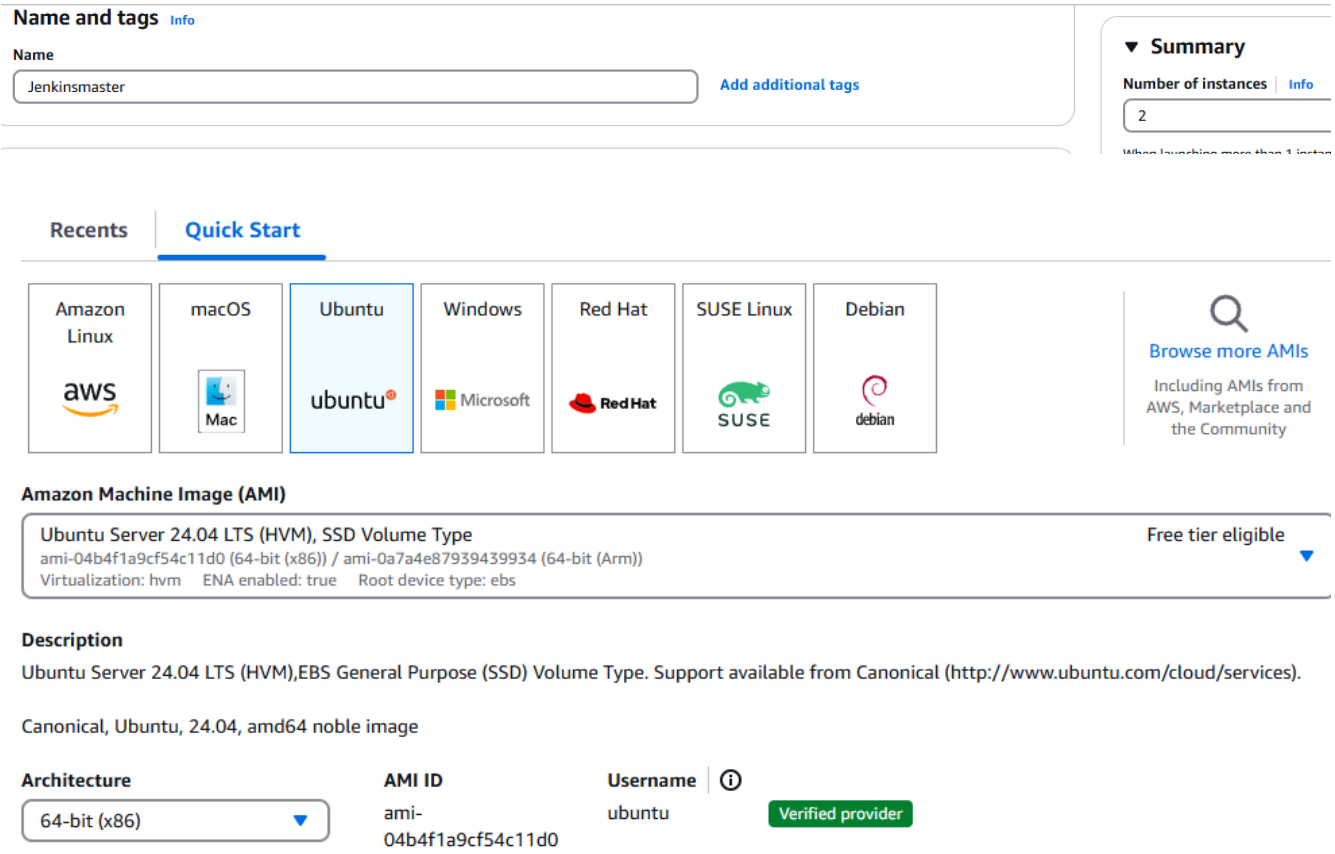
Jenkins setup on AWS EC2 with Ubuntu instances.

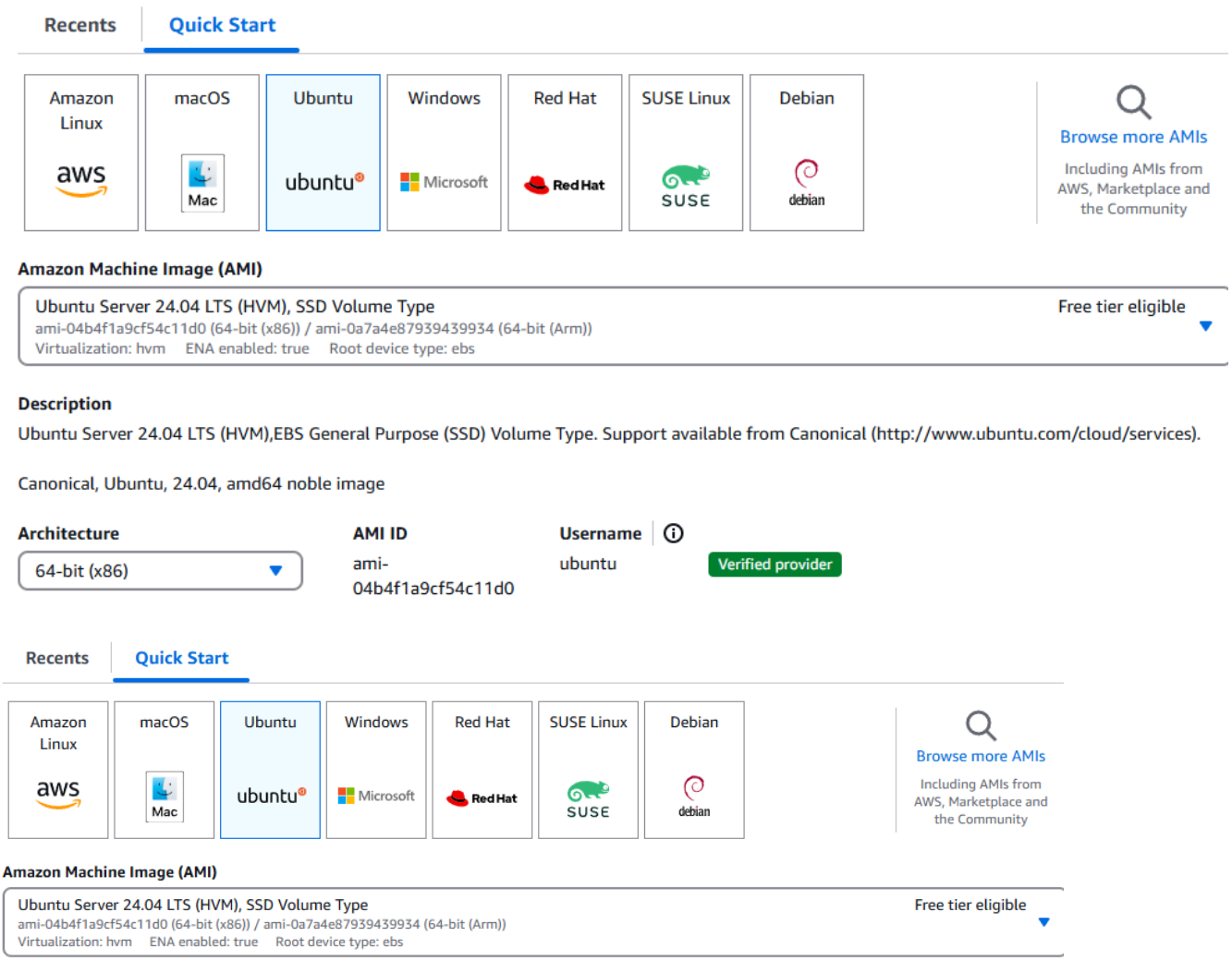
We will be creating two Ec2 instances, 1 is master and second is slave machine. This is the test lab hence we will be using master VM with 4GB ram, in production environment we need to have good amount of ram, its better to have T2 large, xtra large, instance with multi AZ depending about the load, in case of multi region deployment and automation.

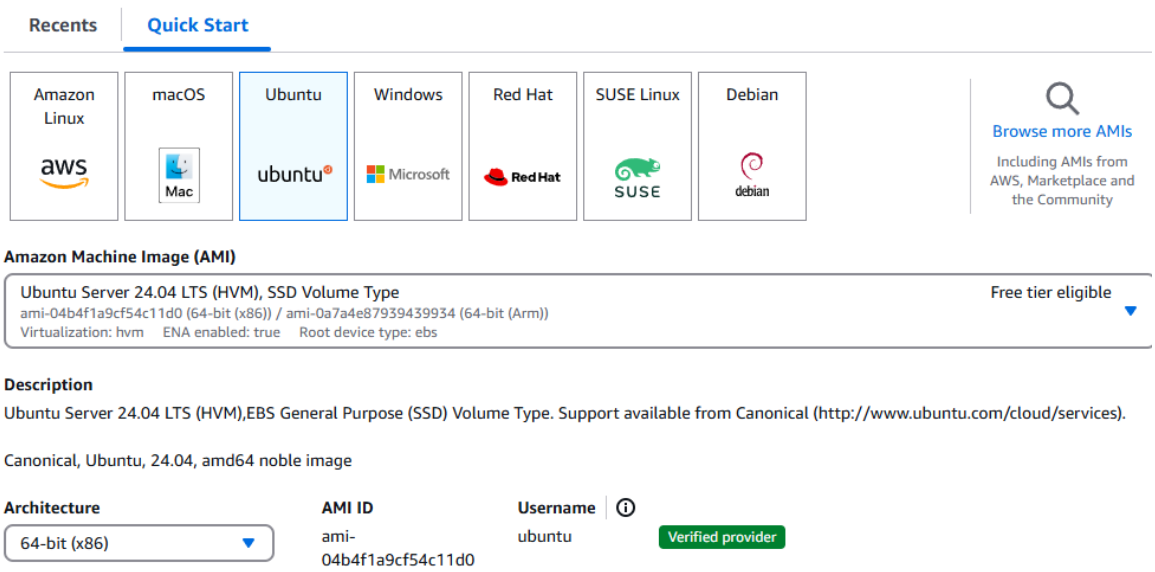
Here I have used the below link to download and setup Master instance, for slave machine we just need to have java installed with SSH key authentication so that master node can get authentication.

