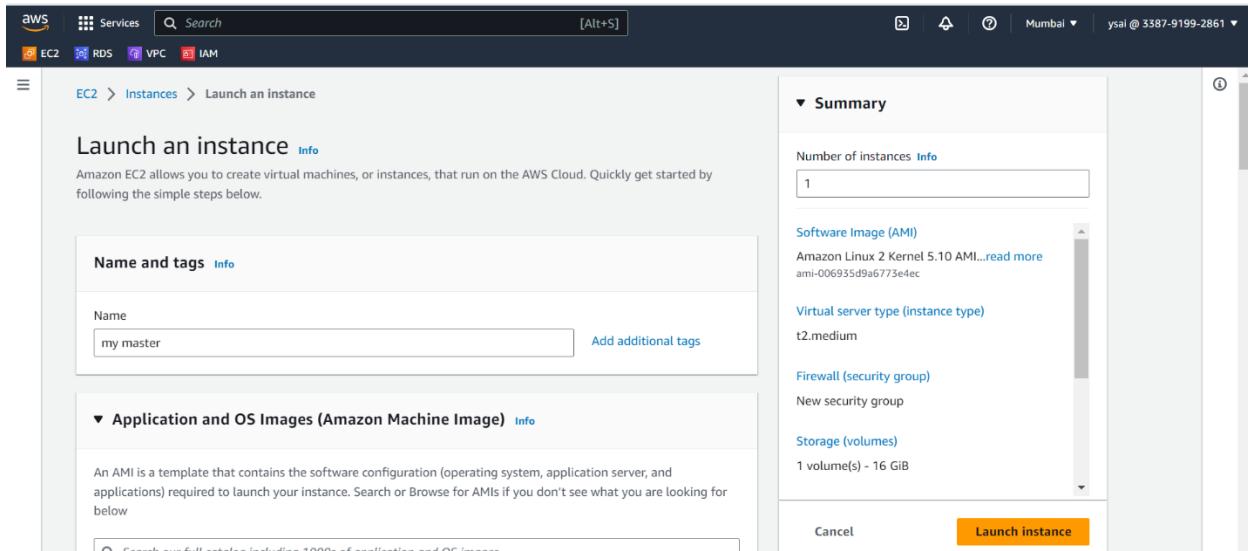
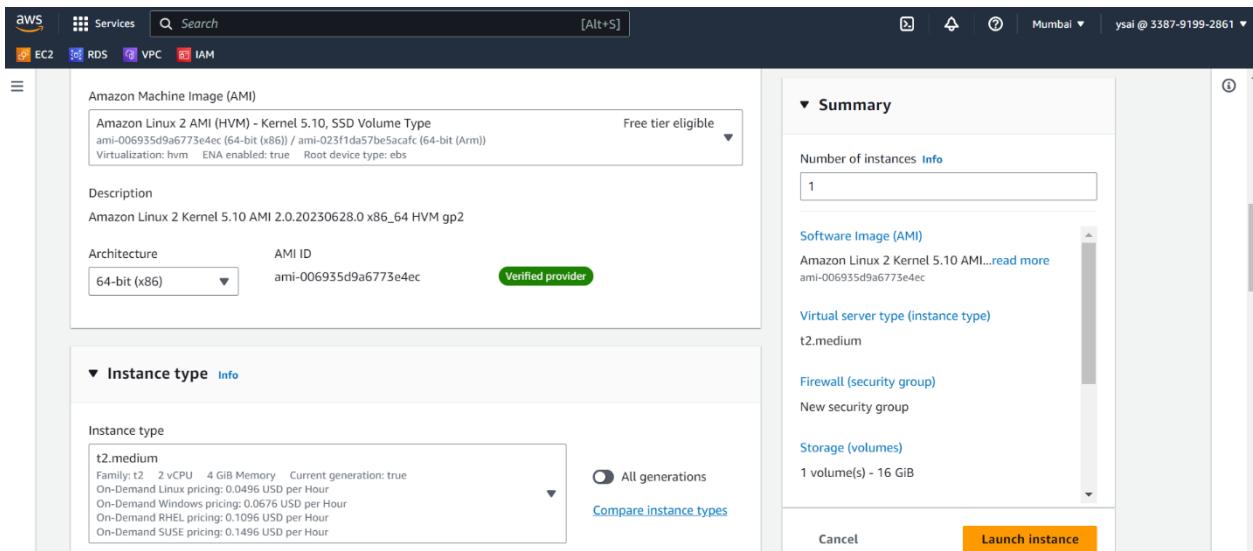


Jenkins Master Slave Configuration

- ❖ Create a EC2 instance and launch a new instance.

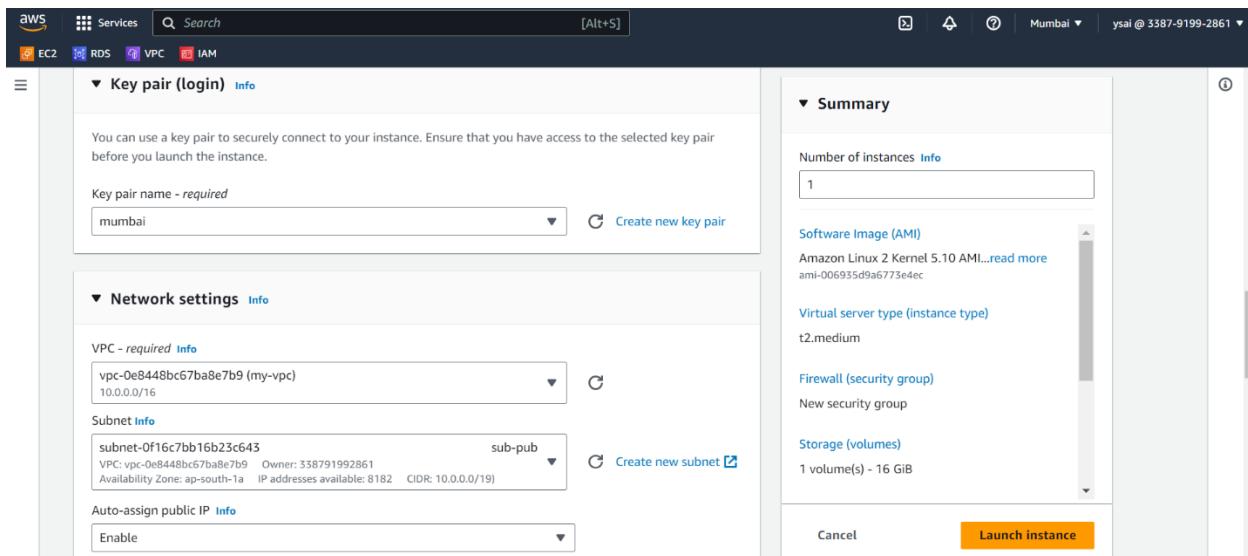


- ❖ Select Amazon Machine Image and Instance type.

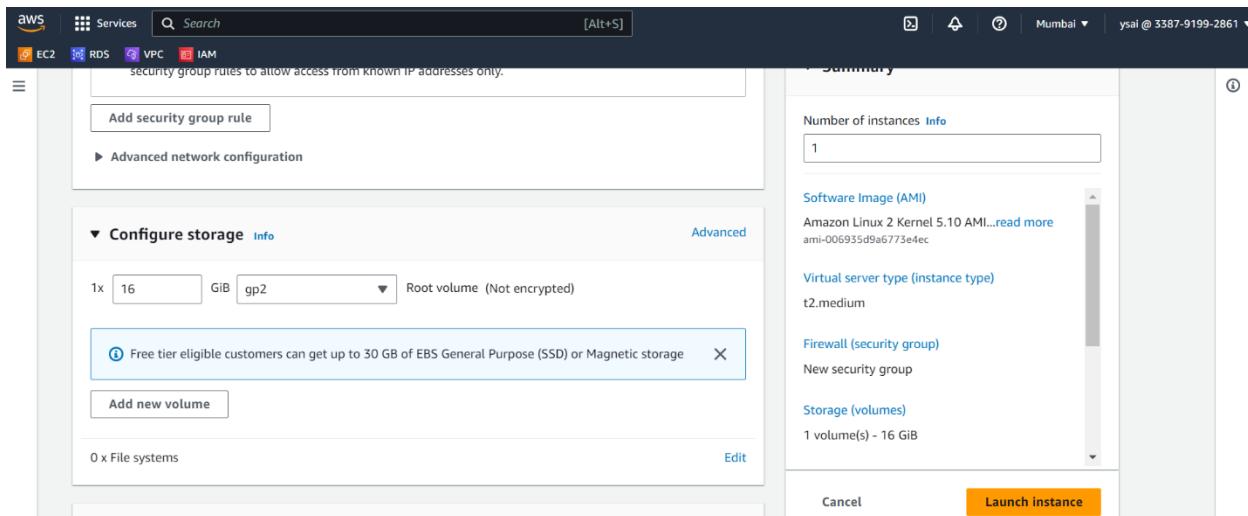


- ❖ Here select the Key pair and network Settings .

