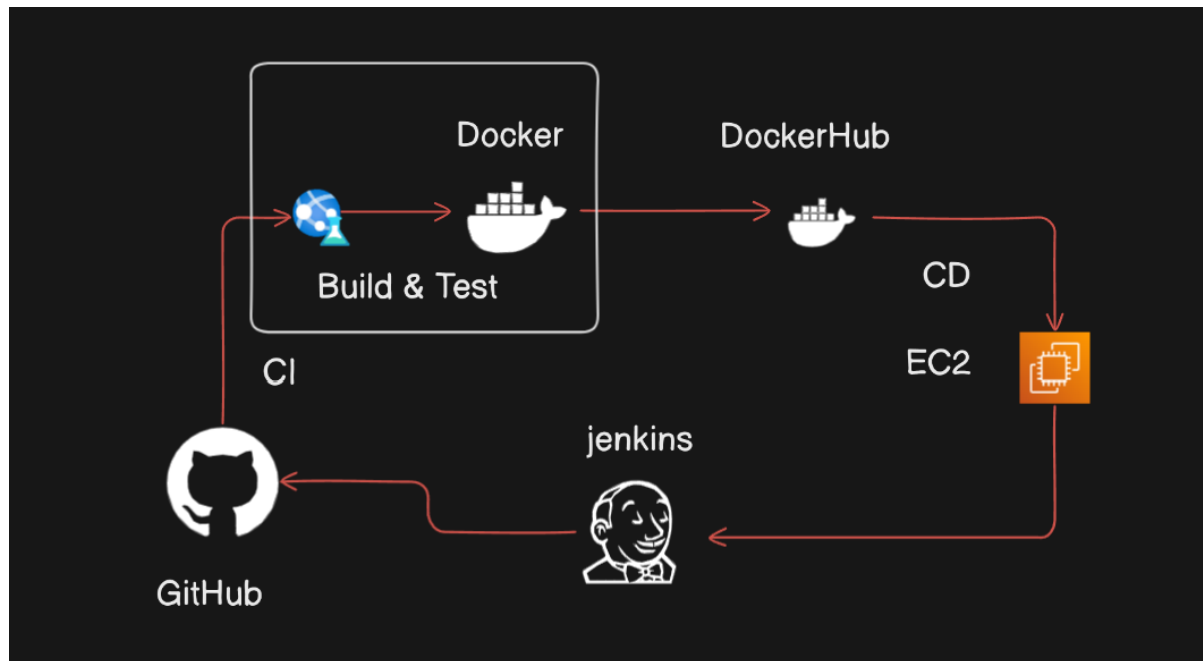


Jenkins Declarative Pipeline with Docker

django-notes-app-CICD Project

Description:

This project involves setting up a continuous integration and continuous deployment (CI/CD) pipeline using GitHub for a Django application. The application will be containerized using docker and deployed to an AWS EC2 instance



- Code will be on GitHub repository
- This code will be tested and for that there should be a virtual environment for which docker is required which contains os, code, tests etc.
- After testing we have to push the container to dockerHub registry so that it should run on any machine by pulling the image on ec2 or k8's cluster
- To automate the entire process we use Jenkins as CI-CD automation tool
- GitHub url for ref: <https://github.com/abhi-255/django-notes-app.git>

Steps:

1) Launch ec2 instance.

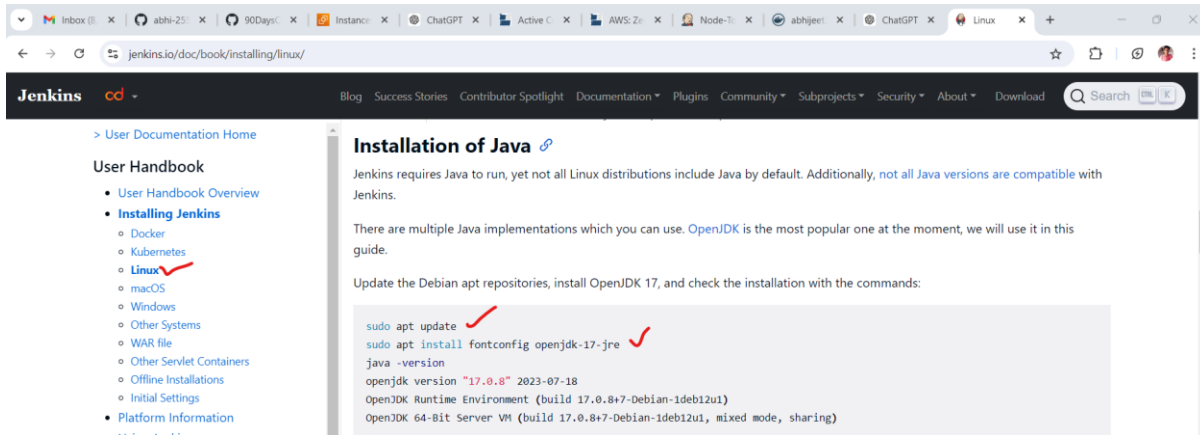
This is our Jenkins master so we need to install Jenkins on this server