1. Login to AWS console and click on Launch Ec2 instance.

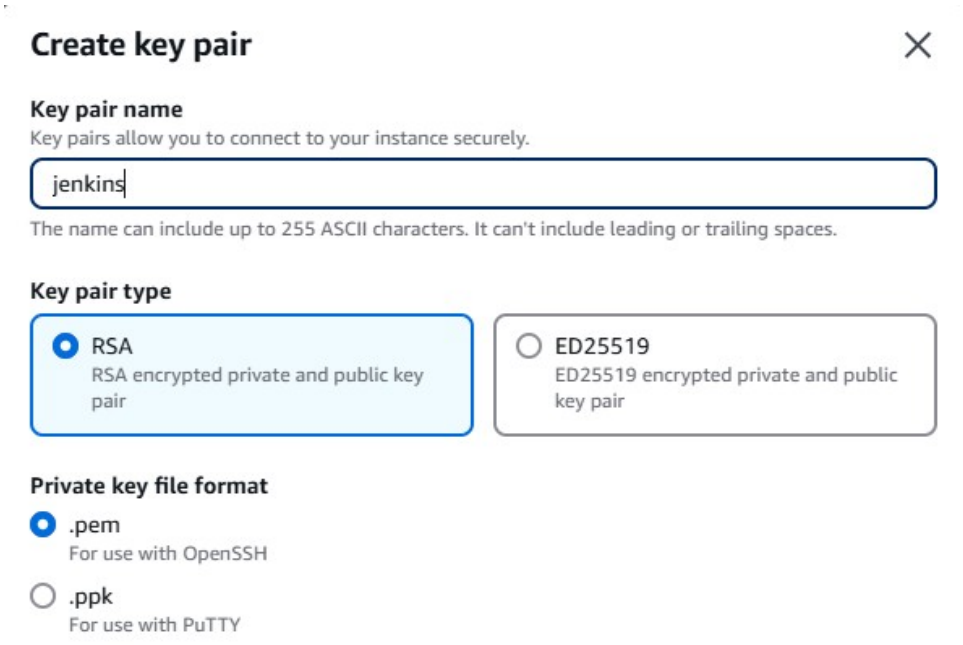


2. We will be using Ubuntu AMI version – Ubuntu 24.04 LTS.

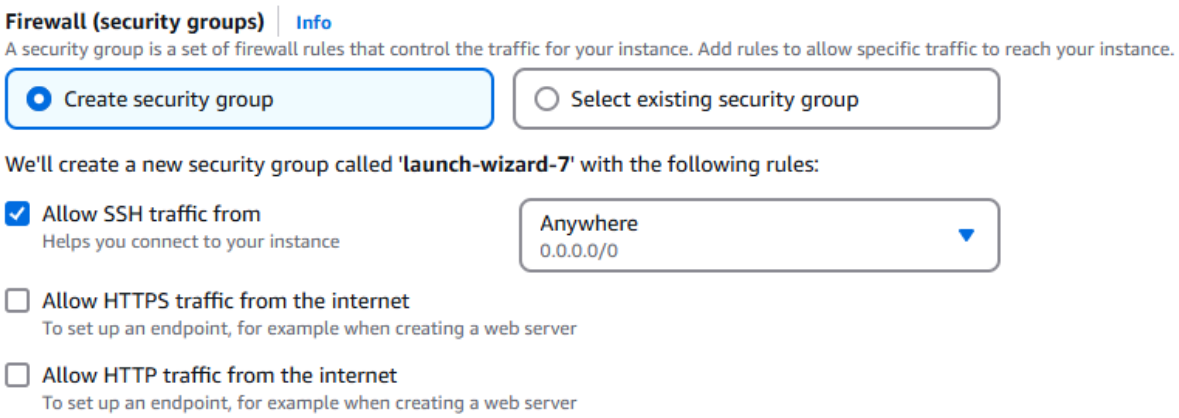




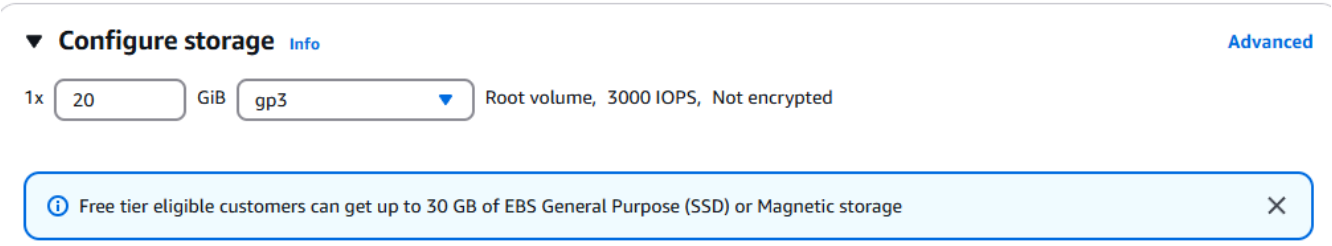
We need to create Keypair, we can login to Jenkins and perform steps to download and install Jenkins. We are creating Public key .pem



Security group we are allowing port 22. There are other ports we need to allow.



Hard disk we are allowing 20 GB, in production environment it will be more. Since this is test lab environment, we are using 20GB- General purpose



We can now see Two instances are created and checks are passed. I have renamed the second instance as Jenkins slave so that we can differentiate.

Jenkinsmaster	i-075535fe1fde238b2	Running	t2.medium	2/2 checks passec	View alarms +
Jenkinsslave	i-0d26316f98cb5933a	Running	t2.medium	2/2 checks passec	View alarms +

We can now connect to our Ubuntu instance, using git bash.

EC2 Instance Connect | Session Manager | SSH client | EC2 serial console

Instance ID

i-075535fe1fde238b2 (Jenkinsmaster)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is jenkins.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   chmod 400 "jenkins.pem"
4. Connect to your instance using its Public DNS:
   ec2-18-212-223-91.compute-1.amazonaws.com

Example:

ssh -i "jenkins.pem" ubuntu@ec2-18-212-223-91.compute-1.amazonaws.com

We can launch Git bash and go to downloads folder. This is path where we have downloaded the public key.

MINGW64:/c/Users/ALLTECHACC/Downloads

```
ALLTECHACC@Santopc MINGW64 ~ (master)
$ cd ~/Downloads/

ALLTECHACC@Santopc MINGW64 ~/Downloads (master)
$ |
```

```
MINGW64:/c/Users/ALLTECHACC/Downloads

ALLTECHACC@Santopc MINGW64 ~ (master)
$ cd ~/Downloads/

ALLTECHACC@Santopc MINGW64 ~/Downloads (master)
$ ssh -i "jenkins.pem" ubuntu@ec2-18-212-223-91.compute-1.amazonaws.com
The authenticity of host 'ec2-18-212-223-91.compute-1.amazonaws.com (18.212.223.
91)' can't be established.
ED25519 key fingerprint is SHA256:q5QGcEvDTE699v4XfSrY6d9UM8B3cOJsIQRlXk++G3I.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

When we get prompt to add the key, we can add the key, type yes, and it will use the key and login to server. We have now logged in successfully to the server.

```
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-172-31-19-251:~$ uptime
 21:56:36 up 27 min,  1 user,  load average: 0.02, 0.01, 0.00
```

We now need to perform the steps as given in the website, for ubuntu machine.

Below is the link for installing jenkins in the master node.

https://www.jenkins.io/doc/book/installing/linux/#debianubuntu

Master node configuration

1. Log into the master node and perform the steps as mentioned in the document.

```
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
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
sudo apt-get update
sudo apt-get install jenkins



sudo apt update
sudo apt install fontconfig openjdk-17-jre
java -version
openjdk version "17.0.13" 2024-10-15
OpenJDK Runtime Environment (build 17.0.13+11-Debian-2)
OpenJDK 64-Bit Server VM (build 17.0.13+11-Debian-2, mixed mode, sharing)
```

We can also perform the automation of the above task by saving it in shell script file and running the script file.

After performing the above steps on ubuntu machine, we have successfully installed the jenkins on master node.

Now we need to start the services. Below are the steps taken from the website to start the services.

Start Jenkins

You can enable the Jenkins service to start at boot with the command:

```
sudo systemctl enable jenkins
```

You can start the Jenkins service with the command:

```
sudo systemctl start jenkins
```

You can check the status of the Jenkins service using the command:

```
sudo systemctl status jenkins
```

After performing the above steps, we have now successfully started the services.