Jenkins Master Slave Configuration

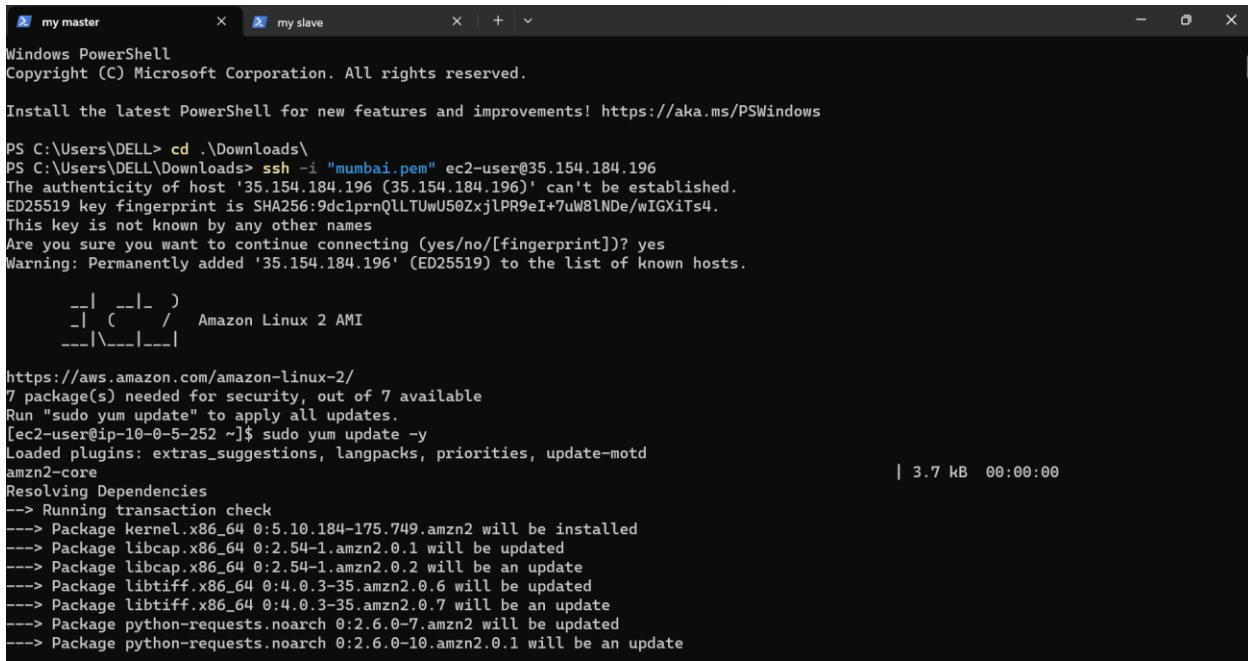


- ❖ Here we are configuring the storage to this instance, for this instance we are taking 16 GiB .



- ❖ Connect your EC2 instance to terminal By using Command “ssh -l “pem.key” ec2-user@ip.address” in master instance .

Jenkins Master Slave Configuration



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

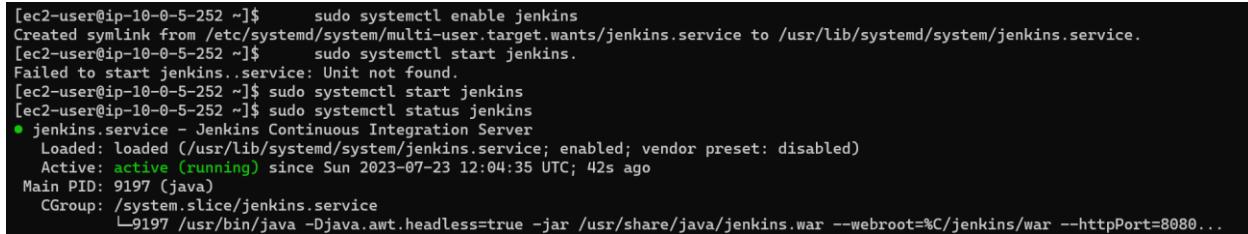
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\DELL> cd .\Downloads\
PS C:\Users\DELL\Downloads> ssh -i "mumbai.pem" ec2-user@35.154.184.196
The authenticity of host '35.154.184.196 (35.154.184.196)' can't be established.
ED25519 key fingerprint is SHA256:9dc1prnQLLTUwU50Zxj1PP9eI+7uW8lNDe/wIGXiTs4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '35.154.184.196' (ED25519) to the list of known hosts.

--| --|- )
-| ( -- / Amazon Linux 2 AMI
---\---|---

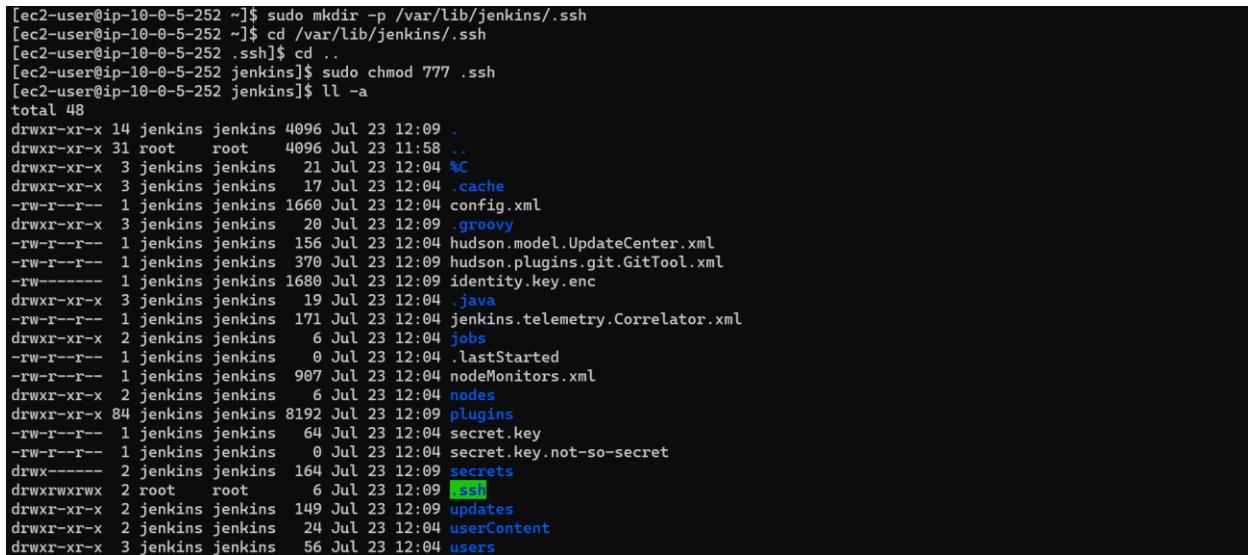
https://aws.amazon.com/amazon-linux-2/
7 package(s) needed for security, out of 7 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-10-0-5-252 ~]$ sudo yum update -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
Resolving Dependencies
--> Running transaction check
--> Package kernel.x86_64 0:5.10.184-175.749.amzn2 will be installed
--> Package libcap.x86_64 0:2.54-1.amzn2.0.1 will be updated
--> Package libcap.x86_64 0:2.54-1.amzn2.0.2 will be an update
--> Package libtiff.x86_64 0:4.0.3-35.amzn2.0.6 will be updated
--> Package libtiff.x86_64 0:4.0.3-35.amzn2.0.7 will be an update
--> Package python-requests.noarch 0:2.6.0-7.amzn2 will be updated
--> Package python-requests.noarch 0:2.6.0-10.amzn2.0.1 will be an update
```

- ❖ In master instance install Jenkins ,enable it and start Jenkins .



```
[ec2-user@ip-10-0-5-252 ~]$ sudo systemctl enable jenkins
Created symlink from /etc/systemd/system/multi-user.target.wants/jenkins.service to /usr/lib/systemd/system/jenkins.service.
[ec2-user@ip-10-0-5-252 ~]$ sudo systemctl start jenkins
Failed to start jenkins.service: Unit not found.
[ec2-user@ip-10-0-5-252 ~]$ sudo systemctl start jenkins
[ec2-user@ip-10-0-5-252 ~]$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; vendor preset: disabled)
     Active: active (running) since Sun 2023-07-23 12:04:35 UTC; 42s ago
       Main PID: 9197 (java)
      CGroup: /system.slice/jenkins.service
             └─9197 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=%C/jenkins/war --httpPort=8080...
```

- ❖ Here we are creating directory to do configuration and in that by using CD command to go that path and then permission to .ssh as 777 permissions.



