



Pimpri Chinchwad Education Trust's
Pimpri Chinchwad College of Engineering
(PCCoE)
(An Autonomous Institute)
Affiliated to Savitribai Phule Pune
University(SPPU) ISO 21001:2018 Certified by
TUV



Course: DevOps Laboratory	Code: BIT26VS01
Name: Amar Vajinath Chavan	PRN: 124B2F001
Assignment 3: Study of Jenkins Architecture and Plugins with Installation and Configuration.	

Aim: To install Jenkins on an AWS EC2 Master node, configure Java on both Master and Worker nodes, and establish a secure connection between them using SSH keys.

Objectives:

1. To understand the concept of Continuous Integration/Continuous Deployment (CI/CD).
2. To study Jenkins Controller-Agent (Master-Slave) architecture.
3. To install and configure Jenkins with essential plugins for Java/Maven projects.

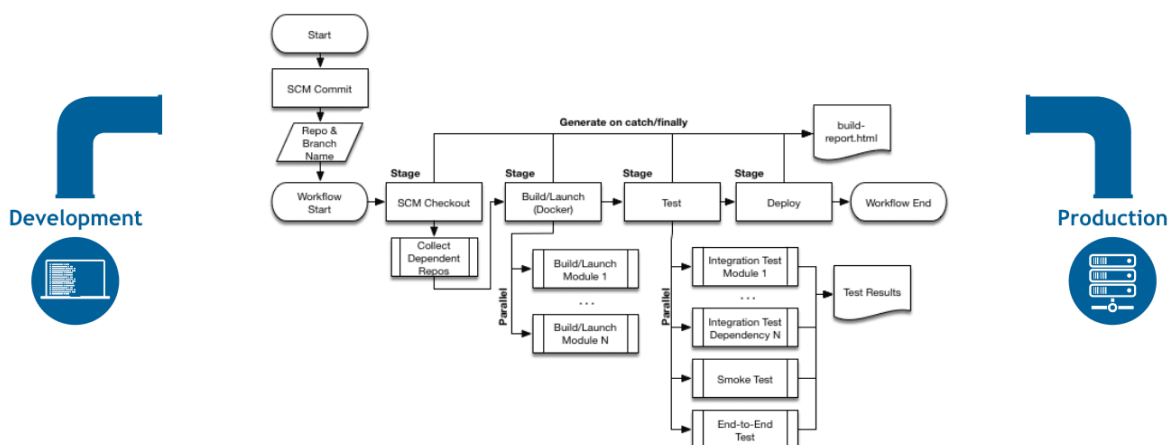
Prerequisites:

- AWS Account with two EC2 Instances (Ubuntu 24.04).
- Security Groups configured for Ports 8080 (Jenkins UI), 22 (SSH), and 8081 (Application).
- Java Development Kit (JDK 21) installed on both nodes.

Theory:

1. Jenkins Overview

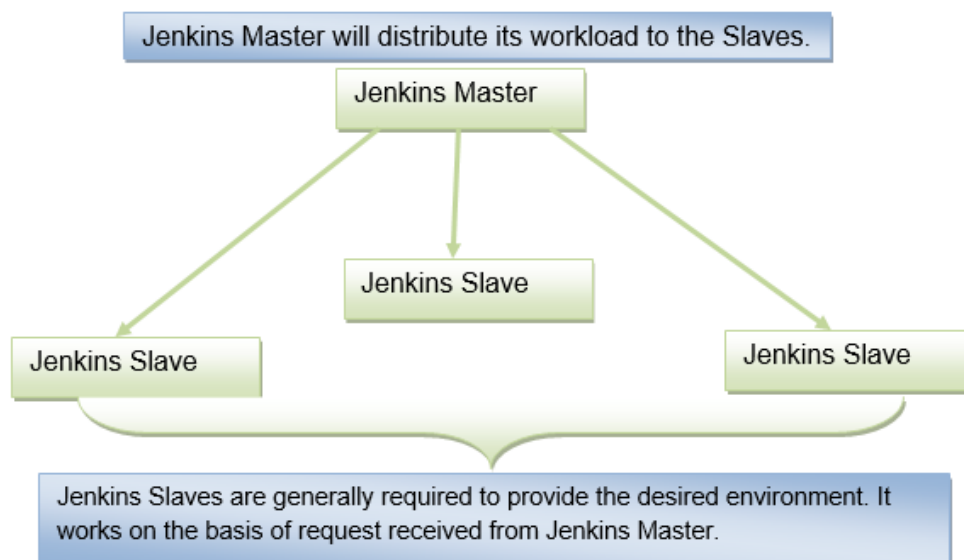
Jenkins is an open-source automation server that facilitates the technical aspects of Continuous Integration and Continuous Delivery. It automates the process of building, testing, and deploying software, allowing teams to focus on code quality.



2. Jenkins Architecture (Controller-Agent)

Jenkins follows a distributed architecture to handle high workloads:

- **Controller (Master):** The "Brain." It manages the configuration, scheduling builds, and monitoring agents.
- **Agent (Worker):** The "Muscle." It executes the actual build commands sent by the controller. This prevents the controller from becoming a bottleneck.



3. Jenkins Plugins

Plugins are the heart of Jenkins' extensibility. They allow integration with various tools:

- **Git Plugin:** Enables Jenkins to pull source code from GitHub.
- **Maven Integration:** Specifically designed for building Java-based projects.
- **SSH Build Agents:** Used to connect the Controller to remote Workers via SSH.
- **Pipeline Plugin:** Allows defining the CI/CD flow as code (Jenkinsfile).

Practical Procedure / Steps:

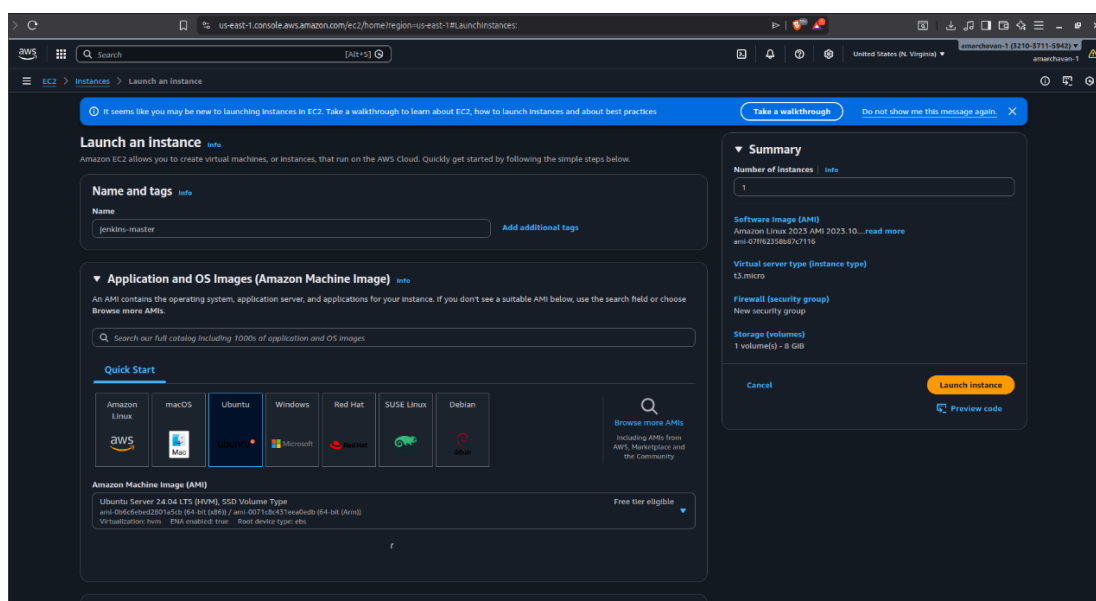
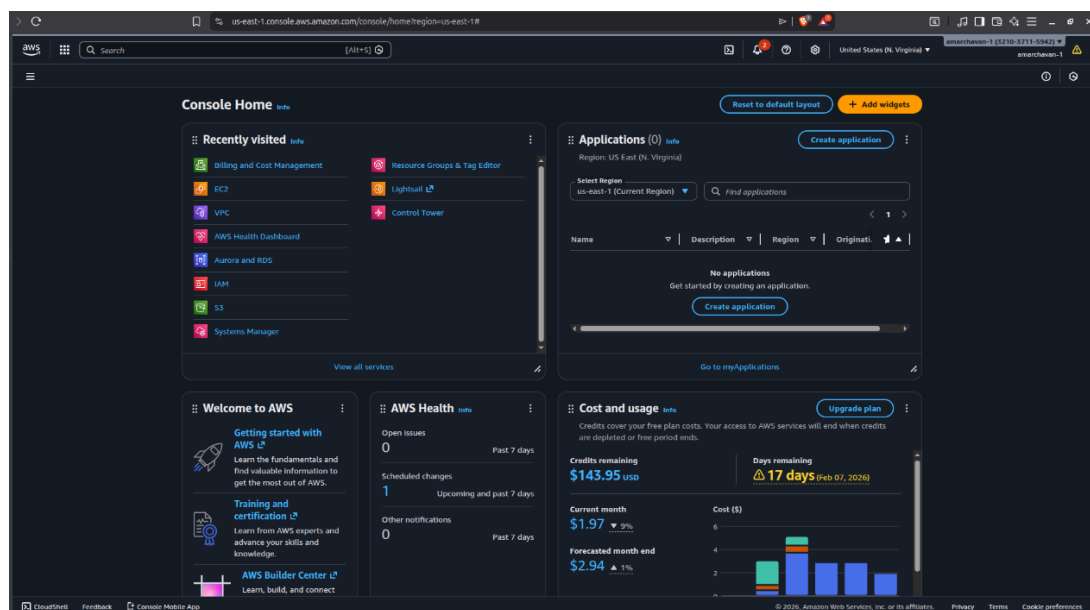
Step 1: AWS EC2 Instance Creation

Before installing Jenkins, we must provision the virtual servers. Repeat these steps for both the **Master** and **Worker** nodes.

1. **Sign in to AWS Console:** Navigate to the EC2 Dashboard.
2. **Launch Instance:** Click on the "Launch Instance" button.
3. **Name and Tags:** Name the first instance: Jenkins-Master.
Name the second instance: Jenkins-Worker.
4. **Application and OS Image:** Select **Ubuntu 24.04 LTS** (Free Tier eligible).
5. **Instance Type:** Select **t2.micro** (1 vCPU, 1 GiB Memory). [or free eligible]

6. **Key Pair:** Select an existing key pair or create a new one (e.g., devops-lab.pem) to enable SSH access.
7. **Network Settings (Security Groups):** * Create a Security Group named Jenkins-SG.
 - **Inbound Rules for Master:**
 - Type: **SSH** | Port: **22** | Source: **My IP**
 - Type: **Custom TCP** | Port: **8080** | Source: **Anywhere (0.0.0.0/0)**
 - **Inbound Rules for Worker:**
 - Type: **SSH** | Port: **22** | Source: **My IP**
 - Type: **Custom TCP** | Port: **8081** | Source: **Anywhere (0.0.0.0/0)** (For your app)
8. **Launch:** Click "Launch Instance" and wait for the status to change to Running.

Jenkins-Master Creation EC2 Screenshots



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#launchinstances

Search [Alt+S]

United States (N. Virginia) amarchavan-1 (1216-3715-5842) amarchavan-1

EC2 > Instances > Launch an instance

ssh TCP 22

Source type: Anywhere Source: 0.0.0.0/0 Description - optional: e.g. SSH for admin desktop

Security group rule 2 (TCP: 8080, 0.0.0.0/0) Remove

Type: Custom TCP Protocol: TCP Port range: 8080

Source type: Anywhere Source: 0.0.0.0/0 Description - optional: e.g. SSH for admin desktop

Security group rule 3 (TCP: 80, 0.0.0.0/0) Remove

Type: HTTP Protocol: TCP Port range: 80

Source type: Anywhere Source: 0.0.0.0/0 Description - optional: e.g. SSH for admin desktop

Security group rule 4 (TCP: 443, 0.0.0.0/0) Remove

Type: HTTPS Protocol: TCP Port range: 443

Source type: Anywhere Source: 0.0.0.0/0 Description - optional: e.g. SSH for admin desktop

Summary

Number of instances: 1

Software image (AMI): Canonical, Ubuntu, 24.04, amd64...read more ami-0b6c4ebc2801a5cb

Virtual server type (instance type): m7i-flex.large

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

Cancel Launch instance Preview code

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#launchinstances

Search [Alt+S]

United States (N. Virginia) amarchavan-1 (1216-3715-5842) amarchavan-1

EC2 > Instances > Launch an instance

Launching instance Creating security groups 27%

Details

Please wait while we launch your instance.
Do not close your browser while this is loading.

Jenkins-Worker (Slave) Creation EC2 Screenshots

Launch an instance

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags

Name: jenkins-worker-1

Application and OS Images (Amazon Machine Image)

Search our full catalog including 1000s of application and OS images

Recent: Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, Debian

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), 55D Volume Type

ami-0b6c6ebd2801a5cb (64-bit (x86)) / ami-0071dc431ea9dcb (64-bit (Arm))

Virtualization: hvm, ENA enabled: true, Root device type: ebs

Description

Ubuntu Server 24.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Canonical, Ubuntu, 24.04, amd64 noble image

Architecture: 64-bit (x86), AMI ID: ami-0b6c6ebd2801a5cb, Publish Date: 2025-12-12, Username: ubuntu

Summary

Number of instances: 1

Software Image (AMI): Canonical, Ubuntu, 24.04, amd64...