```
ubuntu@ip-172-31-19-251:~$ sudo systemctl start jenkins
ubuntu@ip-172-31-19-251:~$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
     Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
     Active: active (running) since Tue 2025-02-25 22:15:55 UTC; 10s ago
   Main PID: 5017 (java)
      Tasks: 52 (limit: 4676)
     Memory: 622.4M (peak: 634.5M)
        CPU: 15.550s
     CGroup: /system.slice/jenkins.service
             └─5017 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Feb 25 22:15:52 ip-172-31-19-251 jenkins[5017]: 3afc818feb5c46a390b71bbb52a36a62
Feb 25 22:15:52 ip-172-31-19-251 jenkins[5017]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Feb 25 22:15:52 ip-172-31-19-251 jenkins[5017]: *************************************************************
Feb 25 22:15:52 ip-172-31-19-251 jenkins[5017]: *************************************************************
Feb 25 22:15:52 ip-172-31-19-251 jenkins[5017]: *************************************************************
Feb 25 22:15:55 ip-172-31-19-251 jenkins[5017]: 2025-02-25 22:15:55.648+0000 [id=30]	INFO	jenkins.InitReactorRunner$1#onAttained: Completed initialization
Feb 25 22:15:55 ip-172-31-19-251 jenkins[5017]: 2025-02-25 22:15:55.666+0000 [id=23]	INFO	hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running
Feb 25 22:15:55 ip-172-31-19-251 systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Feb 25 22:15:55 ip-172-31-19-251 jenkins[5017]: 2025-02-25 22:15:55.941+0000 [id=48]	INFO	h.m.DownloadService$Downloadable#load: Obtained the updated data file for hudson.tasks.Maven.MavenInstaller
Feb 25 22:15:55 ip-172-31-19-251 jenkins[5017]: 2025-02-25 22:15:55.942+0000 [id=48]	INFO	hudson.util.Retrier#start: Performed the action check updates server successfully at the attempt #1
```

Now need to enable port 8080 and http port from security group to allow access to jenkins.
We need to go to AWS console and make the necessary changes.

Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info	
sgr-038d26457e9537eac	SSH	TCP	22	Custom	0.0.0.0/0
–	Custom TCP	TCP	8080	Anyw...	0.0.0.0/0
–	HTTP	TCP	80	My IP	

After performing the above steps we are now able to access the jenkins url.
We need to use Public ip of our Ec2 instance.

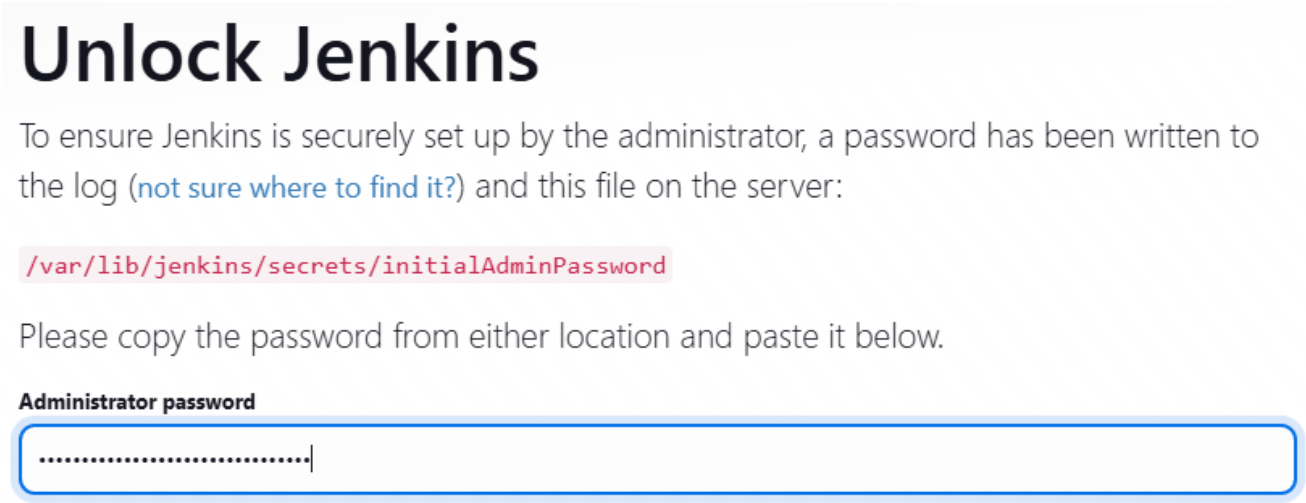
Public IPv4 address
18.212.223.91 | open address

We now need to get the Admin password. We can get the password from the above path.

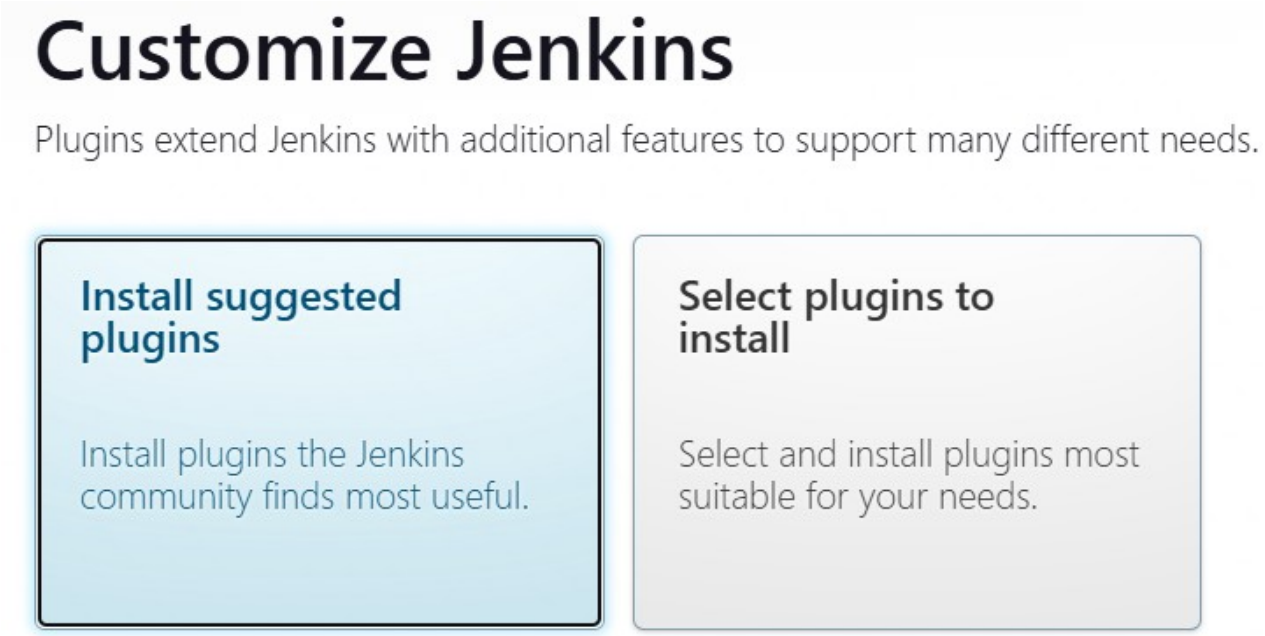
We need to use sudo cat `/var/lib/jenkins/secrets/initialAdminPassword`