```
[ec2-user@ip-10-0-5-252 ~]$ sudo mkdir -p /var/lib/jenkins/.ssh
[ec2-user@ip-10-0-5-252 ~]$ cd /var/lib/jenkins/.ssh
[ec2-user@ip-10-0-5-252 .ssh]$ cd ..
[ec2-user@ip-10-0-5-252 jenkins]$ sudo chmod 777 .ssh
[ec2-user@ip-10-0-5-252 jenkins]$ ll -a
total 48
drwxr-xr-x 14 jenkins jenkins 4096 Jul 23 12:09 .
drwxr-xr-x 31 root root 4096 Jul 23 11:58 ..
drwxr-xr-x 3 jenkins jenkins 21 Jul 23 12:04 %C
drwxr-xr-x 3 jenkins jenkins 17 Jul 23 12:04 .cache
-rw-r--r-- 1 jenkins jenkins 1660 Jul 23 12:04 config.xml
drwxr-xr-x 3 jenkins jenkins 20 Jul 23 12:09 .groovy
-rw-r--r-- 1 jenkins jenkins 156 Jul 23 12:04 hudson.model.UpdateCenter.xml
-rw-r--r-- 1 jenkins jenkins 370 Jul 23 12:09 hudson.plugins.git.GitTool.xml
-rw----- 1 jenkins jenkins 1680 Jul 23 12:09 identity.key.enc
drwxr-xr-x 3 jenkins jenkins 19 Jul 23 12:04 .java
-rw-r--r-- 1 jenkins jenkins 171 Jul 23 12:04 jenkins.telemetry.Correlator.xml
drwxr-xr-x 2 jenkins jenkins 6 Jul 23 12:04 jobs
-rw-r--r-- 1 jenkins jenkins 0 Jul 23 12:04 .lastStarted
-rw-r--r-- 1 jenkins jenkins 907 Jul 23 12:04 nodeMonitors.xml
drwxr-xr-x 2 jenkins jenkins 6 Jul 23 12:04 nodes
drwxr-xr-x 84 jenkins jenkins 8192 Jul 23 12:09 plugins
-rw-r--r-- 1 jenkins jenkins 64 Jul 23 12:04 secret.key
-rw-r--r-- 1 jenkins jenkins 0 Jul 23 12:04 secret.key.not-so-secret
drwx----- 2 jenkins jenkins 164 Jul 23 12:09 secrets
drwxrwxrwx 2 root root 6 Jul 23 12:09 .ssh
drwxr-xr-x 2 jenkins jenkins 149 Jul 23 12:09 updates
drwxr-xr-x 2 jenkins jenkins 24 Jul 23 12:04 userContent
drwxr-xr-x 3 jenkins jenkins 56 Jul 23 12:04 users
```

- ❖ Change directory to .ssh, in that directory we have to give the command.
“Ssh-keyscan -H ip.address >>/var/lib/Jenkins/.ssh/known_hosts”

Jenkins Master Slave Configuration

- ❖ Change the owner to Jenkins for known_hosts.

```
[ec2-user@ip-10-0-5-252 jenkins]$ cd .ssh
[ec2-user@ip-10-0-5-252 .ssh]$ ll -a
total 4
drwxrwxrwx 2 root root 6 Jul 23 12:09 .
drwxr-xr-x 14 jenkins jenkins 4096 Jul 23 12:09 ..
[ec2-user@ip-10-0-5-252 .ssh]$ sudo ssh-keyscan -H 10.0.16.102 >>/var/lib/jenkins/.ssh/known_hosts
[bash: /var/lib/jenkins/.ssh/known_hosts: No such file or directory
[ec2-user@ip-10-0-5-252 .ssh]$ sudo ssh-keyscan -H 10.0.16.102 >>/var/lib/jenkins/.ssh/known_hosts
# 10.0.16.102:22 SSH-2.0-OpenSSH_7.4
# 10.0.16.102:22 SSH-2.0-OpenSSH_7.4
# 10.0.16.102:22 SSH-2.0-OpenSSH_7.4
[ec2-user@ip-10-0-5-252 .ssh]$ sudo chown jenkins:jenkins known_hosts
[ec2-user@ip-10-0-5-252 .ssh]$ ll -a
total 8
drwxrwxrwx 2 root root 25 Jul 23 12:14 .
drwxr-xr-x 14 jenkins jenkins 4096 Jul 23 12:09 ..
-rw-rw-r-- 1 jenkins jenkins 806 Jul 23 12:14 known_hosts
[ec2-user@ip-10-0-5-252 .ssh]$ sudo chmod 700 known_hosts
[ec2-user@ip-10-0-5-252 .ssh]$ ll -a
total 8
drwxrwxrwx 2 root root 25 Jul 23 12:14 .
drwxr-xr-x 14 jenkins jenkins 4096 Jul 23 12:09 ..
-rw-rw-r-- 1 jenkins jenkins 806 Jul 23 12:14 known_hosts
[ec2-user@ip-10-0-5-252 .ssh]$ cd
[ec2-user@ip-10-0-5-252 ~]$ cd /etc
[ec2-user@ip-10-0-5-252 etc]$ sudo vi sudoers
[ec2-user@ip-10-0-5-252 etc]$ cd /var/lib/jenkins/.ssh
[ec2-user@ip-10-0-5-252 .ssh]$ ls
known_hosts
```

- ❖ Before master instance we have to create slave instance . In slave instance we have to update.

```
PS C:\Users\DELL\Downloads> ssh -i "mumbai.pem" ec2-user@13.233.24.202
The authenticity of host '13.233.24.202' can't be established.
ED25519 key fingerprint is SHA256:AuE6dpXyPqRE7CPYaoTMq91LJ78EjCQTSXc9653ktaI.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '13.233.24.202' (ED25519) to the list of known hosts.

__| __|_
_| ( _ / Amazon Linux 2 AMI
___\_\_\_\_|

https://aws.amazon.com/amazon-linux-2/
7 package(s) needed for security, out of 7 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-10-0-16-102 ~]$ sudo yum update -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
Resolving Dependencies
--> Running transaction check
--> Package kernel.x86_64 0:5.10.184-175.749.amzn2 will be installed
--> Package libcap.x86_64 0:2.54-1.amzn2.0.1 will be updated
--> Package libcap.x86_64 0:2.54-1.amzn2.0.2 will be an update
--> Package libtiff.x86_64 0:4.0.3-35.amzn2.0.6 will be updated
--> Package libtiff.x86_64 0:4.0.3-35.amzn2.0.7 will be an update
--> Package python-requests.noarch 0:2.6.0-7.amzn2 will be updated
--> Package python-requests.noarch 0:2.6.0-10.amzn2.0.1 will be an update
--> Package python2-rsa.noarch 0:3.4.1-1.amzn2.0.3 will be updated
--> Package python2-rsa.noarch 0:3.4.1-1.amzn2.0.4 will be an update
--> Package python3-pip.noarch 0:20.2.2-1.amzn2.0.3 will be updated
--> Package python3-pip.noarch 0:20.2.2-1.amzn2.0.4 will be an update
--> Package tcpcdump.x86_64 14:4.9.2-4.amzn2.1 will be updated
```

- ❖ In slave instance we have to create user and password to that user.

```
[ec2-user@ip-10-0-16-102 ~]$ sudo useradd myslave
[ec2-user@ip-10-0-16-102 ~]$ sudo passwd myslave
Changing password for user myslave.
New password:
BAD PASSWORD: The password fails the dictionary check - it is too simplistic/systematic
Retype new password:
passwd: all authentication tokens updated successfully.
[ec2-user@ip-10-0-16-102 ~]$ sudo vi /etc/ssh/sshd_config
[ec2-user@ip-10-0-16-102 ~]$ sudo service sshd restart
sudo: service: command not found
[ec2-user@ip-10-0-16-102 ~]$ sudo service sshd restart
Redirecting to /bin/systemctl restart sshd.service
[ec2-user@ip-10-0-16-102 ~]$ exit
logout
```

- ❖ To login into user instance we have to use this command.

Jenkins Master Slave Configuration

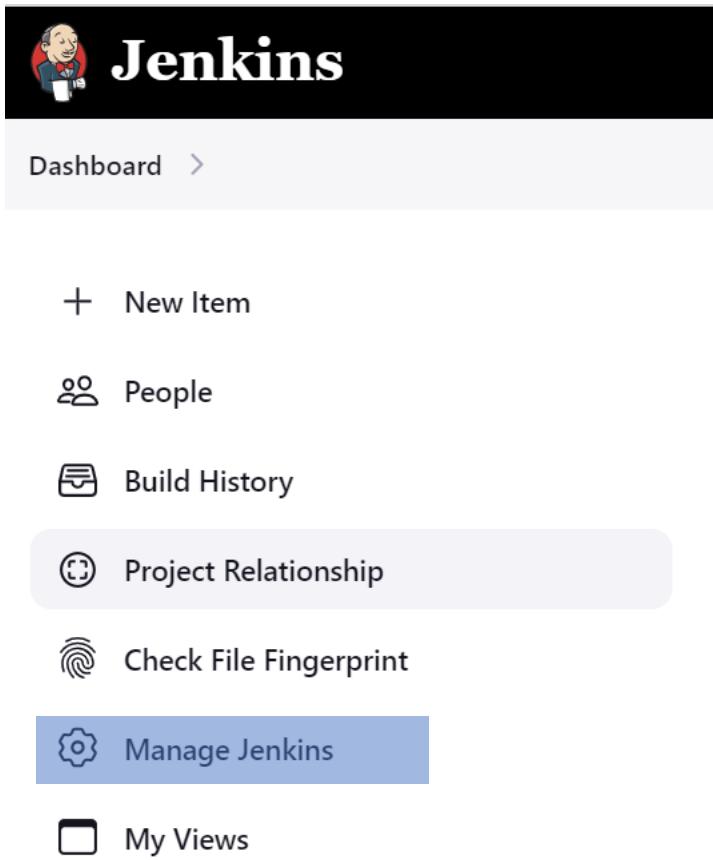
Command : ssh [username @ip.address](#) this is the command to login.