Virtual server type (instance type): t3.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

Launch instances

Key pair (login)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - required: jenkins-worker-1

Network settings

VPC - required: vpc-0d5935897d06791fb (default)

Subnet: subnet-0a05df25baca79ec

Auto-assign public IP: Enable

Firewall (security groups)

Create security group (selected) or Select existing security group

Security group name - required: launch-wizard-5

Description - required: launch-wizard-5 created 2026-01-23T09:32:18.540Z

Inbound Security Group Rules

Security group rule 1 (TCP, 22, 0.0.0.0/0)

Type: ssh, Protocol: TCP, Port range: 22

Summary

Number of instances: 1

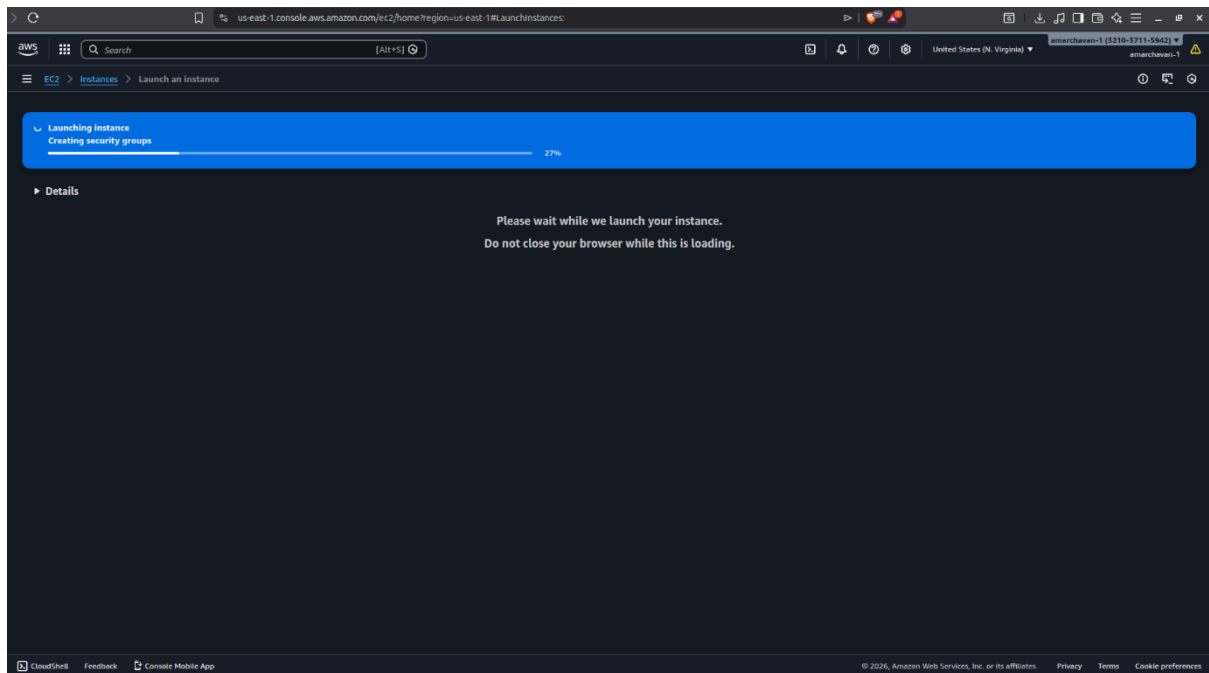
Software Image (AMI): Canonical, Ubuntu, 24.04, amd64...

Virtual server type (instance type): t3.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB

Launch instance



Step 2: Installation on Master Node

Once the instances are running, connect to the **Master** via SSH and execute:

1. Install Java (Prerequisite):

Update the Debian apt repositories, install OpenJDK 21, and check the installation using the following commands:

```
sudo apt update
sudo apt install fontconfig openjdk-21-jre
java -version
```

2. Install Jenkins:

A LTS (Long-Term Support) release is chosen every 12 weeks from the stream of regular releases as the stable release for that time period. It can be installed from the debian-stable apt repository.

```
sudo wget -O /etc/apt/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian-stable/jenkins.io-2026.key
echo "deb [signed-by=/etc/apt/keyrings/jenkins-keyring.asc]" \
https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
```

```
sudo apt update
```

```
sudo apt install Jenkins
```

3. Check Status:

sudo systemctl status Jenkins

Screenshots of Installation on the Mater Node

```
mar@amar-Inspiron-3501: ~/Desktop/AWS-EC2-Jenkins$ chmod 400 "jenkins-master.pem"
mar@amar-Inspiron-3501: ~/Desktop/AWS-EC2-Jenkins$ ssh -i "jenkins-master.pem" ubuntu@ec2-98-93-212-109.compute-1.amazonaws.com
The authenticity of host 'ec2-98-93-212-109.compute-1.amazonaws.com (64:ff9b::625d:d46d)' can't be established.
ED25519 key fingerprint is SHA256:7eL4MsZy8PMsg/2nxIYTU0NDvd2Zbo70M9oBzLI8w.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-98-93-212-109.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1018-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Fri Jan 23 03:44:54 UTC 2026

System load:  0.15          Temperature:   -273.1 C
Usage of /:   6.4% of 27.05GB Processes:      155
Memory usage: 3%           Users logged in: 0
Swap usage:   0%           IP4 address for enp3s0: 172.31.0.106

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-0-106: $ _

ubuntu@ip-172-31-0-106: $ java -version
openjdk version "21.0.9" 2025-10-21
OpenJDK Runtime Environment (build 21.0.9+10-Ubuntu-124.04)
OpenJDK 64-Bit Server VM (build 21.0.9+10-Ubuntu-124.04, mixed mode, sharing)
ubuntu@ip-172-31-0-106: $ sudo wget -O /etc/apt/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian-stable/jenkins.io-2026.key
echo "deb [signed-by=/etc/apt/keyrings/jenkins-keyring.asc]" \
https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt update
sudo apt install jenkins
2026-01-23 03:48:31-- https://pkg.jenkins.io/debian-stable/jenkins.io-2026.key
Resolving pkg.jenkins.io (pkg.jenkins.io)... 146.75.34.133, 2a04:de42:79:645
Connecting to pkg.jenkins.io (pkg.jenkins.io)[146.75.34.133]:443... connected.
HTTP request sent, awaiting response... 200 OK
length: 1680 (1.6K) [application/octet-stream]
Saving to: '/etc/apt/keyrings/jenkins-keyring.asc'

/etc/apt/keyrings/jenkins-keyring.asc 100%[=====] 1.64K --.-KB/s in 0s

2026-01-23 03:48:31 (25.2 MB/s) - '/etc/apt/keyrings/jenkins-keyring.asc' saved [1680/1680]

Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Ign:4 https://pkg.jenkins.io/debian-stable binary/ InRelease
Get:5 https://pkg.jenkins.io/debian-stable binary/ Release [2944 B]
Get:6 https://pkg.jenkins.io/debian-stable binary/ Release.gpg [833 B]
Hit:7 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:8 https://pkg.jenkins.io/debian-stable binary/ Packages [30.5 kB]
Fetched 33.4 kB in 0s (85.3 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
57 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done

Jan 23 03:48:45 ip-172-31-0-106 jenkins[4106]: [LF]> This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Jan 23 03:48:45 ip-172-31-0-106 jenkins[4106]: [LF]>
Jan 23 03:48:45 ip-172-31-0-106 jenkins[4106]: [LF]>
Jan 23 03:48:45 ip-172-31-0-106 jenkins[4106]: [LF]>
Jan 23 03:48:45 ip-172-31-0-106 jenkins[4106]: [LF]>
Jan 23 03:48:48 ip-172-31-0-106 jenkins[4106]: 2026-01-23 03:48:48.943+0000 [id=40] INFO jenkins.InitReactorRunner$1:nonAttained: Completed initialization
Jan 23 03:48:48 ip-172-31-0-106 jenkins[4106]: 2026-01-23 03:48:48.972+0000 [id=30] INFO hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running
Jan 23 03:48:48 ip-172-31-0-106 systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Jan 23 03:48:49 ip-172-31-0-106 jenkins[4106]: 2026-01-23 03:48:49.229+0000 [id=56] INFO h.n.DownloadService$Downloadable#load: Obtained the updated data file for hudson.tasks
Jan 23 03:48:49 ip-172-31-0-106 jenkins[4106]: 2026-01-23 03:48:49.230+0000 [id=56] INFO hudson.util.Retrier#start: Performed the action check updates server successfully at t
[lines 1-20/20] (END)
```

