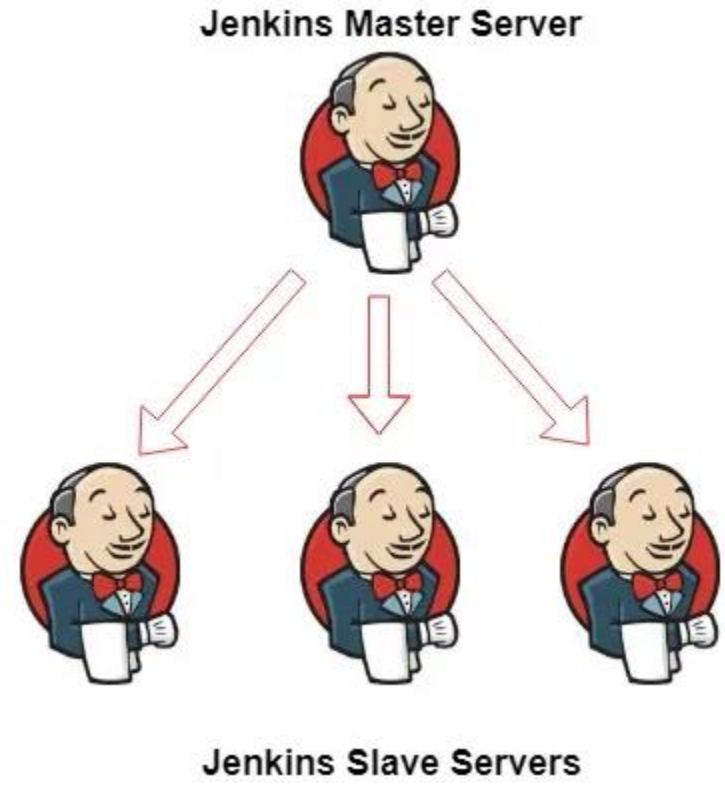
Day 28 Task: Jenkins Agents

Jenkins uses a Master-Slave architecture to manage distributed builds. In this architecture, Master and Slave communicate through TCP/IP protocol.

To run heavy projects which gets build on a regular basis is not a good option.

In such a scenario, need to off load, load from Master by configuring more slaves.



Distributed builds are used to absorb extra load or to run specialized build jobs in a specific operating system or environments.

Jenkins slaves are generally required to provide the desired environment. They work on the basis of requests received from Jenkins master.

Jenkins Master (Server): -

Jenkins's server or master node holds all key configurations. Jenkins master server is like a control server that orchestrates all the workflow defined in the pipelines. For example,

- Scheduling build jobs.

- Dispatching builds to the slaves for the actual execution.
- Monitor the slaves (possibly taking them online and offline as required).
- Recording and presenting the build results.
- A Master instance of Jenkins can also execute build jobs directly.

Jenkins Agent(slave): -

An agent is typically a machine or container that connects to a Jenkins master and this agent that actually execute all the steps mentioned in a Job. When you create a Jenkins job, you have to assign an agent to it. Every agent has a label as a unique identifier.

When you trigger a Jenkins job from the master, the actual execution happens on the agent node that is configured in the job.

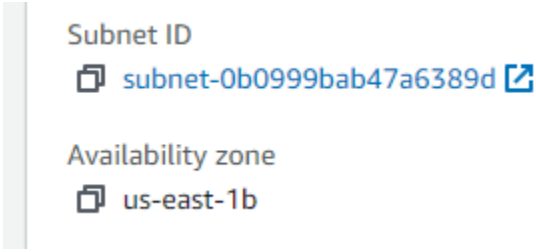
A single, monolithic Jenkins installation can work great for a small team with a relatively small number of projects. As your needs grow, however, it often becomes necessary to scale up. Jenkins provides a way to do this called “master to agent connection.” Instead of serving the Jenkins UI and running build jobs all on a single system, you can provide Jenkins with agents to handle the execution of jobs while the master serves the Jenkins UI and acts as a control node.

Full Jenkins installation on a slave is not required.