```
PS C:\Users\DELL\Downloads> ssh myslave@13.233.24.202
myslave@13.233.24.202's password:
--| --|_
_|| ( _ /   Amazon Linux 2 AMI
---\_\_|_|
https://aws.amazon.com/amazon-linux-2/
```

- ❖ Now generating public/private rsa key pair with help of command “**ssh-keygen**” and cd .ssh
- ❖ cat id_rsa.pub > authorized_keys
- ❖ chmod 700 authorized_keys .

```
[myslave@ip-10-0-16-102 ~]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/myslave/.ssh/id_rsa):
Created directory '/home/myslave/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/myslave/.ssh/id_rsa.
Your public key has been saved in /home/myslave/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:IGYkIIWdrHEV5IxM03+tT5IsMDIVreaxdmY7UA4GFCg myslave@ip-10-0-16-102.ap-south-1.compute.internal
The key's randomart image is:
+---[RSA 2048]---+
|x%o=+
|Eo== .
|oBo * . .
|.o @ + o .
| * 0 o S
|= * = .
|= o +
|o .
+---[SHA256]---+
```

- ❖ Login to Jenkins console and select manage Jenkins .



Jenkins Master Slave Configuration

- ❖ In Manage Jenkins go to Nodes and Cloud.

The screenshot shows the Jenkins 'Manage Jenkins' page. On the left, there's a sidebar with links like 'New Item', 'People', 'Build History', 'Project Relationship', 'Check File Fingerprint', 'Manage Jenkins' (which is selected and highlighted in grey), and 'My Views'. Below these are dropdown menus for 'Build Queue' (No builds in the queue) and 'Build Executor Status'. The main content area is titled 'Manage Jenkins' and contains a yellow warning box: 'Building on the built-in node can be a security issue. You should set the number of executors on the built-in node to 0. See [the documentation](#)'. Below this is the 'System Configuration' section, which includes 'System' (Configure global settings and paths), 'Tools' (Configure tools, their locations and automatic installers), 'Plugins' (Add, remove, disable or enable plugins that can extend the functionality of Jenkins), and 'Nodes and Clouds' (Add, remove, control and monitor the various nodes that Jenkins runs jobs on). A search bar at the top right says 'Search settings'.

- ❖ Create a node by configuring as like below

The screenshot shows the Jenkins 'Configure' page for a node named 'myslave'. The left sidebar lists options: 'Status', 'Delete Agent', 'Configure' (which is selected and highlighted in grey), 'Build History', 'Load Statistics', 'Script Console', 'Log', 'System Information', and 'Disconnect'. The main form has fields for 'Name' (set to 'myslave'), 'Description' (set to 'myslave'), and 'Number of executors' (set to '6'). There are also buttons for '[Plain text] Preview' and a question mark icon for help. At the top right, there's a search bar ('Search (CTRL+K)'), a user icon ('ysai'), and a 'log out' link.

Jenkins Master Slave Configuration

- ❖ Give the remote root directory and label and we have to select the launch method.

The screenshot shows the 'Configure' page for the 'myslave' node. On the left, there is a sidebar with a list of nodes: 1 Idle, 2 Idle, 3 Idle, 4 Idle, 5 Idle, and 6 Idle. The main configuration area includes:

- Remote root directory**: /home/myslave
- Labels**: sai
- Usage**: Use this node as much as possible
- Launch method**: Launch agents via SSH
- Host**: 10.0.16.102

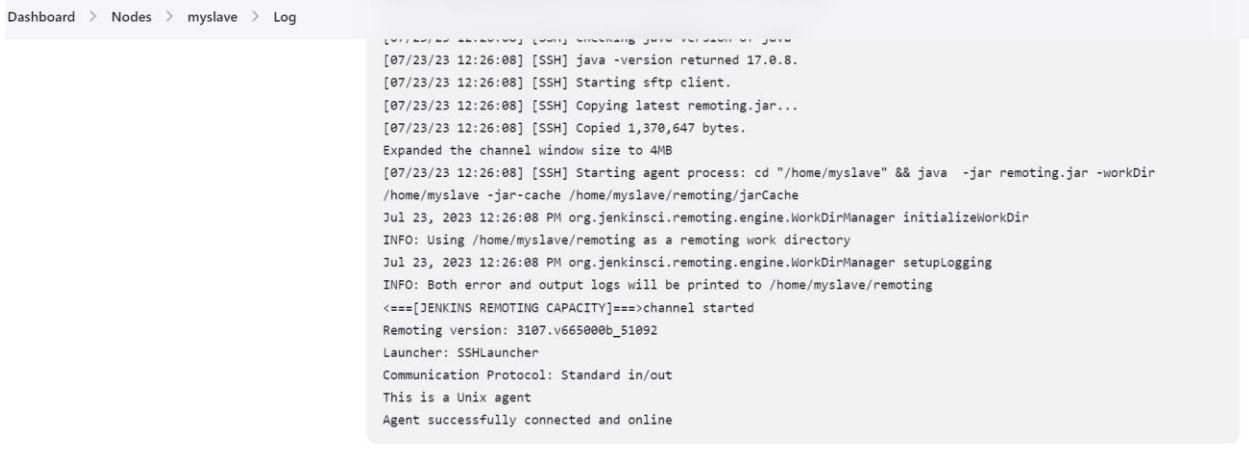
- ❖ Here we selecting the host id of the slave and host key verification as a verification strategy and save the conigure.

The screenshot shows the 'Configure' page for the 'myslave' node, focusing on advanced configuration. It includes:

- Host**: 10.0.16.102
- Credentials**: myslave (myslave)
- Add** button (highlighted in blue)
- Host Key Verification Strategy**: Known hosts file Verification Strategy
- Availability**: Keep this agent online as much as possible
- Save** button (highlighted in blue)

Jenkins Master Slave Configuration

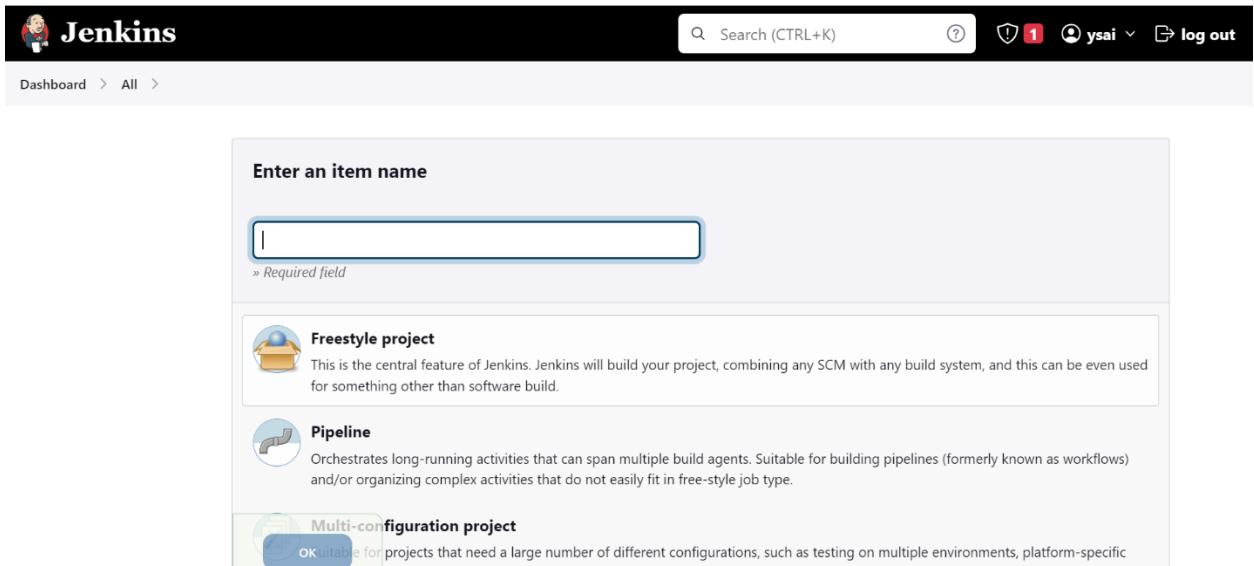
- ❖ After saving the configure go and check in the log console weather the agent is successful or not.



The screenshot shows the Jenkins log console for a node named 'myslave'. The log output is as follows:

```
[07/23/23 12:26:08] [SSH] java -version returned 17.0.8.
[07/23/23 12:26:08] [SSH] Starting sftp client.
[07/23/23 12:26:08] [SSH] Copying latest remoting.jar...
[07/23/23 12:26:08] [SSH] Copied 1,370,647 bytes.
Expanded the channel window size to 4MB
[07/23/23 12:26:08] [SSH] Starting agent process: cd "/home/myslave" && java -jar remoting.jar -workDir /home/myslave -jar-cache /home/myslave/remoting/jarCache
Jul 23, 2023 12:26:08 PM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir
INFO: Using /home/myslave/remoting as a remoting work directory
Jul 23, 2023 12:26:08 PM org.jenkinsci.remoting.engine.WorkDirManager setupLogging
INFO: Both error and output logs will be printed to /home/myslave/remoting
<===[JENKINS REMOTING CAPACITY]==>channel started
Remoting version: 3107.v665800b_51092
Launcher: SSHLauncher
Communication Protocol: Standard in/out
This is a Unix agent
Agent successfully connected and online
```

- ❖ Now, create a job and select the free style project to do config and automation.



The screenshot shows the Jenkins dashboard with the search bar set to 'Search (CTRL+K)'. Below the search bar, there are notifications for 1 new item and a user named 'ysai'. The main content area displays three project types:

- 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.
- Multi-configuration project**: OK suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific

- ❖ In configuration give your description of the project what you are going to do and select the label expression.

Jenkins Master Slave Configuration

The screenshot shows the Jenkins configuration page for a project named "project2". The "General" tab is selected. In the "Description" field, "project2" is entered. Under "Build Triggers", the "Restrict where this project can be run" option is checked, and the "Label Expression" is set to "sai". There are also other build trigger options like "Discard old builds", "GitHub project", "This project is parameterized", "Throttle builds", and "Execute concurrent builds if necessary". At the bottom are "Save" and "Apply" buttons.

- ❖ Select the git hub repository URL with credentials (it requires only when it is private repository).

The screenshot shows the Jenkins configuration page for a project named "project2". The "Source Code Management" tab is selected, and "Git" is chosen. In the "Repositories" section, a single repository is defined with the "Repository URL" as "https://github.com/saiyerra831/tf-abalone.git" and the "Credentials" dropdown containing "saiyerra831/*****". There is also an "Advanced" section and a "Add Repository" button.

- ❖ In build steps give the related commands of your project.

Jenkins Master Slave Configuration

Dashboard > project2 > Configuration

Configure

Build Steps

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Execute shell ?

Command

See [the list of available environment variables](#)

```
#!/bin/bash
sudo yum install -y yum-utils shadow-utils
sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
sudo yum -y install terraform
cd /tf-abalone
sudo terraform init
sudo terraform validate
sudo terraform apply --auto-approve
```

Save

Apply

- ❖ Here do apply and save the configuration and do BUILD NOW then see the console output.

Status

</> Changes

Console Output

View as plain text

Edit Build Information

Delete build '#7'

Git Build Data

Previous Build

Console Output

```
Started by user ysai
Running as SYSTEM
Building remotely on myslave (sai) in workspace /home/myslave/workspace/project2
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Deferred wipeout is used...
[WS-CLEANUP] Done
The recommended git tool is: NONE
using credential 9065ea84-9083-414f-8901-8a1e566b3cc6
Cloning the remote Git repository
Cloning repository https://github.com/saiyerra831/tf-abalone.git
> git init /home/myslave/workspace/project2 # timeout=10
Fetching upstream changes from https://github.com/saiyerra831/tf-abalone.git
> git --version # timeout=10
> git --version # 'git version 2.40.1'
using GIT_ASKPASS to set credentials
> git fetch --tags --force --progress -- https://github.com/saiyerra831/tf-abalone.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/saiyerra831/tf-abalone.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
+ git fetch --tags --force --progress -- https://github.com/saiyerra831/tf-abalone.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
```

- ❖ Go to EC2 console and select the instance and copy the instance Ip address.

Jenkins Master Slave Configuration

The screenshot shows the AWS EC2 Instances page. A single instance, 'tf-ec1' (i-067b67b49a2a22481), is listed as 'Running' with the instance type 't2.micro'. The instance has a public IPv4 address of 54.88.66.164 and a private IP of 192.168.100.96. The status check is 'Initializing' and there are no alarms. The instance is located in the 'us-east-1a' availability zone.

❖ Here is the output of the abalone age prediction by using master and slave configuration.

The screenshot shows a web browser window titled 'Abalone-Age-Prediction-Yshu'. The main content area displays a video player with the title 'What is an Abalone?' and a question mark icon. Below the video, the text 'What is abalone?' is visible. To the right, there is a section titled 'Sample Image Of Abalone' featuring a large, colorful abalone shell with the text 'abalone Shell' written above it. A descriptive paragraph about the shell's healing properties is also present.

Master and slave configuration using different accounts, Here I am the master instance and there are five members in my team, they did their task as like slave instance.

1. Satya's slave configuration
2. Sangita slave configuration
3. Ajjal slave configuration
4. Jagadeesh slave configuration
5. John slave configuration.

Satya's configuration :

Jenkins Master Slave Configuration

- ❖ In this satya created one ec2 instance and created one user in that instance and shared to me.

The screenshot shows the AWS EC2 Instances page. There are four instances listed:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Available
publicec2-py	i-09b984977785cc41	Terminated	t2.medium	-	No alarms	+ ap-sout
publicec2-py	i-06b36030102d0f82a	Running	t2.medium	2/2 checks passed	No alarms	+ ap-sout
publicec2-py	i-0ae5a2698a4cedbe0	Terminated	t2.medium	-	No alarms	+ ap-sout
publicec2-py	i-05e7c5e85df5df666	Terminated	t2.medium	-	No alarms	+ ap-sout

Detailed view for instance i-06b36030102d0f82a (publicec2-py):

- Instance summary:
 - Instance ID: i-06b36030102d0f82a (publicec2-py)
 - Public IPv4 address: 43.205.238.37 | [open address](#)
 - Private IPv4 addresses: 10.0.1.199
 - Public IPv4 DNS: -
- Instance state: Running
- Hostname type: IP name: ip-10-0-1-199.ap-south-1.compute.internal
- Private IP DNS name (IPv4 only): ip-10-0-1-199.ap-south-1.compute.internal

- ❖ Here I added satya slave in my manage Jenkins = Nodes.

The screenshot shows the Jenkins Node configuration page for the node 'satya'.

Node navigation path: Dashboard > Nodes > satya

Actions available for the node:

- Status
- Delete Agent
- Configure
- Build History
- Load Statistics
- Log

Jenkins Master Slave Configuration

- ❖ Created a new project by using satya slave as a satya-project

The screenshot shows the Jenkins configuration interface for a project named 'satya-project'. The 'General' tab is selected. In the 'Description' field, the text 'satya-project' is entered. Below the description, there are several checkboxes: 'Discard old builds', 'GitHub project', 'This project is parameterized', and 'Throttle builds'. At the bottom of the page are 'Save' and 'Apply' buttons.

Configure

General

Enabled

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Description

satya-project

[Plain text] [Preview](#)

- Discard old builds
- GitHub project
- This project is parameterized
- Throttle builds

Save

Apply

- ❖ Here I mentioned in my configuration where to run this project. I gave Satya's slave label to identify the user.

Dashboard > satya-project > Configuration

Configure

General

- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

- Discard old builds
- GitHub project
- This project is parameterized
- Throttle builds
- Execute concurrent builds if necessary
- Restrict where this project can be run

Label Expression

satya

Label **satya** matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.

Advanced ▾

- ❖ In build environment I mention that delete workspace creating the project.

Dashboard > satya-project > Configuration

Configure

Build Environment

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

- Delete workspace before build starts

Advanced ▾

- Use secret text(s) or file(s)
- Add timestamps to the Console Output
- Inspect build log for published build scans
- Terminate a build if it's stuck
- With Ant

Jenkins Master Slave Configuration

- ❖ Here I gave commands in execute shell to do automation.

The screenshot shows the Jenkins project configuration interface for a 'satya-project'. On the left, a sidebar lists 'Configure' sections: General, Source Code Management, Build Triggers, Build Environment, Build Steps (which is selected and highlighted in grey), and Post-build Actions. The main panel is titled 'Execute shell' and contains a command box with the following script:

```
#!/bin/bash
sudo yum install git -y
sudo yum install -y yum-utils shadow-utils
sudo yum-config-manager --add-repo https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
sudo yum -y install terraform
git clone https://Satyavenkat2813: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@github.com/Satyavenkat2813/
cd /home/satya/workspace/satya-project/infra
sudo terraform init
sudo terraform apply --auto-approve
```

Below the command box are 'Save' and 'Apply' buttons. At the bottom of the main panel is an 'Advanced' dropdown menu.

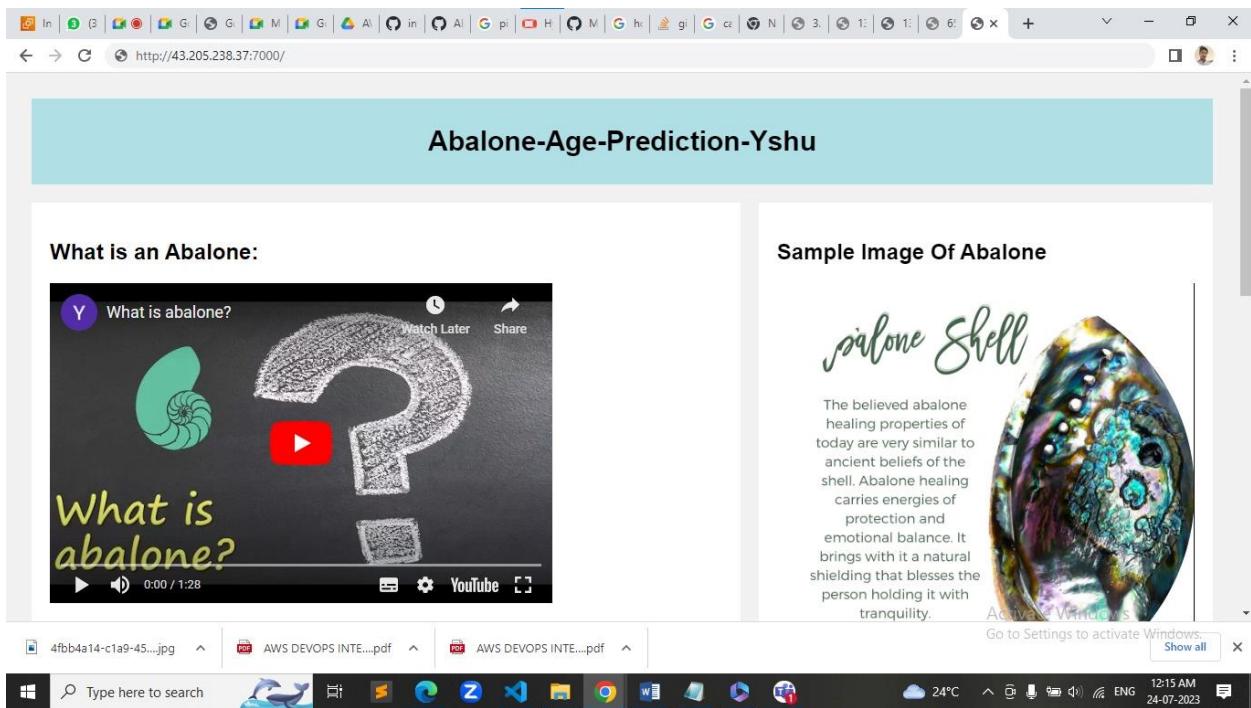
- ❖ After configuring I made build now and checked weather the console output is successful or not.

The screenshot shows the Jenkins console output for build #15 of the 'satya-project'. The top navigation bar includes links for Dashboard, satya-project, #15, and Console Output, along with user information for 'ysai' and a 'log out' button. The main content area is titled 'Console Output' with a green checkmark icon. The left sidebar provides options for Status, Changes, Console Output (which is selected and highlighted in grey), View as plain text, Edit Build Information, Delete build '#15', and Previous Build. The right panel displays the build logs:

```
Started by user ysai
Running as SYSTEM
Building remotely on satya in workspace /home/satya/workspace/satya-project
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Deferred wipeout is used...
[WS-CLEANUP] Done
[satya-project] $ /bin/bash /tmp/jenkins16224450324668508612.sh
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Package git-2.40.1-1.amzn2.0.1.x86_64 already installed and latest version
Nothing to do
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Package yum-utils-1.1.31-46.amzn2.0.1.noarch already installed and latest version
Package 2:shadow-utils-4.1.5.1-24.amzn2.0.2.x86_64 already installed and latest version
Nothing to do
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
```

Output :

Jenkins Master Slave Configuration



Sangita's slave configuration :

- ❖ In this sangita created one ec2 instance and created one user in that instance and shared to me

Instances (1/1) [Info](#)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Ava
sangita	i-0e7d90f5d9cc70e8d	Running	t2.medium	2/2 checks passed	No alarms	us-e

Instance: i-0e7d90f5d9cc70e8d (sangita)

Details	Security	Networking	Storage	Status checks	Monitoring	Tags	
Instance summary Info							
Instance ID i-0e7d90f5d9cc70e8d (sangita)	Public IPv4 address 18.218.69.233 open address	Private IPv4 addresses 172.31.1.77					

Jenkins Master Slave Configuration

- ❖ Here I added Sangita slave in my manage Jenkins = Nodes.

The screenshot shows the Jenkins interface for managing nodes. At the top, there's a navigation bar with the Jenkins logo and the word "Jenkins". Below it, the URL "Dashboard > Nodes > sangita" is visible. On the left, a sidebar contains several options: "Status" (selected), "Delete Agent", "Configure", "Build History", "Load Statistics", and "Log". Under "Configure", a section titled "Build Executor Status" is expanded, showing a dropdown menu. The main content area is currently empty, indicating no build configurations for this node.

- ❖ Created a new project by using Sangita slave as a Sangita-project.

The screenshot shows the Jenkins interface for configuring a new project. The top navigation bar includes the Jenkins logo, a search bar ("Search (CTRL+K)"), and user information ("ysai"). The URL "Dashboard > sangita-project > Configuration" is shown. The main content area is divided into two tabs: "Configure" (selected) and "General". The "General" tab contains fields for "Description" (set to "sangita-project") and several checkboxes: "Discard old builds", "GitHub project", "This project is parameterized", and "Throttle builds". Both the "Save" and "Apply" buttons are visible at the bottom. A "Enabled" switch is also present in the top right of the "General" tab.

Jenkins Master Slave Configuration

- ❖ Here I mentioned in my configuration where to run this project. I gave Sangita's slave label to identify the user.

The screenshot shows the 'Configuration' page for a Jenkins project named 'sangita-project'. In the 'General' section, there is a checkbox labeled 'Restrict where this project can be run' which is checked. Below it, a 'Label Expression' field contains the value 'sangita'. A note below the field states: 'Label sangita matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.' On the left sidebar, under 'Source Code Management', there are two options: 'None' (selected) and 'Git'.

- ❖ In build environment I mention that delete workspace creating the project.

The screenshot shows the 'Build Environment' section of the configuration page. Under 'General', there is a checked checkbox for 'Delete workspace before build starts'. Below it, there is an 'Advanced' dropdown menu. Under 'Advanced', several other options are listed: 'Use secret text(s) or file(s)', 'Add timestamps to the Console Output', 'Inspect build log for published build scans', 'Terminate a build if it's stuck', and 'With Ant'. The 'Build Environment' tab is highlighted in the sidebar.

- ❖ Here I gave commands in execute shell to do automation.

Jenkins Master Slave Configuration

The screenshot shows the Jenkins 'Configuration' screen for a project named 'sangita-project'. On the left, a sidebar lists configuration sections: General, Source Code Management, Build Triggers, Build Environment, Build Steps (which is selected), and Post-build Actions. The main area displays a 'Execute shell' build step. The 'Command' field contains a shell script to install Docker and run Docker Compose. Buttons at the bottom allow saving or applying the changes.