Step 3: Configuration on Worker Node

Connect to the **Worker** via SSH and only install the Java runtime:

Install Java:

```
sudo apt update
sudo apt install fontconfig openjdk-21-jre
java -version
```

Screenshots of Configuration on Worker Node

```
amar@amar-Inspiron-3581:~/Desktop/AWS-EC2-Jenkins$ chmod 480 "jenkins-worker-1.pen"
amar@amar-Inspiron-3581:~/Desktop/AWS-EC2-Jenkins$ ssh -o ServerAliveInterval=60 -t "jenkins-worker-1.pen" ubuntu@ec2-18-287-123-211.compute-1.amazonaws.com
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1018-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Fri Jan 23 09:38:38 UTC 2026

System load: 0.05           Temperature: -273.1 C
Usage of /: 25.9% of 6.71GB Processes: 114
Memory usage: 23%          Users logged in: 0
Swap usage: 0%             IPv4 address for ens5: 172.31.6.4

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See 'man sudo_root' for details.

ubuntu@ip-172-31-6-4: ~$ sudo apt update
sudo apt install fontconfig openjdk-21-jre
java -version
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:5 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [381 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]

ubuntu@ip-172-31-6-4: ~$ java --version
openjdk 21.0.9 2025-10-21
OpenJDK Runtime Environment (build 21.0.9+10-Ubuntu-124.04)
OpenJDK 64-Bit Server VM (build 21.0.9+10-Ubuntu-124.04, mixed mode, sharing)
ubuntu@ip-172-31-6-4: ~$
```

Step 4: Establishing Master-Worker Connectivity (SSH Handshake)

This step connects the " Master Node" to the " Worker Node."

1. Generate SSH Keys on Master:

Shh -keygen


```
ubuntu@Jenkins-Master: ~/.ssh
ubuntu@Jenkins-Master: ~/.ssh$ cd ~/.ssh
ubuntu@Jenkins-Master: ~/.ssh$ ls
authorized_keys
ubuntu@Jenkins-Master: ~/.ssh$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_ed25519
Your public key has been saved in /home/ubuntu/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:odm2b0CDgd0nd0V77hTy8lJEEdHgRqY/B+T3QvXoqw ubuntu@Jenkins-Master
The key's randomart image is:
+--[ED25519 256]--+
|      o  =+ +*=+
|      . *  .+.o o|
|      + o  =.o .|
|      o o B *+.+
|      . 5 = o =.+|
|      + .  +*=+
|      + .  =*+.
|      .  .oo|
|      E.o. .+|
+----[SHA256]-----+
ubuntu@Jenkins-Master: ~/.ssh$ ls
authorized_keys  id_ed25519  id_ed25519.pub
ubuntu@Jenkins-Master: ~/.ssh$
```

2. Authorize Master on Worker:

- Copy the content of /var/lib/jenkins/.ssh/id_rsa.pub (Master).
- On the Worker node, open ~/.ssh/authorized_keys and paste the key