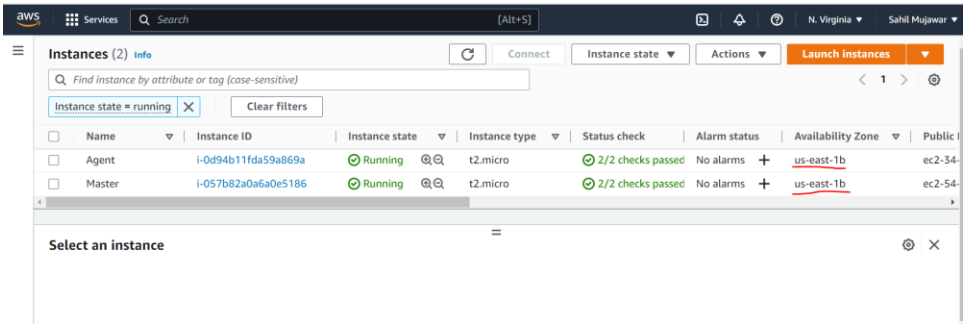
Follow below steps to run a project using Jenkins Agent: -

1] before starting connecting Jenkins agent to Jenkins master, first create a new EC2 Instance, same .as you create master one.

Note: - [Make sure Your Jenkins master **SubnetID and Availability** one is same as Jenkins slave.]



This SubnetID and Availability zone ,you can find in networking TAB.



2] After Creating an agent EC2 instance, you can make a connection between master and slave.

1. Go to master command prompt and type

ssh-keygen

After that **.ssh** folder is created, this folder contains your public and private key

```
ubuntu@ip-172-31-92-26:~$ cd /home/ubuntu/
ubuntu@ip-172-31-92-26:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_rsa
Your public key has been saved in /home/ubuntu/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:2Tv/bWF1wsEUS3P+td5Hp5hdbwCaQpN5YvN/gaqbDVE ubuntu@ip-172-31-92-26
The key's randomart image is:
+---[RSA 3072]----+
|             o=..|
|         o   .o= |
|        O E ....o|
|       o X o oo *|
|        S = . ooB|
|         o + +.O+|
|        . + + =.*|
|         = o . oo|
|        +.. .....|
+----[SHA256]-----+
ubuntu@ip-172-31-92-26:~$ cd .ssh
ubuntu@ip-172-31-92-26:~/.ssh$ ls
authorized_keys  id_rsa  id_rsa.pub
```