```
ubuntu@ip-172-31-19-251:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
3afc818feb5c46a390b71bbb52a36a62
```

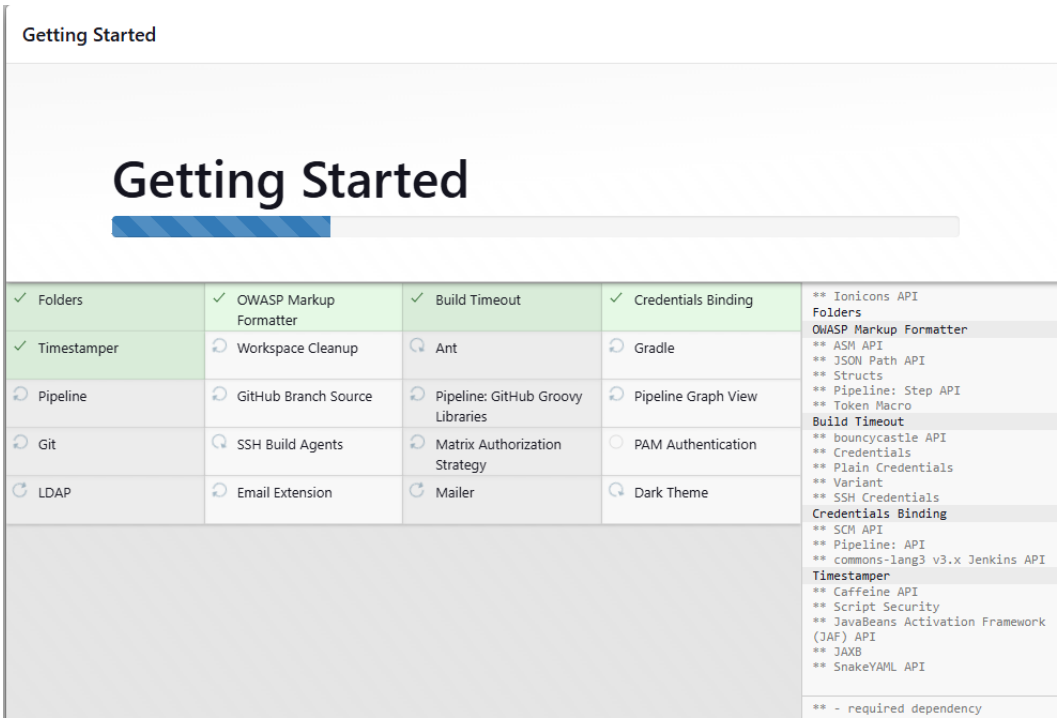
We just need to copy and paste it should login.



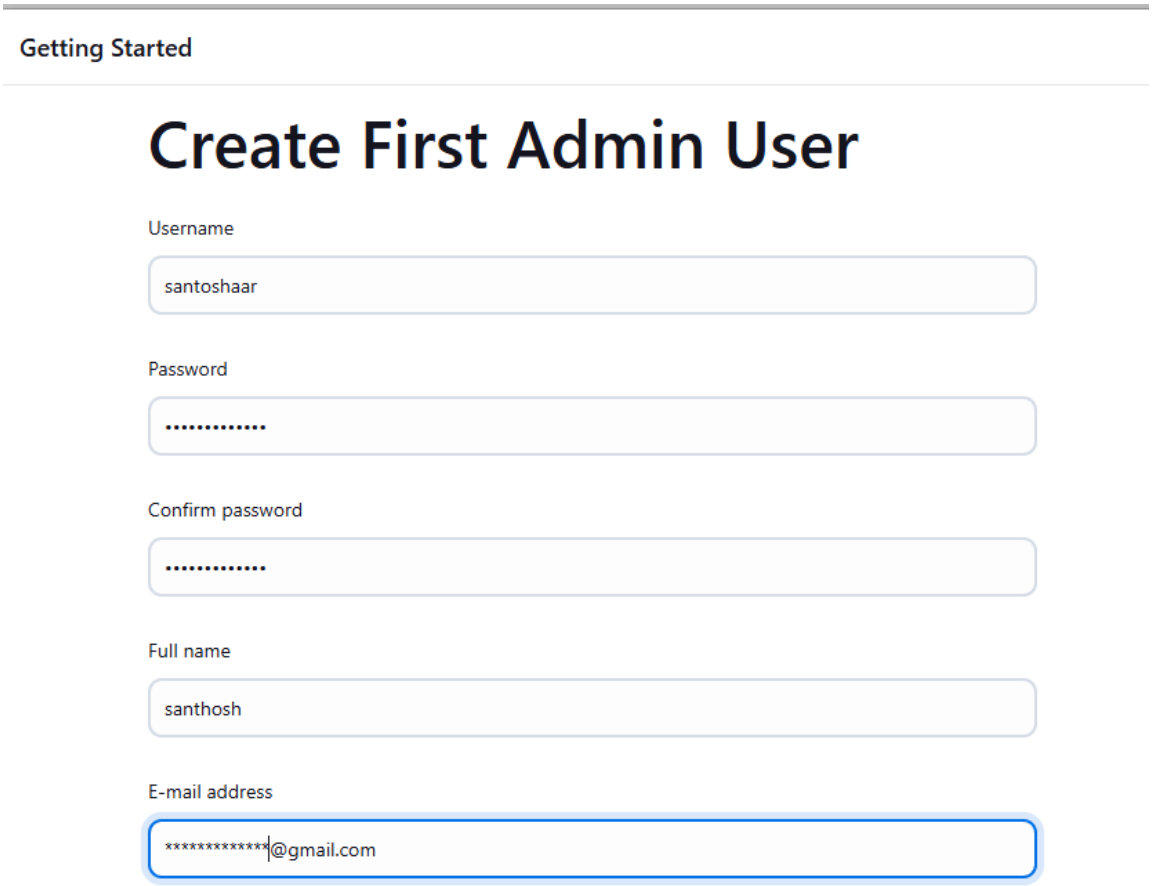
We are now successfully logged into jenkins.



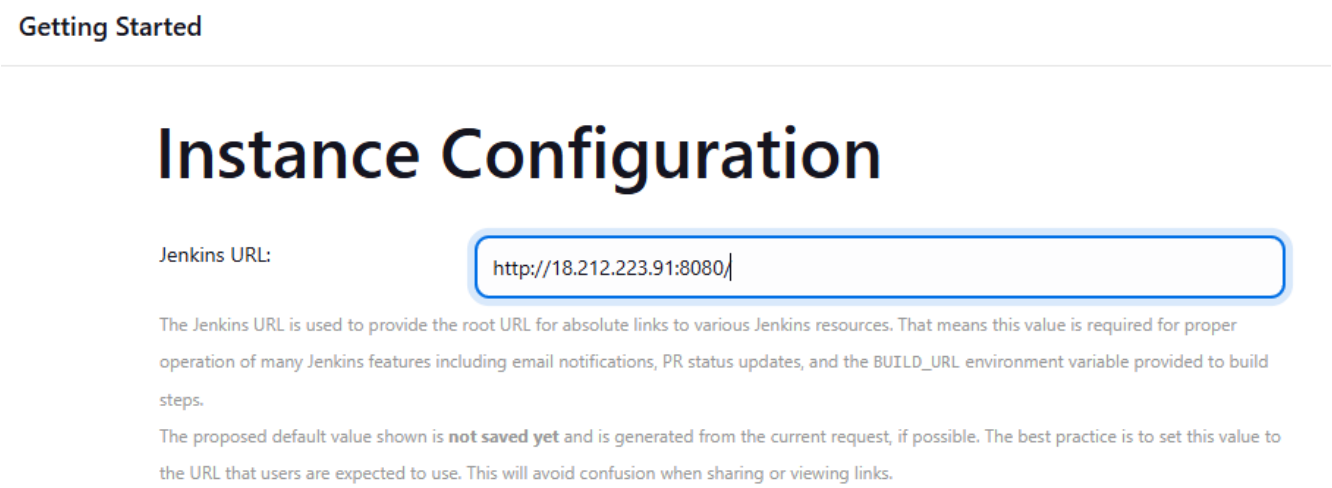
Select the option Install suggested plugins, these will be installed.



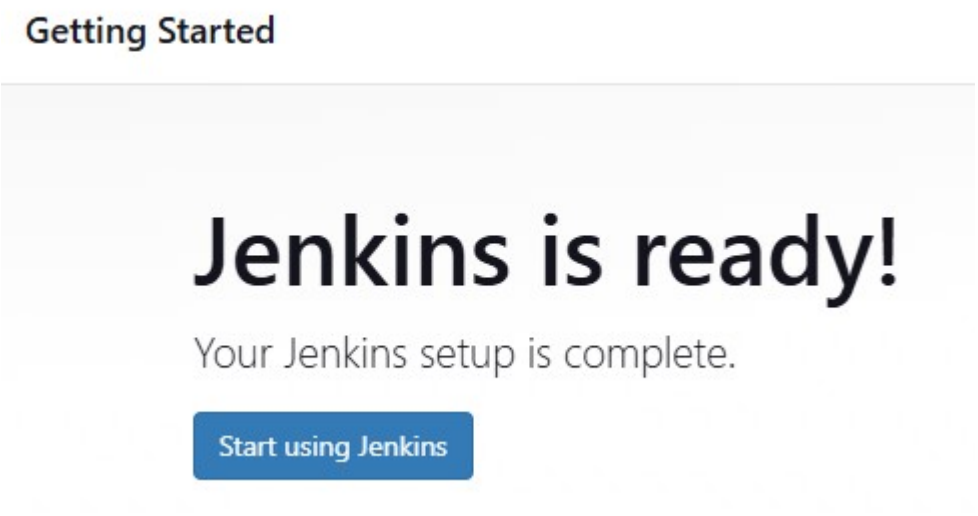
Plugins are now installed and we are now creating user in Jenkins, this is the admin user.



We can now configure the Jenkins url, we can leave it as default. Click on Save and finish



Jenkins Master is now successfully installed.

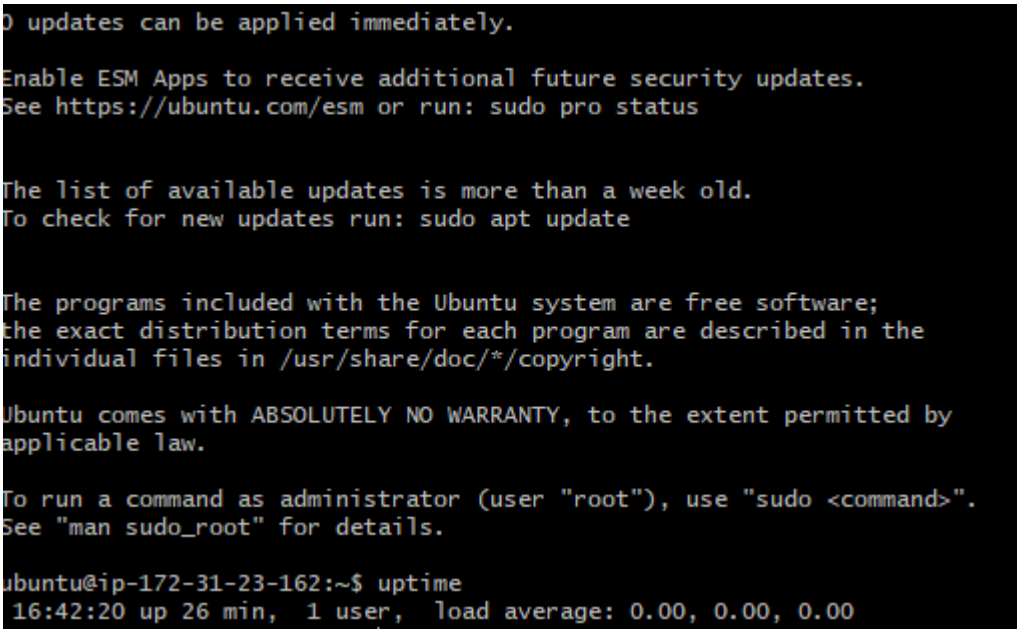


We will now move on Slave configuration.

Slave Node/Worker node configuration.

On slave node, we only need to have java installed, we will be installing Java on ubuntu machine.

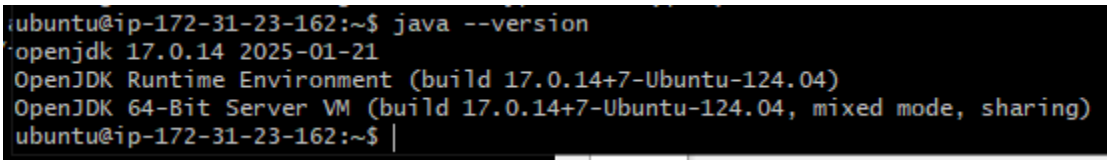
We have successfully logged in to the Ubuntu machine.