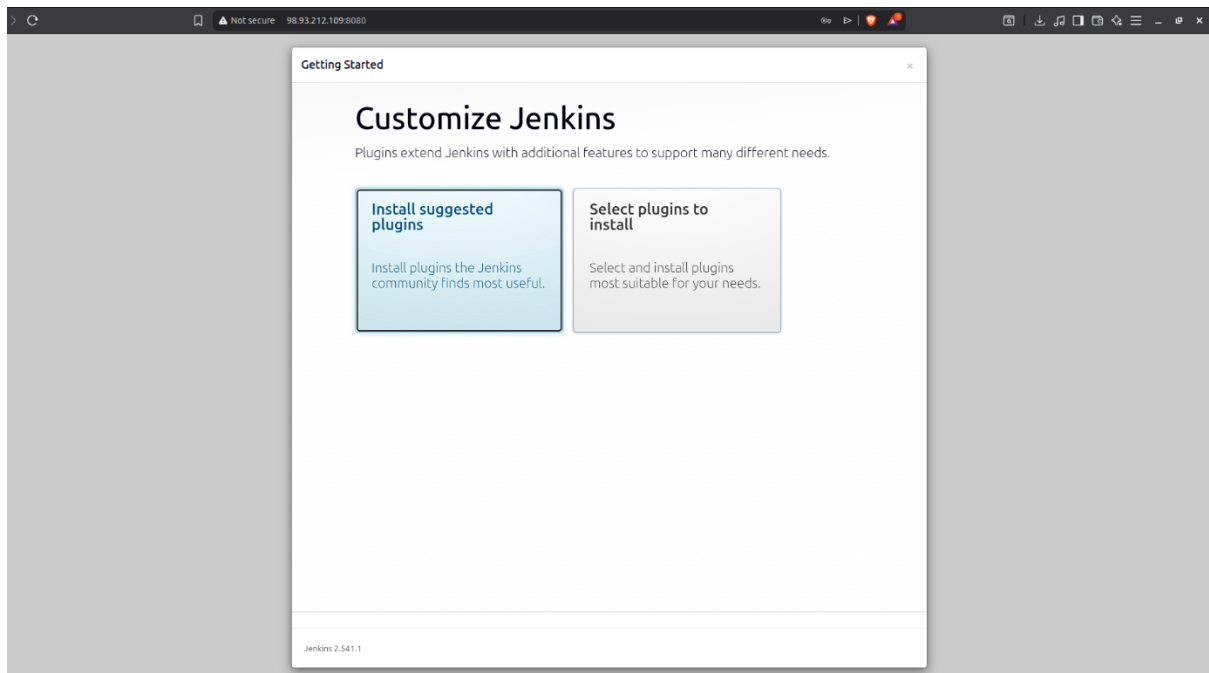
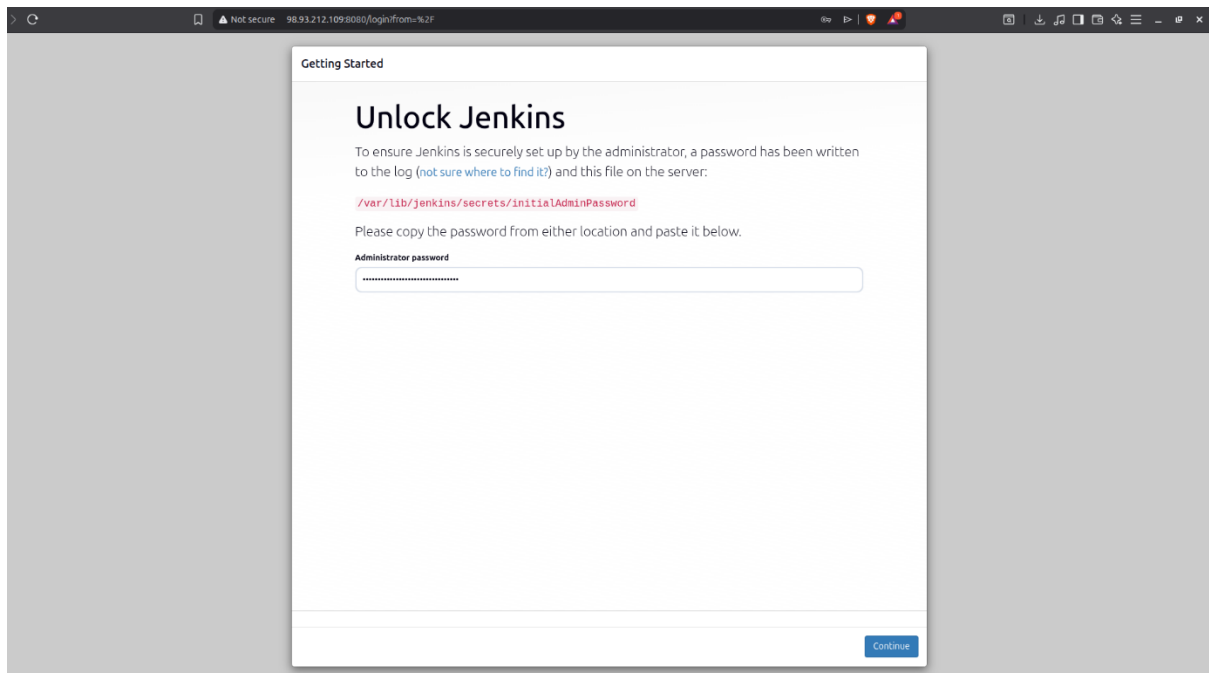
```
ubuntu@Jenkins-Master: ~/.ssh
ubuntu@Jenkins-Master: ~/.ssh$ cd ~/.ssh
ubuntu@Jenkins-Master: ~/.ssh$ ls
authorized_keys  id_ed25519  id_ed25519.pub
ubuntu@Jenkins-Master: ~/.ssh$ cat id_ed25519
-BEGIN OPENSSH PRIVATE KEY-----
bnNzaC1rZktdjEAAAABGSwbnUAAAABbn9uZ0AAAAAABAAAAWAAAAZtc2gtZW
UxOQAAACCrGq3Jhk2UHaA3juzWxgK7YxqzkrKwB8r0w7ALd/SwAAAJhVRG51VURk
AAZtc2gtZWYNTUxOQAAACCrGq3Jhk2UHaA3juzWxgK7YxqzkrKwB8r0w7ALd/Sw
D7Jk3xStU0a1Pu6XXNiTIhwaVcb+/LKu21kySFQgqauqsakmGTZQdoDe07Na/Gart
CSsrDxKvTdsAt39LAAAFxVldW50dUBKZW5raW5zLU1hc3Rlcg==
-END OPENSSH PRIVATE KEY-----
ubuntu@Jenkins-Master: ~/.ssh$ cat id_ed25519.pub
ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIKsaokmGTZQdoDe07Na/GartjGrOCsSrDxKvTdsAt39L ubuntu@Jenkins-Master
ubuntu@Jenkins-Master: ~/.ssh$
```

