 00:54:01.804+0000 [id=22] >
May 19 00:54:01 ip-172-31-12-114 systemd[1]: Started Jenkins Continuous Integration Serv>
May 19 00:54:01 ip-172-31-12-114 jenkins[5961]: 2023-05-19 00:54:01.936+0000 [id=44] >
May 19 00:54:01 ip-172-31-12-114 jenkins[5961]: 2023-05-19 00:54:01.937+0000 [id=44] >
lines 1-20/20 (END)
```

After making sure the Jenkins server is up and running you can now go to the EC2 Console and

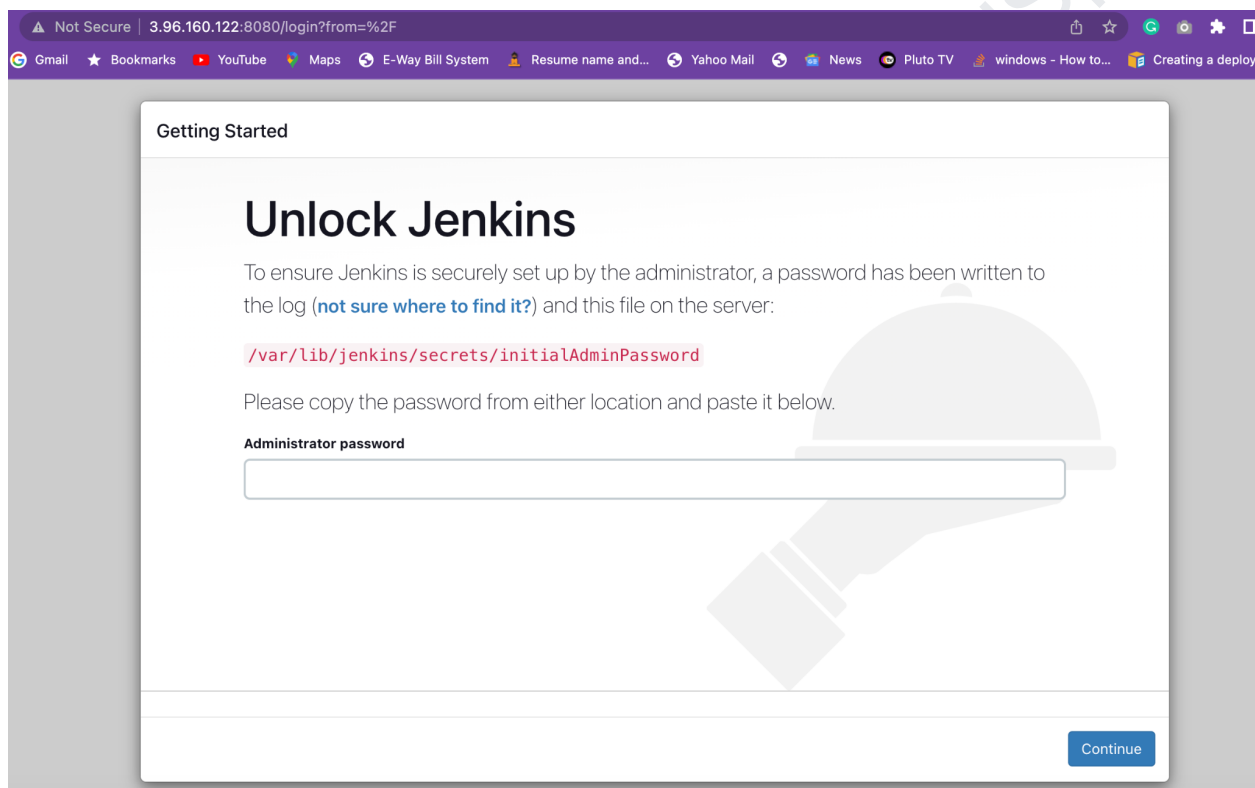
copy the public IP address of the server and prepare the Jenkins URL as:

<http://public-ip-address:8080/>

This will open the below web page on the browser where you need to provide the initial Admin password for the Jenkins server which you can get by executing a cat command on the terminal connected with the EC2 VM as given below:

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

This will give you the password and simply copy paste it in the screen given below:



Click on the Install Suggested plugins and let Jenkins install the plugins needed to support the CI/CD operations as below:

# Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

## Install suggested plugins

Install plugins the Jenkins community finds most useful.