id_rsa ----------> private key

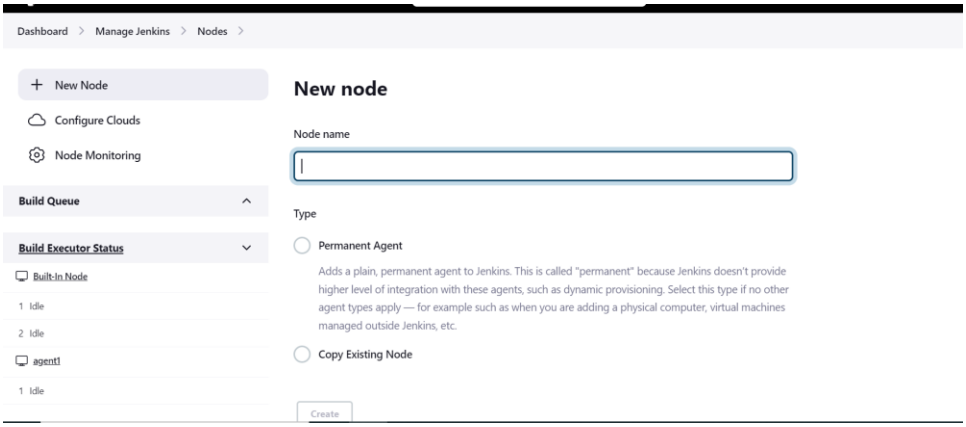
id_rsa.pub ----------> public key

2. Copy this master public key and paste into agent authorized key.

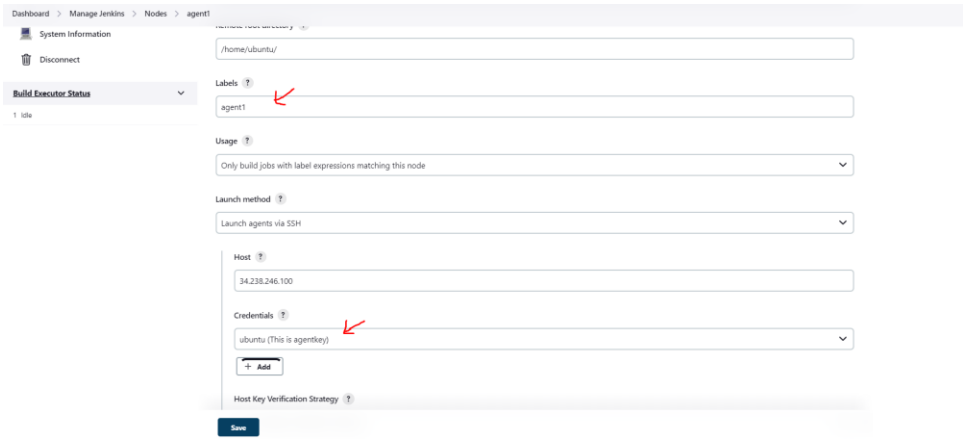
```
ubuntu@ip-172-31-81-198:~$ cd /home/ubuntu/
ubuntu@ip-172-31-81-198:~$ cd .ssh
ubuntu@ip-172-31-81-198:~/.ssh$ ls
authorized_keys
ubuntu@ip-172-31-81-198:~/.ssh$ vi authorized_keys
ubuntu@ip-172-31-81-198:~/.ssh$ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQD2ryao6nt1qD4nZuvvsYM/D3nHSjMg2mY9AfwUplfbbQcT+TDhAGiNxO9nmZRkCB9GXVxncCT944KU1Zl0mLfnUYV
ETm7h2tXp+/OUKz7VD6vu629o70R5ChfkEabdjVqMjsh6he+CQkFrbJiWInT0h/yTPKQpUrM5xcZ0h7fOMAtXLhoEJjI2tfv05ZquIO5lBZOZOisFPSfLxMgsEQg+u5
BL43Iczu8P1QUWWfLgYWjHGDvRIRKegRCi+9 Rushi-new-keypair

ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQC0JbDpS8AntxKkjSstfHhVo2xamm08m/pAU5YPbAZ5BJNlH4bwojDcbaeSOwgkJiOrpUSU8pDcbYywBqO2f4eu+xt
L83bUdesqcH9COA7JKCVQ2kmdYwTbhf+yMHq6+6a9woou3jWRWuB2C63SLjVrD0V4YmpJOjZK1I1OrcHLzg2kZiFqxpiXzKkJW1SWMqFcQQDoqqrfMDL64nSI+niafI
QAdO9oQjAloKmPjlwYR9XzSIvBX7crzGHOGsTGh1QPSAFCe5bWEYzVpn8cR99NxiQ1hu8l9QJSBGrOCJgYWfG4oNZfVNf0XqCaPL6Omk9pWijaEvSaGPK6Lit8U0lWG
VFE5rpVk+AkPjF36NWqYR5eubqbI0BpOdMk= ubuntu@ip-172-31-92-26
```

3. After that go to **Manage Jenkins** and click on **Manage nodes and clouds**

Click on New Node and type mode name



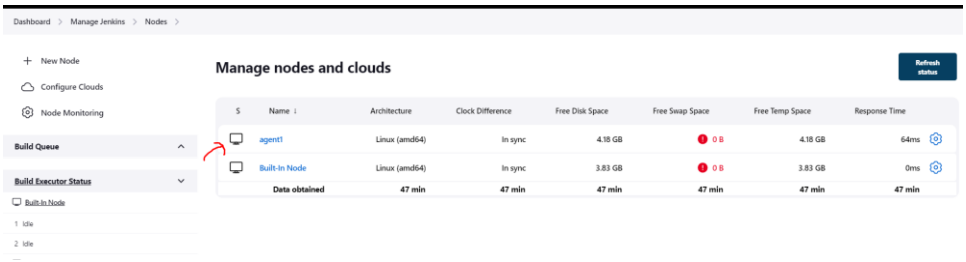
Also add the Labels and inside Launch method select Launch agents via ssh.