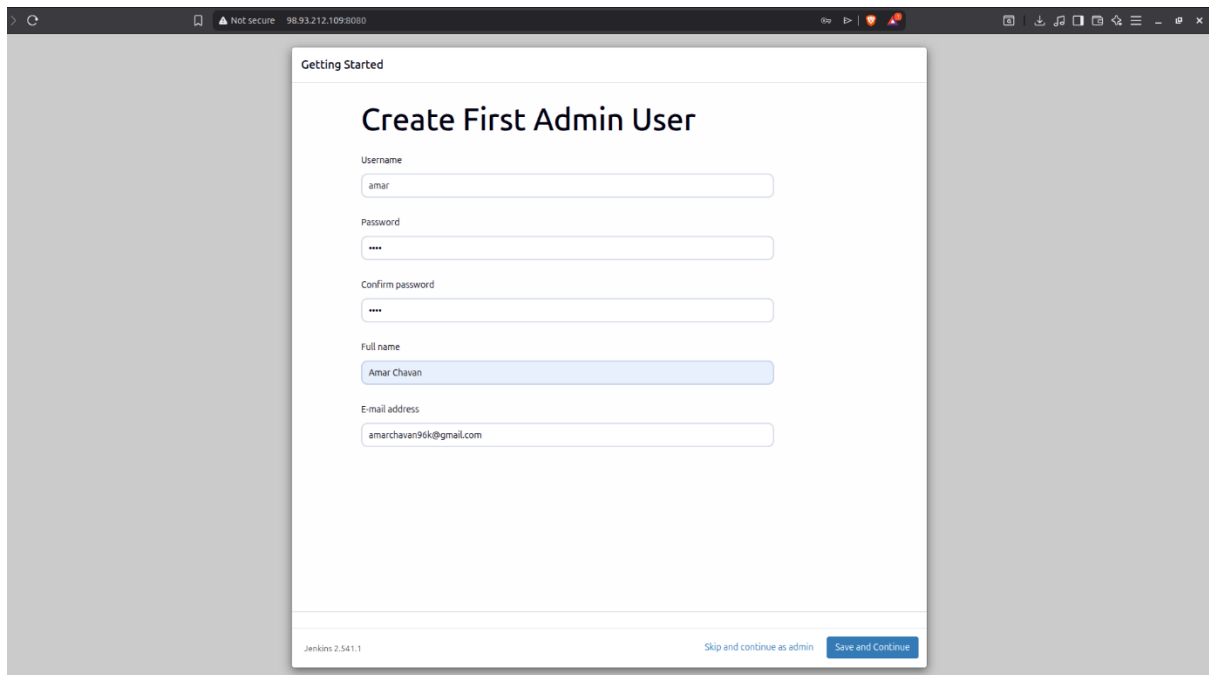
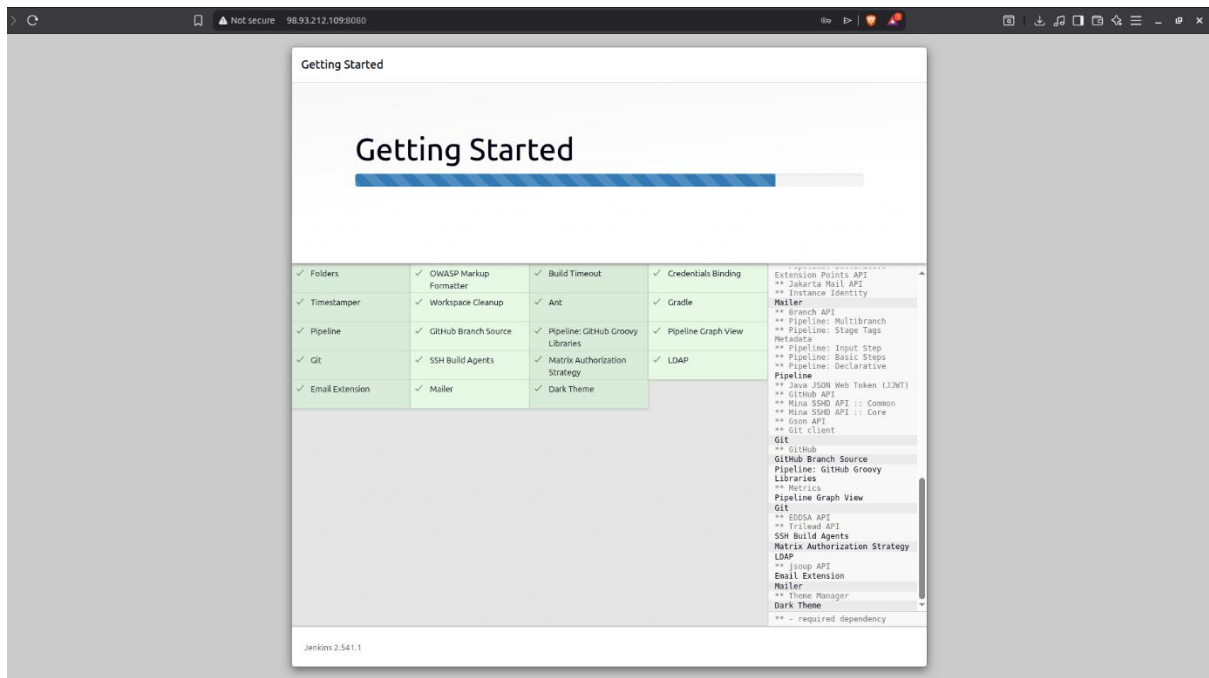
```
ubuntu@Jenkins-Worker-01: /
ubuntu@Jenkins-Worker-01: /$ cd /
ubuntu@Jenkins-Worker-01: /$ nano authorized_keys
ubuntu@Jenkins-Worker-01: /$ #add
ubuntu@Jenkins-Worker-01: /$ #public key of the master node added here for the ssh connection between master node and the worker node
ubuntu@Jenkins-Worker-01: /$
```