```
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-172-31-23-162:~$ uptime
 16:42:20 up 26 min,  1 user,  load average: 0.00, 0.00, 0.00
```

Referring to the link we have downloaded and installed Java

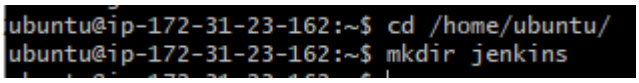
```
sudo apt update
sudo apt install fontconfig openjdk-17-jre
java -version
openjdk version "17.0.13" 2024-10-15
OpenJDK Runtime Environment (build 17.0.13+11-Debian-2)
OpenJDK 64-Bit Server VM (build 17.0.13+11-Debian-2, mixed mode, sharing)

Java is now successfully installed.
```



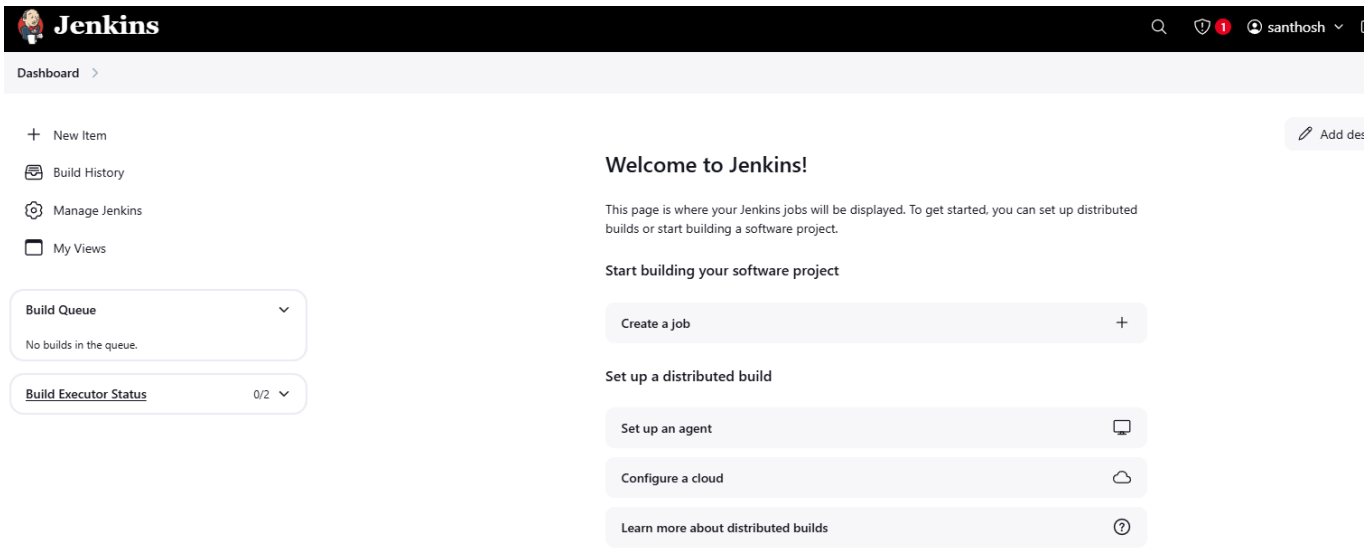
We need to create remote root directory.

We can go to home directory of ubuntu user and create new directory. I have created directory **jenkins**

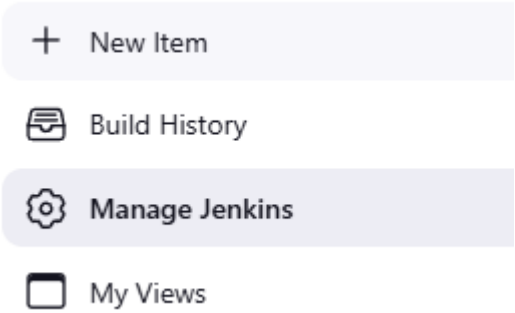
```
ubuntu@ip-172-31-23-162:~$ cd /home/ubuntu/
ubuntu@ip-172-31-23-162:~$ mkdir jenkins
```



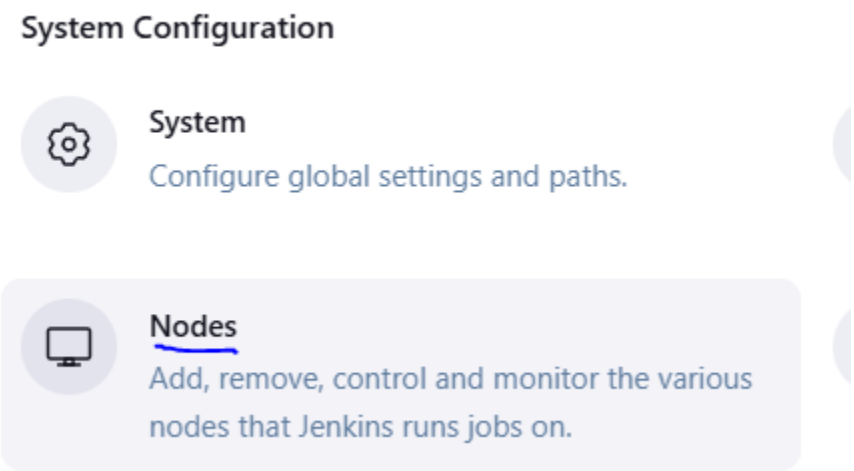
We need to login to Jenkins with the credential that we initially created.



Click on Manage Jenkins

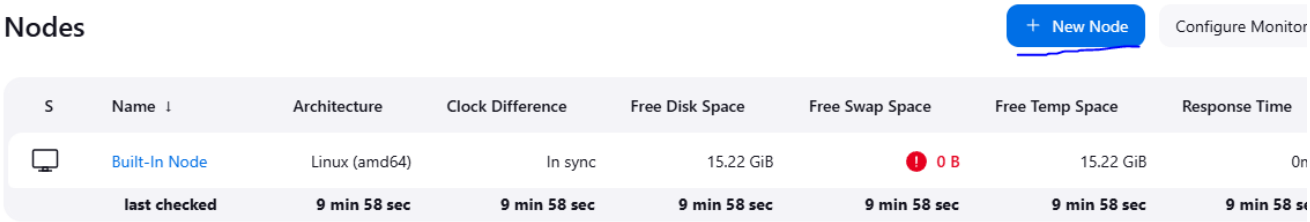


Click on Nodes

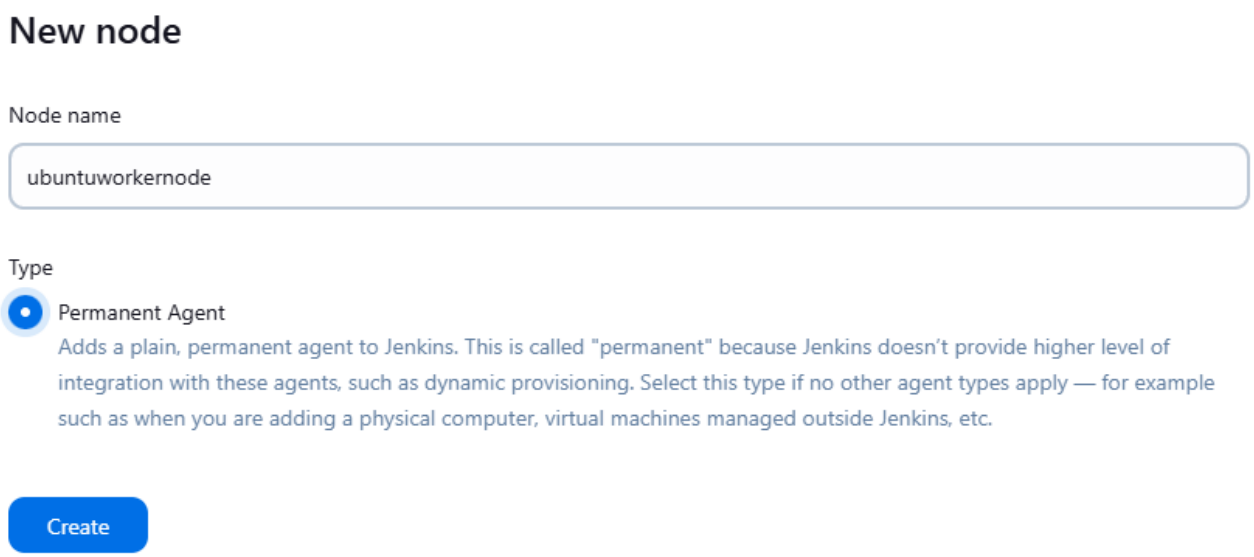


Here we can see our main master node showing details of drive information

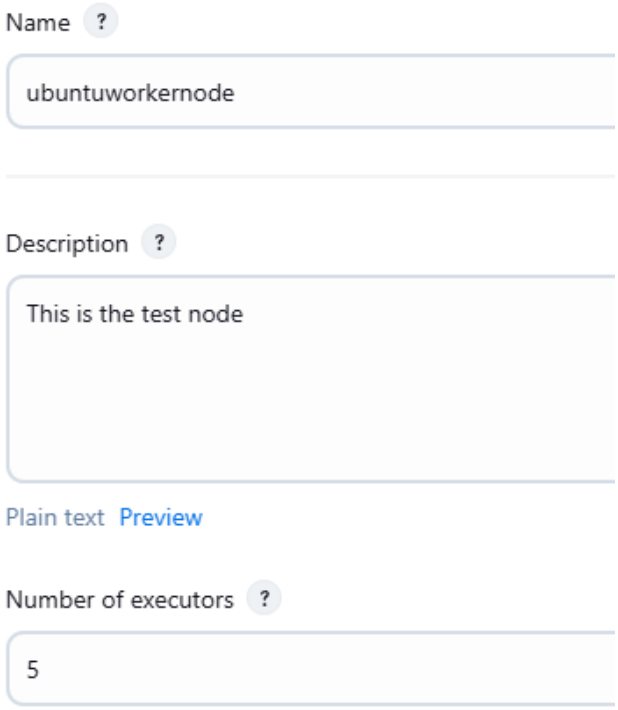
Click on New node. This is on top right hand corner



We are giving name **ubuntuworkernode – Click create**

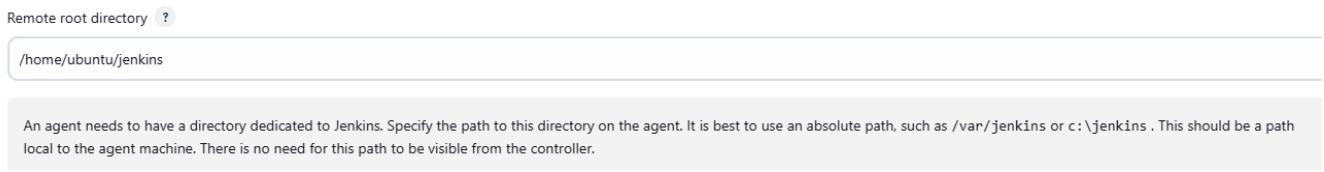


We can update the number of executers here, I have given 5, this is the maximum number of build that this node can perform. This can vary depending on load and type of build.

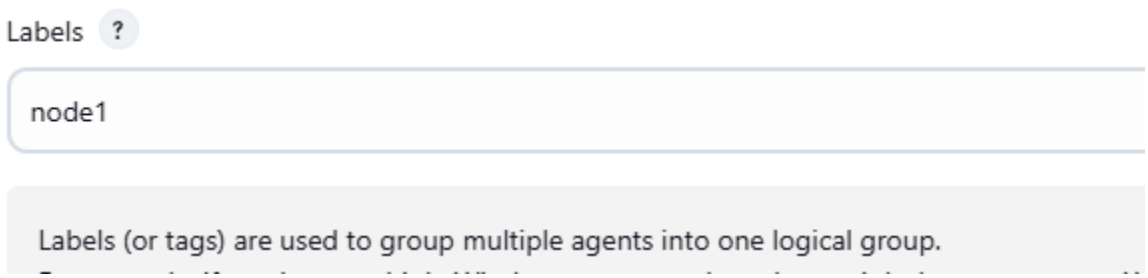


We had created directory for the jenkins we can update the directory name.

/home/ubuntu/jenkins

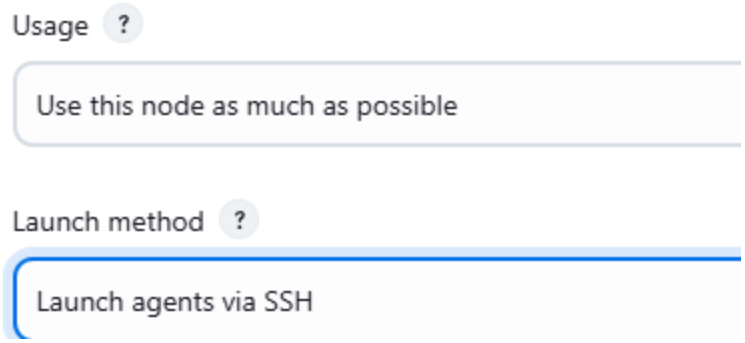


We need to label the node, in case of multiple nodes we can specify the labels to that it will be deployed on that node.



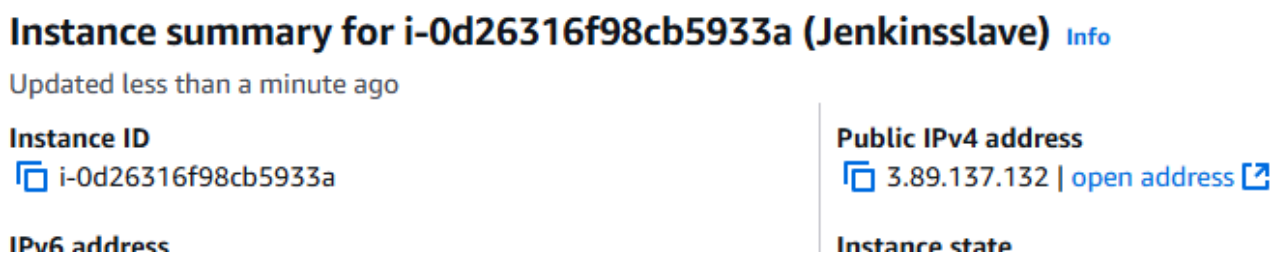
We can specify the option usage "**Use this node as much as possible"**
Second option is **"Only build jobs with label expression matching this node"**

This will be selected in case of multi node and multi deployment environment since we have only 1 node we will be selecting use this node as much as possible.

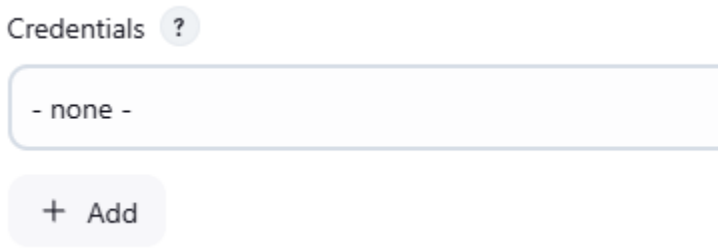


We can select Launch agents via SSH. This is launch method we will be selecting. This is secure way of having worker node.

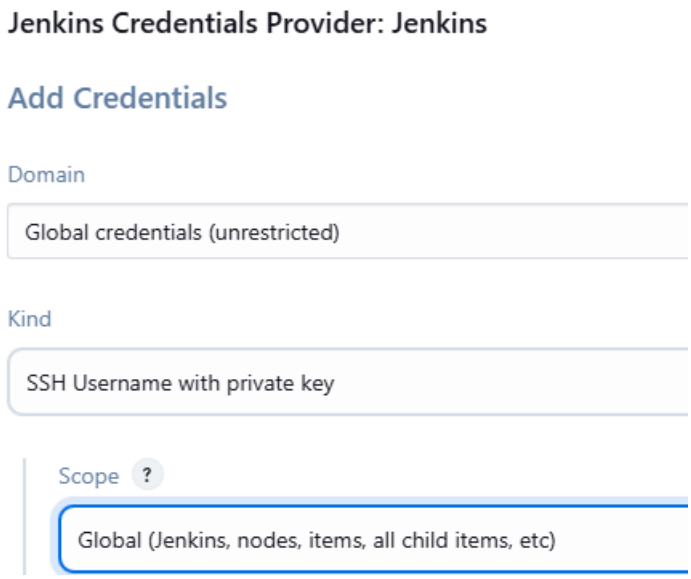
We need to specify the host name, we will be specifying the host name of **SLAVE node** from our AWS Public IP



We need to add the credentials, this is very important part . Click on Add button.



We need to select SSH username and private key



Since we are using ssh connection, we need to setup public and private key.

We now go to our slave node of ubuntu and run the command **ssh-keygen**