Inside Launch Method, put Host Ip address and Jenkins master Private key Using

SSH username and private key

It looks like below image.



4. After that you install java on Jenkins Agent and the java version is same as Jenkins master java version.

Also, for this project needs docker, so install docker and docker-compose on Jenkins agent.

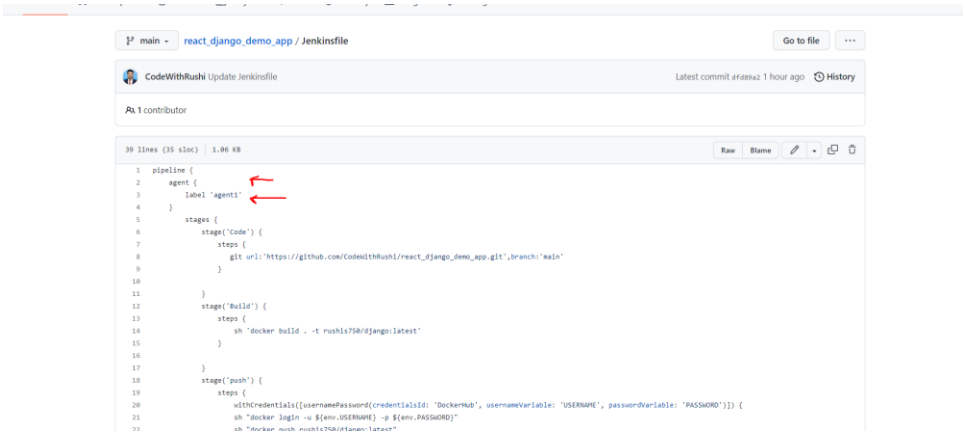
Give the docker permission to current user.

```
ubuntu@ip-172-31-81-198:~$ sudo usermod -a -G docker $USER
ubuntu@ip-172-31-81-198:~$ sudo reboot
```

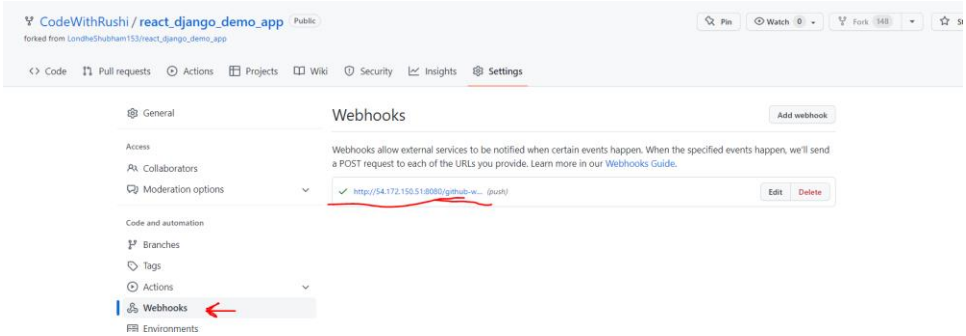
5. check the Jenkins agent log, your agent is successfully connected to master or not?

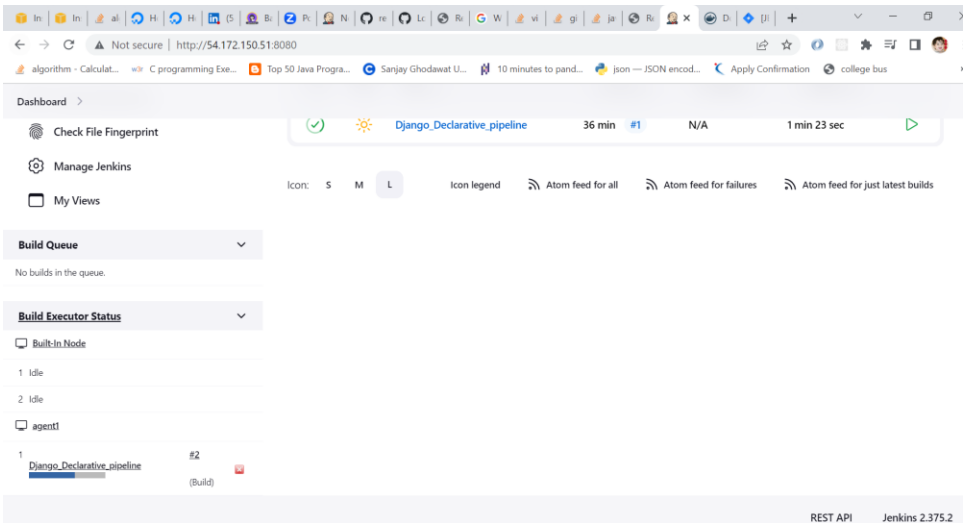
3] After Agent connecting to Master, add the label to Declarative pipeline.