```
#!/bin/bash
sudo yum install git -y
git clone https://github.com/sangitagit/docker-compose.git
sudo yum -y install docker
sudo service docker start
sudo usermod -a -G docker ec2-user
sudo chmod 666 /var/run/docker.sock
sudo systemctl enable docker
sudo chkconfig docker on
sudo curl -L "https://github.com/docker/compose/releases/download/1.27.4/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
cd /home/sangita/workspace/sangita-project/docker-compose
sudo docker-compose up -d
```

Save Apply

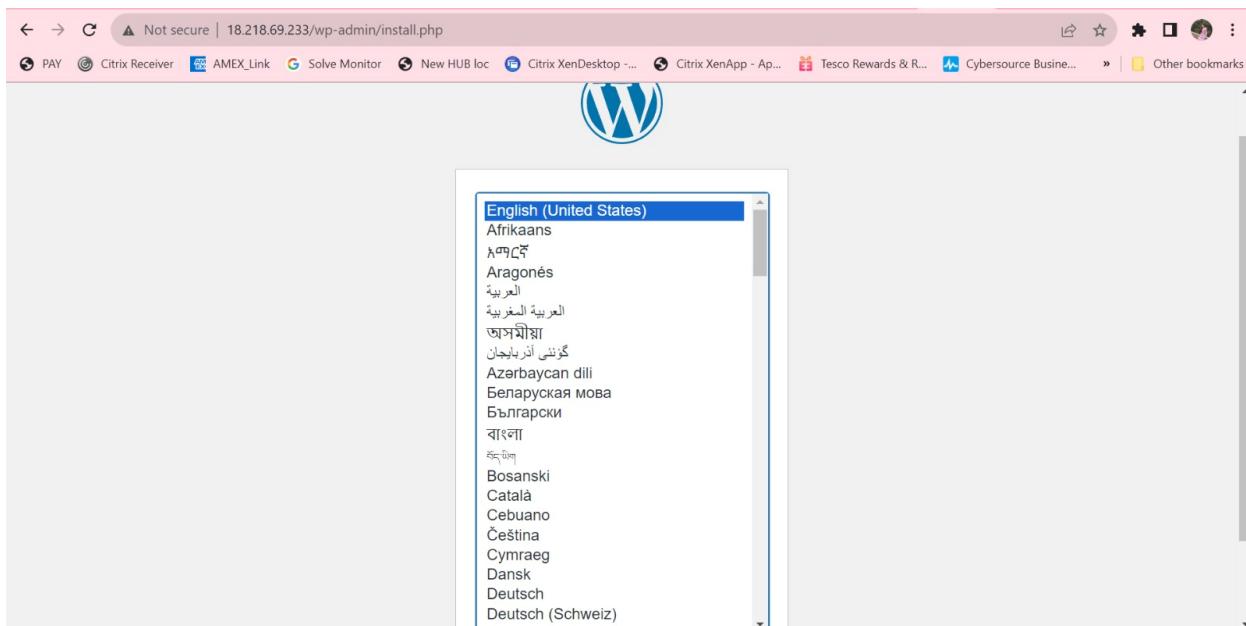
- ❖ After configuring I made build now and checked weather the console output is successful or not.

The screenshot shows the Jenkins 'Console Output' page for build #3 of the 'sangita-project'. The left sidebar includes links for Status, Changes, Console Output (which is selected), View as plain text, Edit Build Information, Delete build '#3', and Previous Build. The main content area shows the 'Console Output' with a green checkmark icon. The output log shows the execution of a shell script that installs Docker and runs Docker Compose, indicating a successful build.

```
Started by user ysai
Running as SYSTEM
Building remotely on sangita in workspace /home/sangita/workspace/sangita-project
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Deferred wipeout is used...
[WS-CLEANUP] Done
[sangita-project] $ /bin/bash /tmp/jenkins10688549007513304307.sh
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Package git-2.40.1-1.amzn2.0.1.x86_64 already installed and latest version
Nothing to do
Cloning into 'docker-compose'...
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Package docker-20.10.23-1.amzn2.0.1.x86_64 already installed and latest version
Nothing to do
Redirecting to /bin/systemctl start docker.service
Note: Forwarding request to 'systemctl enable docker.service'.
% Total    % Received % Xferd  Average Speed   Time     Time   Current
          0      0      0      0      0      0      0      0      0      0      0      0
```

Output :

Jenkins Master Slave Configuration



Ajalal slave configuration :

- ❖ In this Ajjal created one ec2 instance and created one user in that instance and shared to me

How would you rate your experience with this service console? ☆☆☆☆☆

Instances (1/2) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
master1	i-0bbf360ddcc5fa0b1	Stopped	t2.medium	-	No alarms	us-east-1d
slave-3	i-075e05df207e0351f	Running	t2.medium	2/2 checks passed	No alarms	us-east-1c

Instance: i-075e05df207e0351f (slave-3)

Details Security Networking Storage Status checks Monitoring Tags

Instance summary Info

Instance ID i-075e05df207e0351f (slave-3)	Public IPv4 address 44.202.128.183 open address	Private IPv4 addresses 172.31.86.175
IPv6 address	Instance state	Public IPv4 DNS

- ❖ Here I added Ajjal slave in my manage Jenkins = Nodes.

Jenkins Master Slave Configuration

The screenshot shows the Jenkins interface for managing nodes. At the top, there's a navigation bar with the Jenkins logo and the word "Jenkins". Below it, the URL "Dashboard > Nodes > ajlal" is visible. On the left, a sidebar lists several options: "Status" (selected), "Delete Agent", "Configure", "Build History", "Load Statistics", and "Log". A dropdown menu titled "Build Executor Status" is open at the bottom of the sidebar.

- ❖ Created a new project by using Ajjal slave as a Ajjal-project.

The screenshot shows the Jenkins configuration page for a project named "Ajjal-project". The top navigation bar includes the Jenkins logo, a search bar, and user information for "ysai". The URL "Dashboard > Ajjal-project > Configuration" is shown. The main area has two tabs: "Configure" (selected) and "General". The "General" tab contains a "Description" field with the value "Ajjal project". There are also several checkboxes for build options: "Discard old builds", "GitHub project", "This project is parameterized", and "Throttle builds". At the bottom are "Save" and "Apply" buttons. To the right of the tabs, there's an "Enabled" switch which is turned on.

Jenkins Master Slave Configuration

- ❖ Here I mentioned in my configuration where to run this project. I gave Ajjal's slave label to identify the user.

The screenshot shows the Jenkins 'Configuration' page for the 'Ajjal-project'. The 'General' tab is selected. Under 'Restrict where this project can be run', there is a 'Label Expression' field containing 'Ajjal'. A note below it states: 'Label Ajjal matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.'

- ❖ Here I gave git repository to clone into that repo

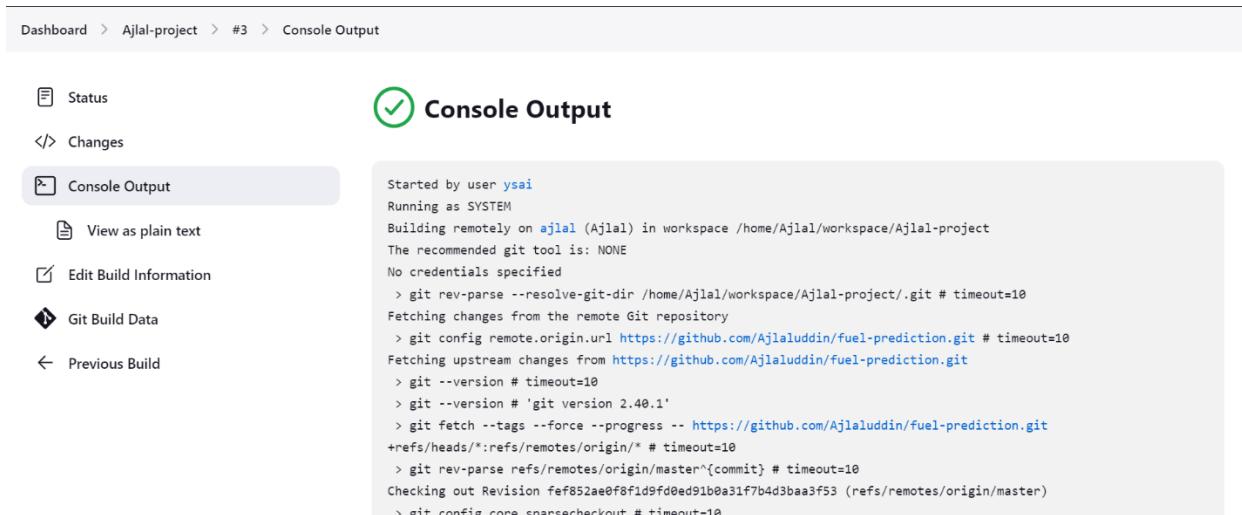
The screenshot shows the Jenkins 'Configuration' page for the 'Ajjal-project'. The 'Source Code Management' tab is selected. Under 'Git', the 'Repository URL' is set to 'https://github.com/Ajjaluddin/fuel-prediction.git'. The 'Credentials' dropdown is set to '- none -'. There is also an 'Add' button and an 'Advanced' section.

- ❖ Now I am mentioning the branch specifier

The screenshot shows the Jenkins 'Configuration' page for the 'Ajjal-project'. The 'Build Triggers' tab is selected. Under 'Branches to build', the 'Branch Specifier' is set to '/master'. There is an 'Add Branch' button. Below it, the 'Repository browser' is set to '(Auto)'. There is also an 'Additional Behaviours' section with an 'Add' button.

Jenkins Master Slave Configuration

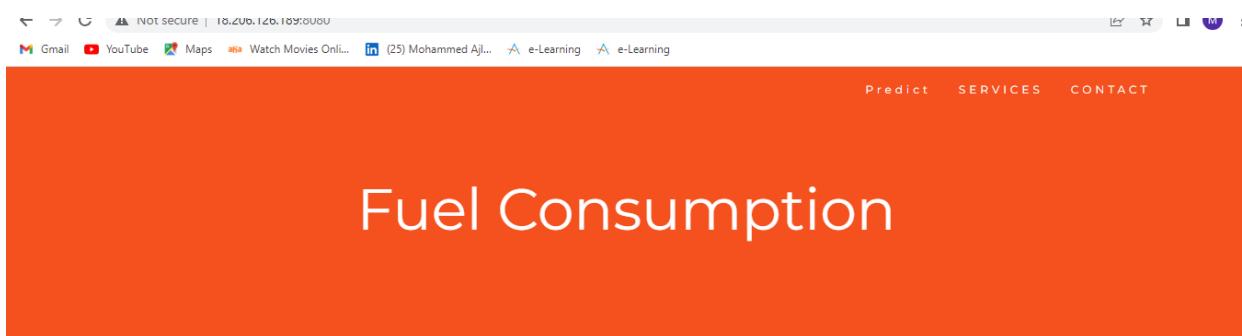
- ❖ After configuring I made build now and checked weather the console output is successful or not.



The screenshot shows the Jenkins interface for a build named '#3'. The 'Console Output' tab is selected, displaying a green checkmark icon and the title 'Console Output'. The log output is as follows:

```
Started by user ysa1
Running as SYSTEM
Building remotely on ajjal (Ajjal) in workspace /home/Ajjal/workspace/Ajjal-project
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /home/Ajjal/workspace/Ajjal-project/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/Ajjaluddin/fuel-prediction.git # timeout=10
Fetching upstream changes from https://github.com/Ajjaluddin/fuel-prediction.git
> git --version # timeout=10
> git --version # 'git version 2.40.1'
> git fetch --tags --force --progress -- https://github.com/Ajjaluddin/fuel-prediction.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision fef852ae0f8f1d9fd0ed91b0a31f7b4d3baa3f53 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
```

Output :



This is a detailed view of the same web form shown in the previous screenshot. It includes fields for 'Make *' (Acura), 'Model *' (ILX), 'Vehicle Class *' (Compact), 'Transmission *' (AM8), 'Fuel Type *' (Fuel Type), and 'Engine *' (Engine). Each field has a small red asterisk indicating it is required.

Jagadesh slave configuration :

- ❖ Created a new project by using Jagadeesh slave as a Jaggu.

Jenkins Master Slave Configuration

The screenshot shows the Jenkins configuration interface for a project named 'jaggu'. The 'General' tab is selected. In the 'Description' field, the value 'jaggu' is entered. Below the description, there are four optional checkboxes: 'Discard old builds', 'GitHub project', 'This project is parameterized', and 'Throttle builds'. At the bottom of the page are two buttons: 'Save' and 'Apply'.

- ❖ Here I mentioned in my configuration where to run this project. I gave jaggu's slave label to identify the user.

The screenshot shows the Jenkins configuration interface for a project named 'jaggu'. The 'Source Code Management' tab is selected. Under 'Label Expression', the value 'jaggu' is entered. A note below states 'Label jaggu matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.' The 'Advanced' dropdown is open. In the 'Source Code Management' section, 'None' is selected as the provider, with 'Git' as an alternative option.

- ❖ In build environment I mention that delete workspace creating the project.

Jenkins Master Slave Configuration

Dashboard > jaggu > Configuration

Configure

Build Environment

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

- Delete workspace before build starts
 - Advanced
- Use secret text(s) or file(s) ?
- Add timestamps to the Console Output
- Inspect build log for published build scans
- Terminate a build if it's stuck
- With Ant ?

- ❖ Here I gave commands in execute shell to do automation.

Dashboard > jaggu > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions

Execute shell

Command
See [the list of available environment variables](#)

```
#!/bin/bash
sudo yum -y update
sudo yum -y install git
sudo yum install python3-pip -y
git clone https://Jagadeesh0007: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@github.com/Jagadeesh0007/Abalone-Age-Prediction-copy
cd /home/workspace/jaggu/Abalone-Age-Prediction-copy
screen -m -d python3 app.py
```

Advanced

Add build step ▾

Save

Apply

- ❖ After configuring I made build now and checked weather the console output is successful or not.

Jenkins Master Slave Configuration

The screenshot shows the Jenkins interface for a build named 'jaggu'. The top navigation bar includes links for 'Dashboard', 'jaggu', '#7', and 'Console Output'. On the left, a sidebar lists options like 'Status', 'Changes', 'Console Output' (which is selected), 'View as plain text', 'Edit Build Information', 'Delete build', and 'Previous Build'. The main content area is titled 'Console Output' with a green checkmark icon. It displays the build log:

```
Started by user ysai
Running as SYSTEM
Building remotely on jaggu in workspace /home/jaggu/workspace/jaggu
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Deferred wipeout is used...
[WS-CLEANUP] Done
[jaggu] $ /bin/bash /tmp/jenkins13950572265044227282.sh
Last metadata expiration check: 1:49:56 ago on Mon Jul 24 06:28:47 2023.
Dependencies resolved.
Nothing to do.
Complete!
Last metadata expiration check: 1:49:56 ago on Mon Jul 24 06:28:47 2023.
Package git-2.40.1-1.amzn2023.0.1.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
```

Output :

The screenshot shows a web browser window with the title 'Abalone-Age-Prediction-Yshu'. The main content area contains two sections: 'What is an Abalone?' and 'Sample Image Of Abalone'. The 'What is an Abalone?' section features a video thumbnail from YouTube with the title 'What is abalone?'. The 'Sample Image Of Abalone' section shows a large, iridescent abalone shell with the text 'abalone shell' written above it. A descriptive paragraph below the image states: 'The believed abalone healing properties of today are very similar to ancient beliefs of the shell. Abalone healing carries energies of protection and emotional balance. It brings with it a natural shielding that blesses the person holding it with tranquility.'

John slave configuration :

- ❖ In this John created one ec2 instance and created one user in that instance and shared to me.

Jenkins Master Slave Configuration

The screenshot shows the AWS EC2 Management Console. The main view displays a table of instances with one entry: 'j-john' (Instance ID: i-027cc7db0cb4e69a8). The instance is listed as 'Running' with a status check of '2/2 checks passed'. Below the table, the 'Instance: i-027cc7db0cb4e69a8 (j-john)' details page is open. It shows the public IPv4 address (3.27.18.159) and private IP (172.31.31.71). The DNS name is ec2-3-27-18-159.ap-southeast-2.compute.amazonaws.com. The 'Status Checks' tab is selected, showing a green circle indicating success.

- ❖ Here I added John slave in my manage Jenkins = Nodes.

The screenshot shows the Jenkins node configuration page for 'john'. The top navigation bar includes 'Dashboard > Nodes > john'. Below this, there are several options: 'Status' (selected), 'Delete Agent', 'Configure', 'Build History', 'Load Statistics', and 'Log'. A 'Build Executor Status' section is also visible at the bottom.

Jenkins Master Slave Configuration

- ❖ Created a new project by using John slave as a John-project.

The screenshot shows the Jenkins configuration interface for a project named 'john-project'. The 'General' tab is selected. In the 'Description' field, 'john-project' is entered. Under build triggers, the 'Discard old builds' checkbox is unchecked. Under build environment, the 'Label Expression' field contains 'john'. The 'Save' button is highlighted in blue, while 'Apply' is greyed out.

- ❖ Here I mentioned in my configuration where to run this project. I gave John's slave label to identify the user.

The screenshot shows the Jenkins configuration interface under the 'Build Environment' section. The 'Label Expression' field is set to 'john'. Below it, the 'Source Code Management' section is shown with 'None' selected as the provider.

- ❖ In build environment I mention that delete workspace creating the project.

Jenkins Master Slave Configuration

The screenshot shows the Jenkins configuration interface for a project named "john-project". The "Build Environment" section is selected. Under "Delete workspace before build starts", the checkbox is checked. Other options like "Use secret text(s) or file(s)", "Add timestamps to the Console Output", "Inspect build log for published build scans", "Terminate a build if it's stuck", and "With Ant" are available but unchecked.

- ❖ Here I gave commands in execute shell to do automation.

The screenshot shows the Jenkins configuration interface for a project named "john-project". The "Build Steps" section is selected. A single "Execute shell" step is defined with the following command:

```
#!/bin/bash
sudo yum install git -y
git clone https://kavalakuntlajohnwesly:ghp_OEmjmtL4yz5fCEhys2j54NC08nkWuv2OispY@github.com/kavalakunt
sudo yum -y install docker
sudo service docker start
sudo usermod -a -G docker ec2-user
sudo chmod 666 /var/run/docker.sock
sudo systemctl enable docker
sudo chkconfig docker on
sudo curl -L "https://github.com/docker/compose/releases/download/1.27.4/docker-compose-$(uname -s)-$(
sudo chmod +x /usr/local/bin/docker-compose
sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
cd /home/john/workspace/john-project/Docker-compose
sudo docker-compose up -d
```

- ❖ After configuring I made build now and checked weather the console output is successful or not.

Jenkins Master Slave Configuration

The screenshot shows the Jenkins interface for a build named 'john-project' (Build #3). The 'Console Output' tab is selected. The output log shows the following:

```
Started by user ysai
Running as SYSTEM
Building remotely on john in workspace /home/john/workspace/john-project
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Deferred wipeout is used...
[WS-CLEANUP] Done
[john-project] $ /bin/bash /tmp/jenkins4443635962069181488.sh
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Package git-2.40.1-1.amzn2.0.1.x86_64 already installed and latest version
Nothing to do
Cloning into 'Docker-compose'...
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Package docker-20.10.23-1.amzn2.0.1.x86_64 already installed and latest version
Nothing to do
Redirecting to /bin/systemctl start docker.service
```

Output :