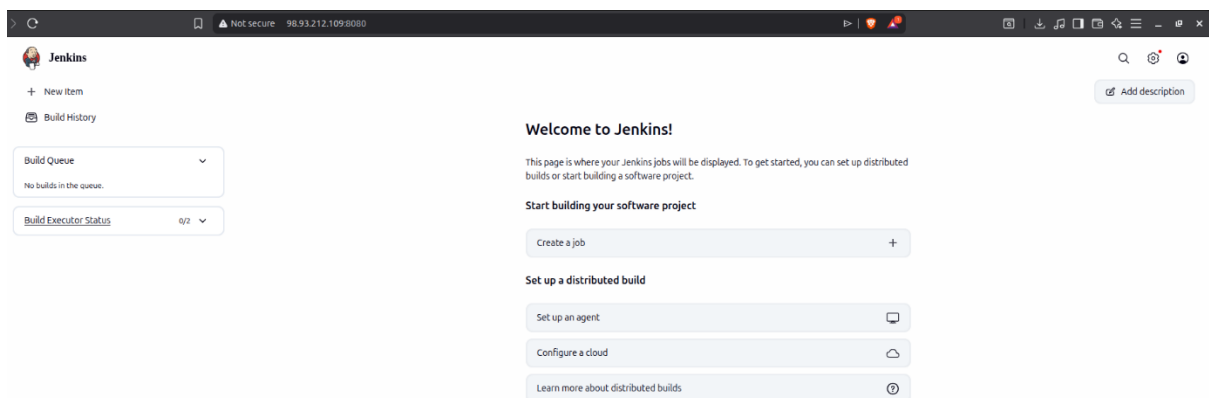
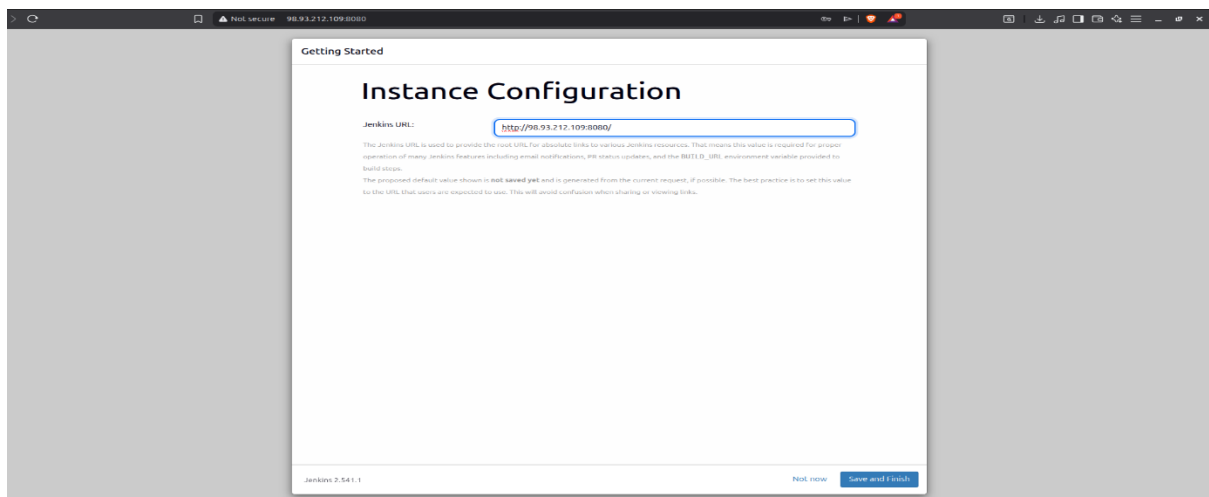
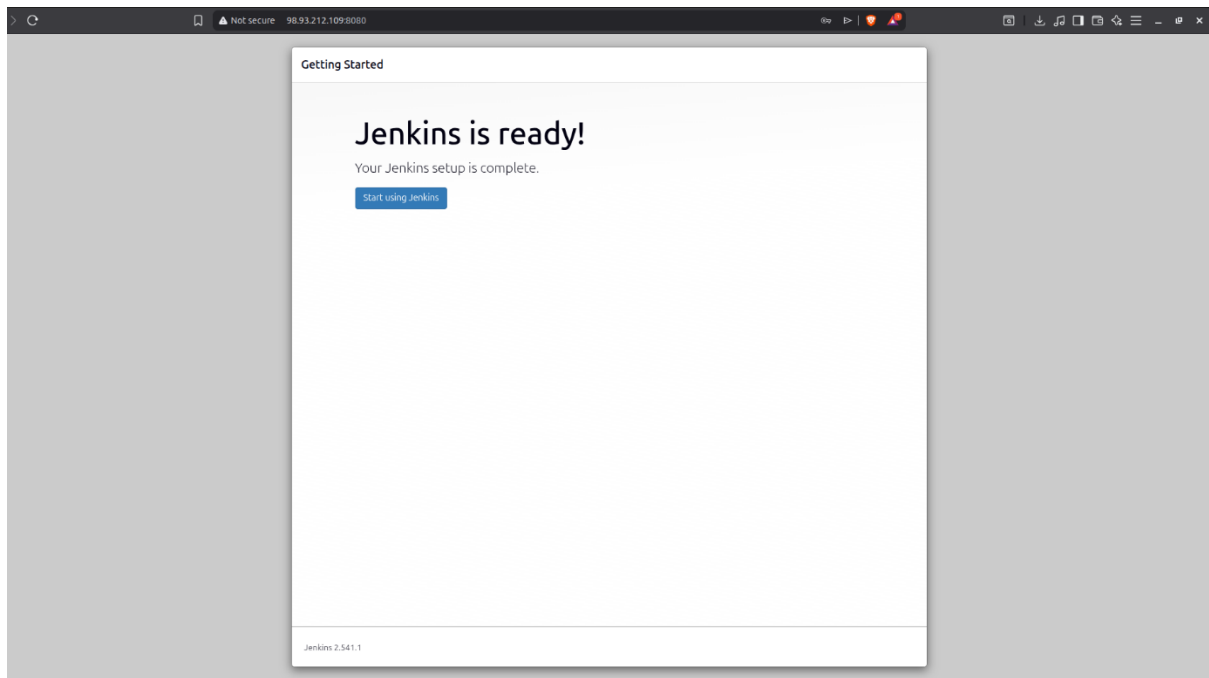
Step 5: Jenkins UI Setup & Node Registration

1. **Access Jenkins:** Open your browser and navigate to `http://<Master-Public-IP>:8080`.
2. **Unlock Jenkins:** Retrieve the initial admin password from the Master terminal: `sudo cat /var/lib/jenkins/secrets/initialAdminPassword`
3. **Install Plugins:** Select "Install Suggested Plugins."