This is the syntax to add label in Declarative pipeline. You can add label in Jenkinsfile.

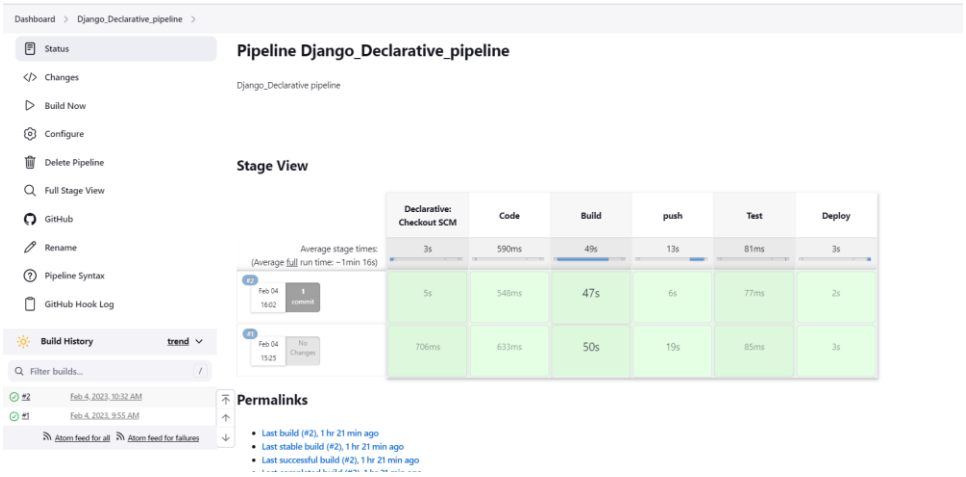


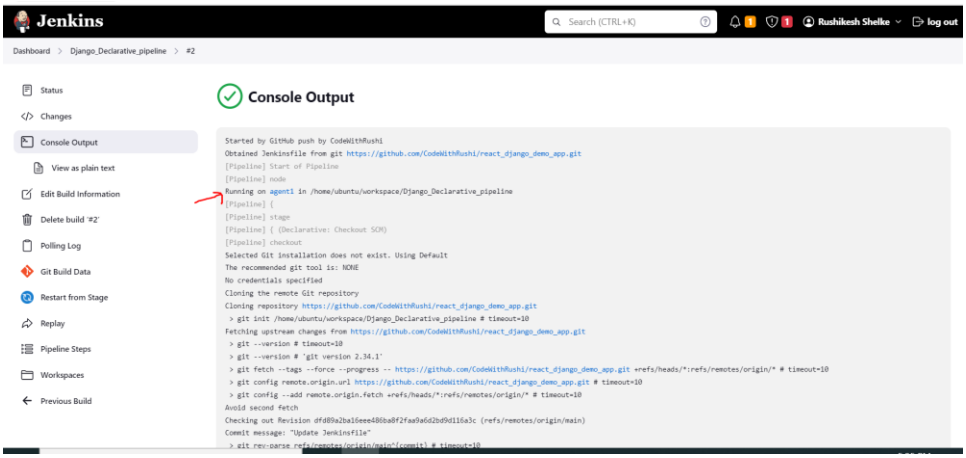
And do the commit. Using webhooks it's automatically starting the build on Jenkins UI.





4] After Successfully Run the pipeline, check the logs and full view of pipeline





5] Add your Project Port no into your Jenkins agent security Group. Copy the public Ip address and put the port no.

Your application is running