## Select plugins to install

Select and install plugins most suitable for your needs.

Jenkins 2.387.3

Now create the Jenkins user by providing the details required as given below:

## Create First Admin User

Username

Password

Confirm password

Full name

E-mail address

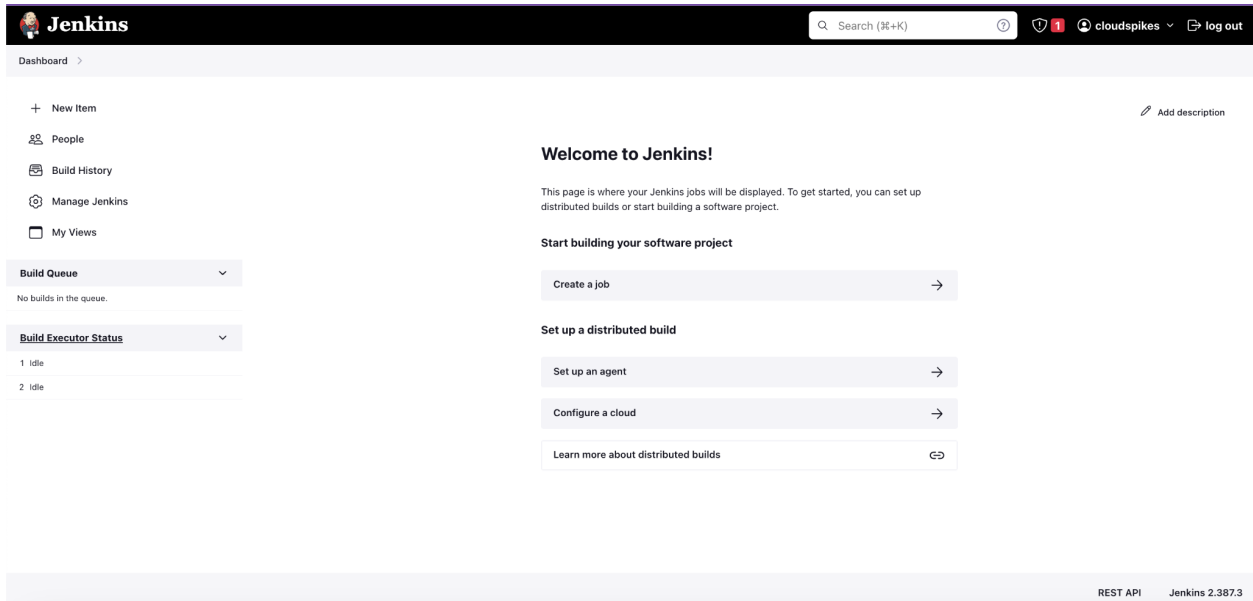
Jenkins 2.387.3

[Skip and continue as admin](#)

[Save and Continue](#)



Click “Save and Continue” after providing the required user information and save the “Jenkins URL” in the next screen. Finally, click on “Start using Jenins” and you will be redirected to the home page of the Jenkins dashboard as below:



The screenshot shows the Jenkins dashboard interface. At the top is a dark header with the Jenkins logo, a search bar, and user information. Below the header is a sidebar with navigation links like 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'My Views'. The main area displays a 'Welcome to Jenkins!' message and options to 'Start building your software project' (Create a job) and 'Set up a distributed build' (Set up an agent, Configure a cloud, Learn more about distributed builds). The bottom of the dashboard shows 'REST API' and 'Jenkins 2.387.3'.

**Jenkins**

Search (英+K)

cloudspikes log out

Dashboard

+ New Item

People

Build History

Manage Jenkins

My Views

**Build Queue**

No builds in the queue.

**Build Executor Status**

1 Idle

2 Idle

**Welcome to Jenkins!**

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

**Start building your software project**

Create a job →

**Set up a distributed build**

Set up an agent →

Configure a cloud →

Learn more about distributed builds ↗

REST API Jenkins 2.387.3