```
ubuntu@ip-172-31-23-162:~$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_ed25519): |
```

Press Enter key, keep the default value.
Press Enter key again.

```
ubuntu@ip-172-31-23-162:~$ 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:YGMbHsJtLzg6vr7uWsdS2mvDO5wtJsINAejyePkS7so ubuntu@ip-172-31-23-162
The key's randomart image is:
+--[ED25519 256]--+
|.                |
|o  . .           |
|o   o X          |
|.o   B B         |
|.o..+ + S        |
|..=* . .         |
|.oO=+o           |
|o++=@..          |
|oEOBo=           |
+----[SHA256]-----+
```

So our keys are saved in /home/ubuntu.ssh

```
cd /home/ubuntu/.ssh/
```

```
+----[SHA256]-----+
ubuntu@ip-172-31-23-162:~$ cd /home/ubuntu/.ssh/
ubuntu@ip-172-31-23-162:~/.ssh$ ls
authorized_keys  id_ed25519  id_ed25519.pub
```

Now we need to copy the private key to the Jenkins portal.

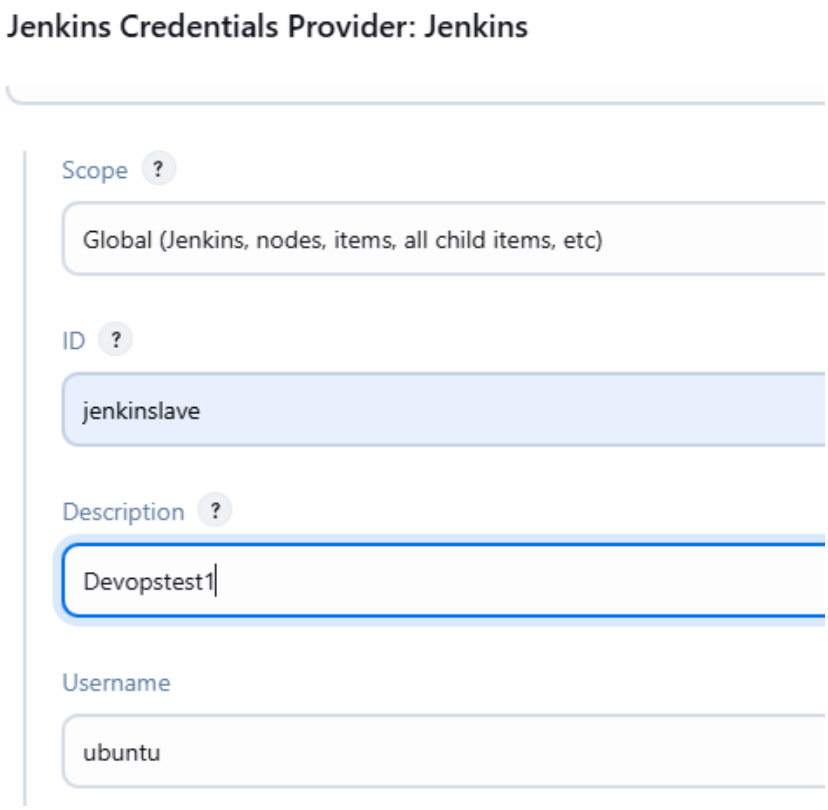
Use the cat command