Screenshots of Jenkins UI Setup

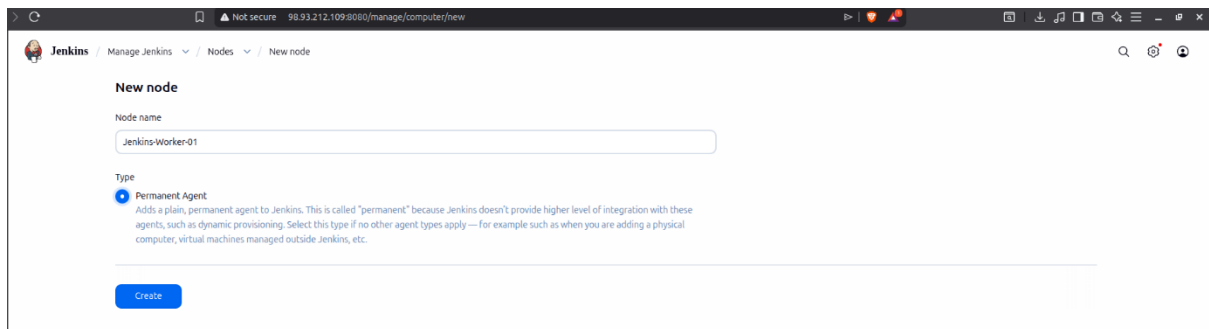




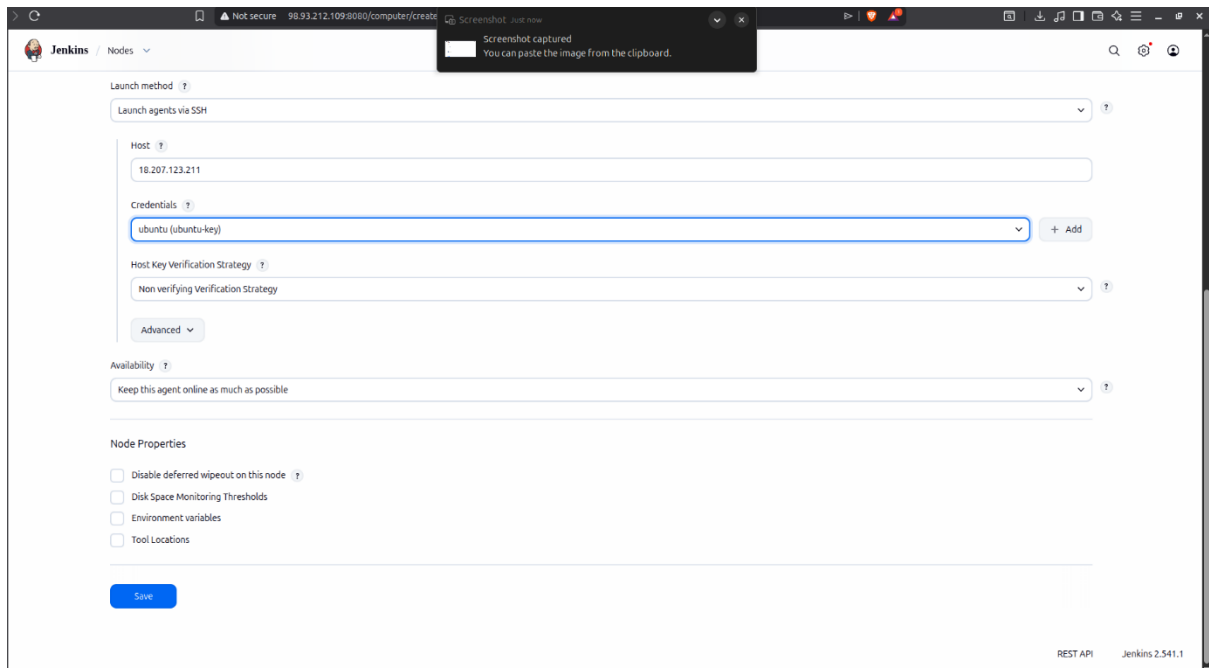


Navigate to **Manage Jenkins > Nodes > New Node**.

- **Node Name:** Jenkins-Worker-01
- **Type:** Permanent Agent
- **Remote Root Directory:** /home/ubuntu (Create this folder on the worker first).
- **Launch Method:** Select "**Launch agents via SSH**".
- **Host:** <Worker-Private-IP>
- Click Save Button and check the logs of the Node



The screenshot shows the Jenkins 'New node' configuration page. The 'Node name' field contains 'Jenkins-Worker-01'. The 'Type' dropdown is set to 'Permanent Agent'. A 'Create' button is visible at the bottom.



The screenshot shows the Jenkins 'Launch agents via SSH' configuration page. The 'Launch method' is set to 'Launch agents via SSH'. The 'Host' field contains '16.207.123.211'. The 'Credentials' dropdown is set to 'ubuntu (ubuntu-key)'. The 'Host key Verification Strategy' is set to 'Non verifying Verification Strategy'. The 'Availability' dropdown is set to 'Keep this agent online as much as possible'. The 'Node Properties' section is expanded, showing options for 'Disable deferred wipeout on this node', 'Disk Space Monitoring Thresholds', 'Environment variables', and 'Tool Locations'. A 'Save' button is visible at the bottom.

Jenkins / Nodes

Name: Jenkins-Worker-01

Description: Jenkins-Worker-01

Plain text: [Preview](#)

Number of executors: 1

Remote root directory:

Labels: Jenkins-Worker-01

Usage: Use this node as much as possible

Launch method: Launch agents via SSH

Host: 18.207.123.211

[Save](#)

Jenkins / Nodes / Jenkins-Worker-01 / Log

```

PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
PIPESTATUS=([0]=0*)
PPID=3906
PS4='+ '
PWD=/home/ubuntu
SHELL=/bin/bash
SHELLOPTS=braceexpand:hashall:interactive-comments
SHLVL=1
SSH_CLIENT='98.93.212.109 40618 22'
SSH_CONNECTION='98.93.212.109 40618 172.31.6.4 22'
TERM=dumb
UID=1000
USER=ubuntu
XDG_RUNTIME_DIR=/run/user/1000
XDG_SESSION_CLASS=user
XDG_SESSION_ID=14
XDG_SESSION_TYPE=ttty
_=1'
[01/23/26 10:20:47] [SSH] Starting sftp client.
[01/23/26 10:20:47] [SSH] Copying latest remoting.jar...
[01/23/26 10:20:47] [SSH] Copied 1,407,915 bytes.
Expanded the channel window size to 4MB
[01/23/26 10:20:47] [SSH] Starting agent process: cd "/home/ubuntu" && java -jar remoting.jar -workDir /home/ubuntu -jar-cache /home/ubuntu/remoting/jarCache
Jan 23, 2026 10:20:47 AM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir
INFO: Using /home/ubuntu/remoting as a remoting work directory
Jan 23, 2026 10:20:47 AM org.jenkinsci.remoting.engine.WorkDirManager setupLogging
INFO: Both error and output logs will be printed to /home/ubuntu/remoting
>channel started
Remoting version: 3352.v17a_fb_4b_2773f
Launcher: SSHLauncher
Communication Protocol: Standard in/out
This is a Unix agent
Agent successfully connected and online
  
```

REST API Jenkins 2.541.1

Important Note on Networking: Although we have two EC2 instances, they are completely independent and "anonymous" to one another. To establish a Master-Worker connection, we use the **Private IP** of the Worker.

- **Why Private IP?** Using the Private IP ensures the communication stays within the AWS internal network (VPC). This is faster, more secure, and free of data transfer costs compared to using the Public IP.
- **The Challenge:** Since the Master doesn't "know" the Worker exists, we provide the Private IP as a direct pointer and the **Private SSH Key** as the digital "security clearance" to bridge the two servers.

- **Credentials:** 1. Click **Add > Jenkins**. 2. Kind: **SSH Username with private key**. 3. Username: ubuntu (or the default user of your OS). 4. Private Key: Select "Enter directly" and paste the content of the **id_rsa** file generated on the Master node.
- **Host Key Verification Strategy:** Select "**Non-verifying Verification Strategy**" (for lab environments) or "Known hosts file".

Conclusion

The successful completion of this assignment demonstrates the fundamental principles of **Distributed Computing** and **Continuous Integration** in a DevOps environment. Through this implementation, we can draw the following conclusions:

- **Scalability & Resource Optimization:** By offloading build and test executions to a Worker node, we ensure that the Jenkins Master remains responsive for administrative tasks and UI management. This prevents the "Single Point of Failure" regarding resource exhaustion.
- **Security through Private Networking:** Utilizing the **Worker's Private IP** for communication highlights a critical security best practice in Cloud Infrastructure. It ensures that internal CI/CD traffic remains within the VPC, reducing exposure to the public internet and eliminating unnecessary data transfer costs.
- **Authentication via SSH Keys:** The use of **RSA Key Pairs** for establishing the handshake between nodes provides a secure, password-less authentication method that is essential for automated systems, removing the need for manual human intervention during build triggers.
- **Architectural Understanding:** This setup provides a practical understanding of how independent cloud instances, which are "anonymous" to each other by default, can be unified into a single, cohesive automation cluster through proper networking and credential management.

Overall, this assignment provides a robust foundation for building complex, production-ready CI/CD pipelines where multiple worker nodes can be scaled horizontally to handle diverse operating systems and high-frequency deployment cycles.