```
ubuntu@ip-172-31-23-162:~/.ssh$ cat id_ed25519
```

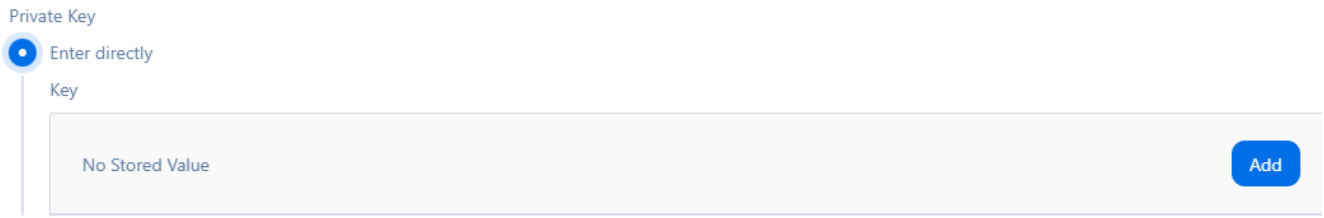
We need to update the details in private key

Select below options in the credential section.

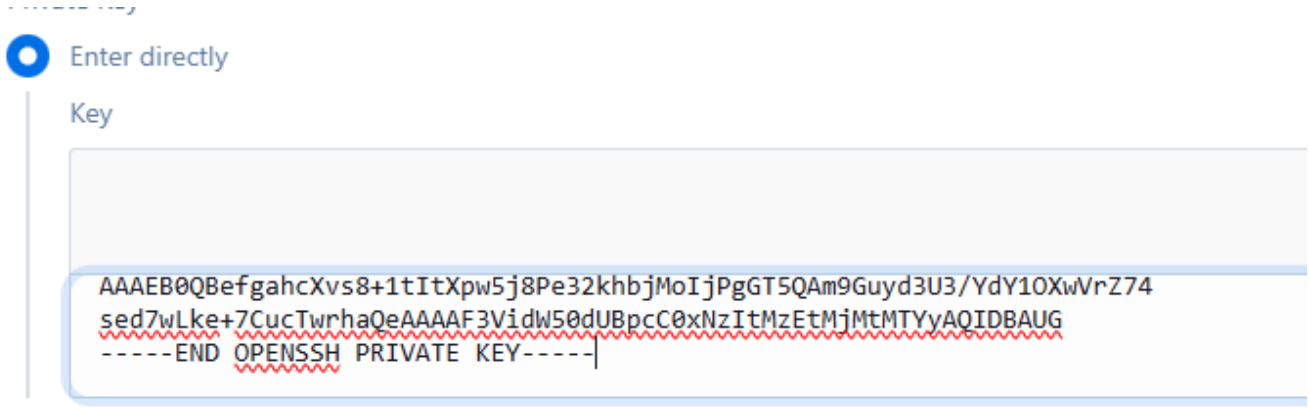
Username should be the username which used to login to slave node
In case if you are using redhat you need to specify root if you are using default redhat login.
Here I am giving ubuntu. Since I have logged in as ubuntu.



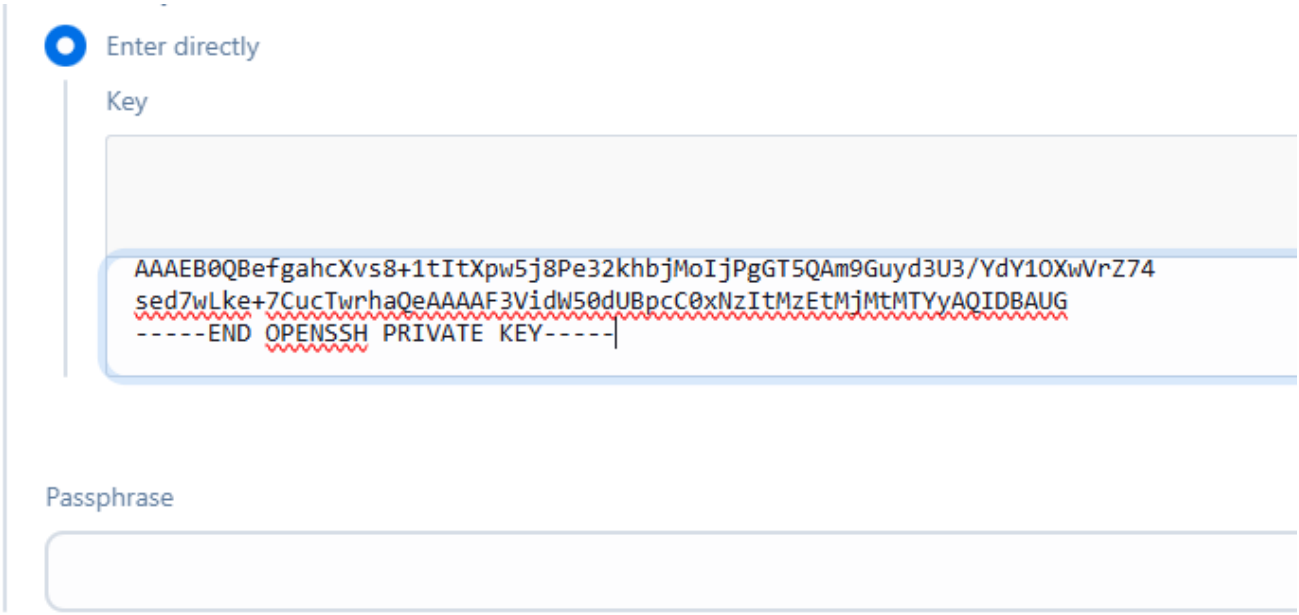
Select the option enter directly and enter the key details. Click Add



Copy and paste the key from ubuntu path where your private key is saved. Make sure there are no additional spaces.



If passphrase is given, mention the passphrase, since we have not given any passphrase, I have left it blank.



Click Add button the add the credential.

Select the credential which we have configured.

Host key Verification strategy. Since we are using test environment we can use non verifying strategy.

The Non-Verifying Host Key Verification Strategy in Jenkins is an approach where Jenkins does not perform any verification of the host key during the SSH connection. Essentially, this strategy bypasses the standard security checks, which can be useful for testing or in environments

Manually Provided Key Verification Strategy:

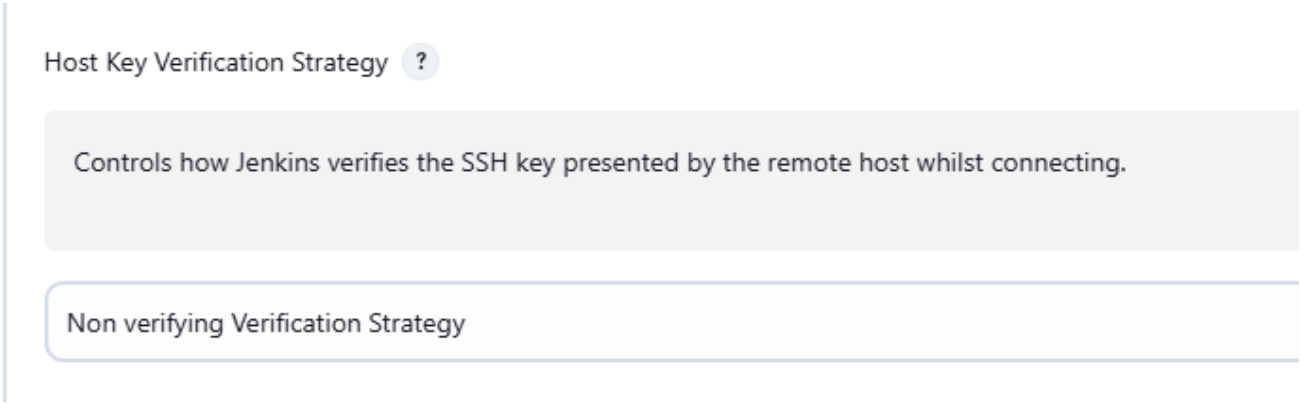
- This strategy checks if the key provided by the remote host matches the key set by the user who configured the connection.

Manually Trusted Key Verification Strategy:

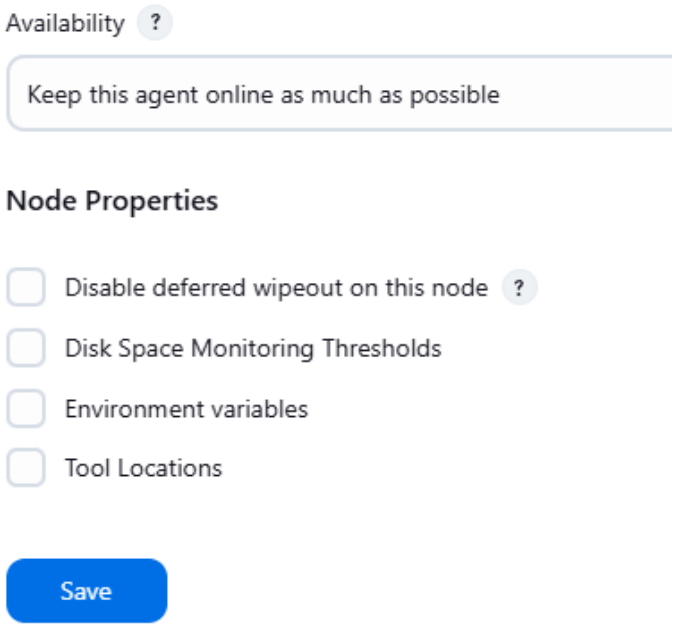
- This strategy checks if the remote key matches the key currently marked as trusted for this host.

- Depending on the configuration, the key will be automatically trusted for the first connection, or an authorized user will be asked to approve the key.

Since this is test environment we are using non Verifying Verification Strategy.



Now we have added the key and created credential, we are now good to save.



- We can now see the node is added however this is not yet completed and **donot** launch the node.

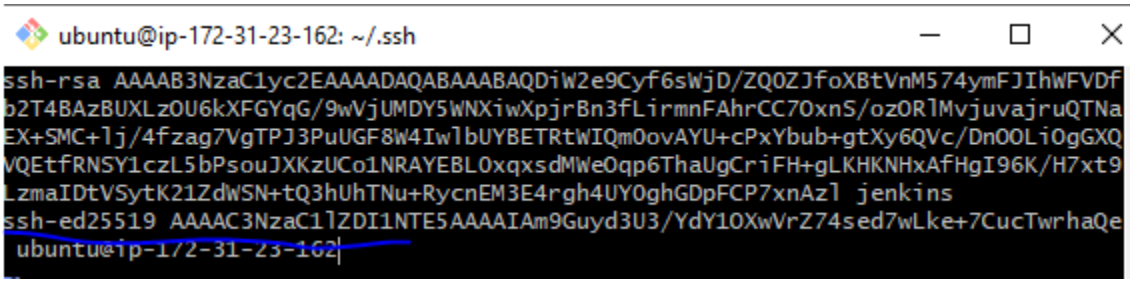
We now need to add the public key on the same node .

Cat the public key

```
 cat id_ed25519.pub
```

```
use the vi editor to add the key
```

```
 vi authorized_keys
```



We need to click on Launch node and Now our slave node is online.

```
TERM=dumb
UID=1000
USER=ubuntu
XDG_RUNTIME_DIR=/run/user/1000
XDG_SESSION_CLASS=user
XDG_SESSION_ID=26
XDG_SESSION_TYPE=tty
_=']'
[02/26/25 18:55:57] [SSH] Starting sftp client.
[02/26/25 18:55:57] [SSH] Copying latest remoting.jar...
[02/26/25 18:55:57] [SSH] Copied 1,395,562 bytes.
Expanded the channel window size to 4MB
[02/26/25 18:55:57] [SSH] Starting agent process: cd "/home/ubuntu/jenkins" && java  -jar remoting.jar -w
/home/ubuntu/jenkins/remoting/jarCache
Feb 26, 2025 6:55:57 PM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir
INFO: Using /home/ubuntu/jenkins/remoting as a remoting work directory
Feb 26, 2025 6:55:57 PM org.jenkinsci.remoting.engine.WorkDirManager setupLogging
INFO: Both error and output logs will be printed to /home/ubuntu/jenkins/remoting
<===[JENKINS REMOTING CAPACITY]===>channel started
Remoting version: 3283.v92c105e0f819
Launcher: SSHLauncher
Communication Protocol: Standard in/out
This is a Unix agent
Agent successfully connected and online
```

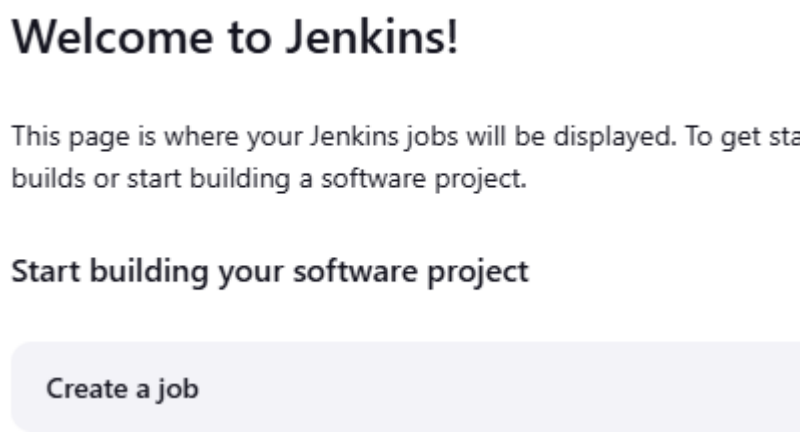
When we click on nodes, we can see both are in sync

Nodes

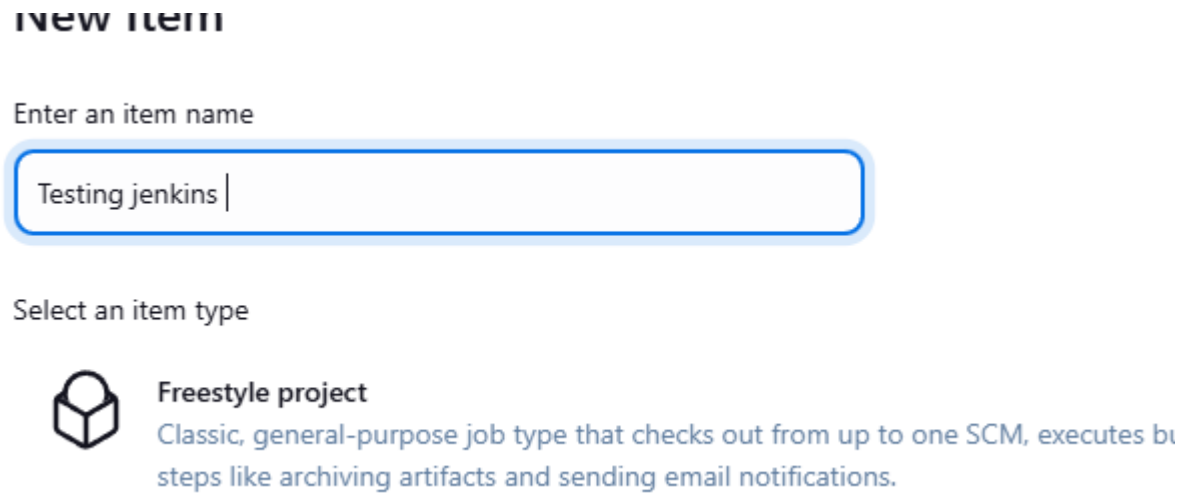
S	Name ↓	Architecture	Clock Difference	Free Disk Space
	Built-In Node	Linux (amd64)	In sync	15.35 GiB
	ubuntuworkernode	Linux (amd64)	In sync	15.79 GiB
	last checked	**2 min 47 sec**	**2 min 47 sec**	**2 min 47 sec**

We will now run a small test job that we can run on server.

Click on Dashboard – Click on + sign to create new job



Type as testing jenkins- Select Free style project. Click ok

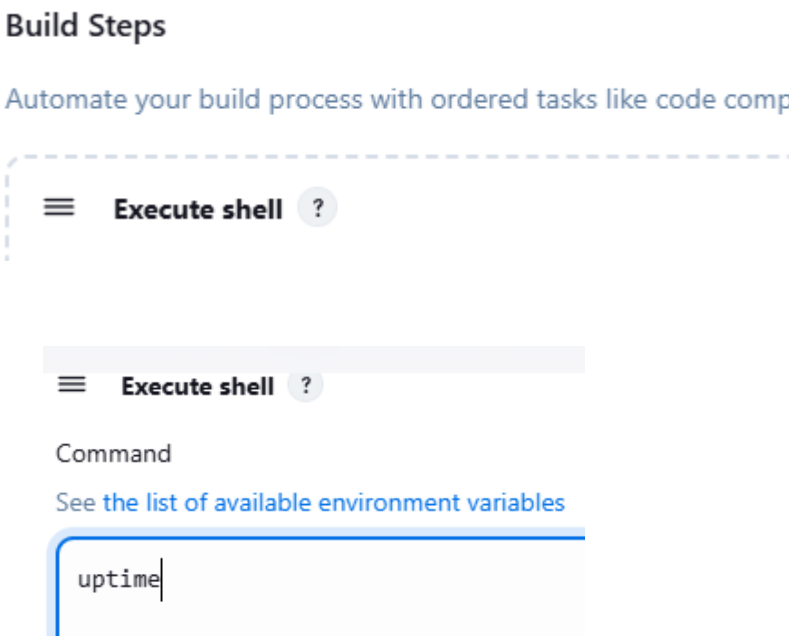


Select Build steps select option Execute Shell.

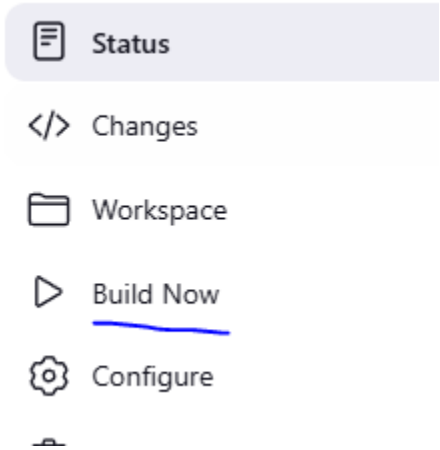
Here we can execute commands.

For this job we are selecting **uptime**

Click Save

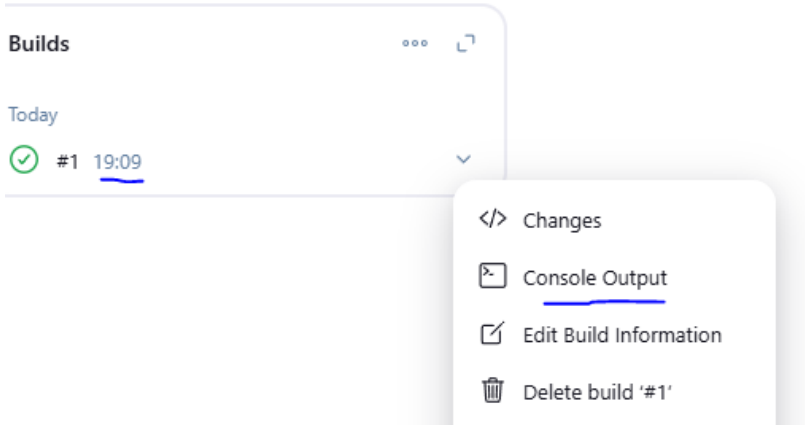


After clicking on Save, we can click **Build Now** option



We can now see our job is successful and node is working fine.

We can click on console output to see the status.



Console is now showing good and shows uptime of server

Console Output

```
Started by user santhosh
Running as SYSTEM
Building on the built-in node in workspace /var/lib/jenkins/workspace/Testing jenkins
[Testing jenkins] $ /bin/sh -xe /tmp/jenkins14812687760815393334.sh
+ uptime
 19:09:35 up  2:53,  2 users,  load average: 0.00, 0.01, 0.00
Finished: SUCCESS
```

END of Document