2) Jenkins Installation:

First need to Install java

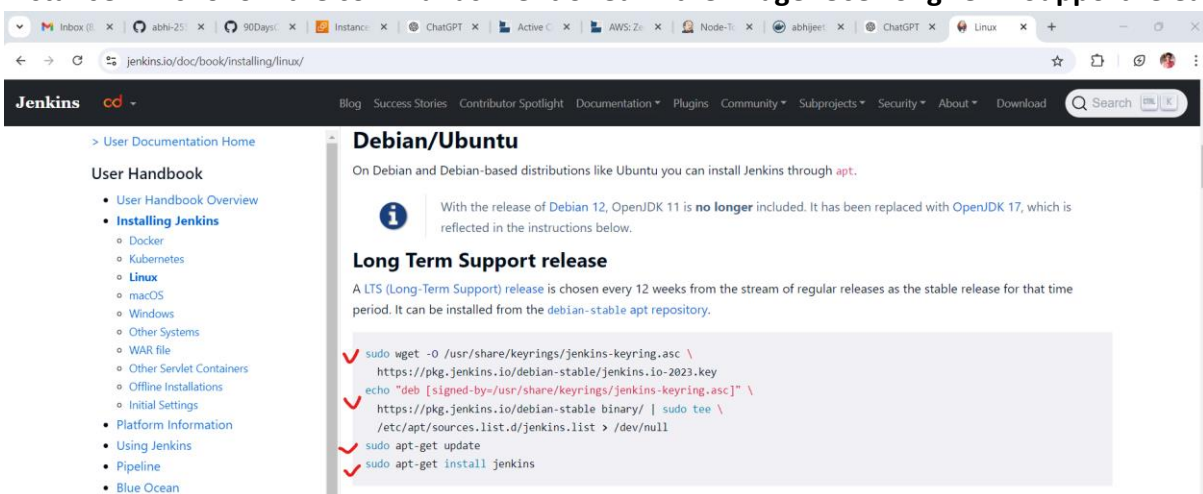
Search for install Jenkins on ubuntu on google

Follow the commands marked in the image to install java:



sudo apt update
sudo apt install fontconfig openjdk-17-jre
java -version

3) Install Jenkins follow the commands mentioned in the image: Use Long Term Support release



Run the following commands:

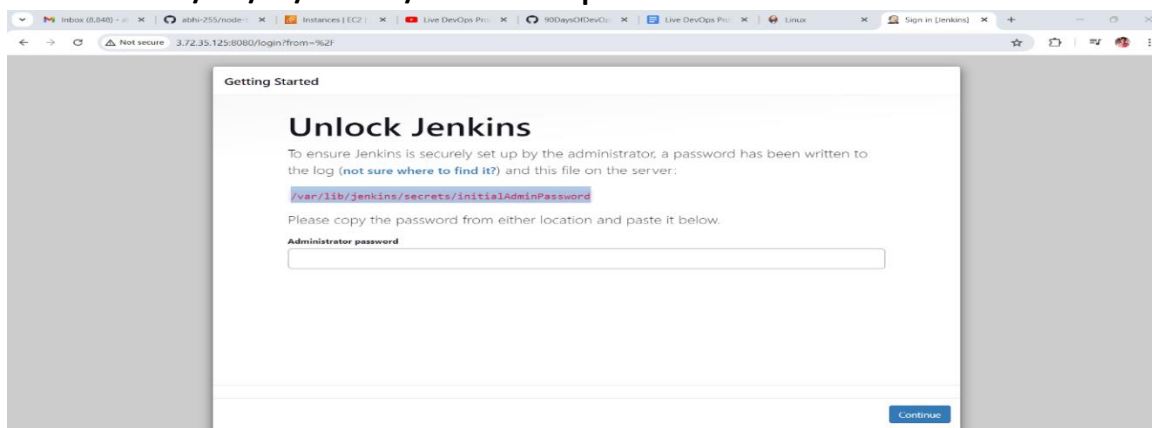
a) sudo systemctl status Jenkins

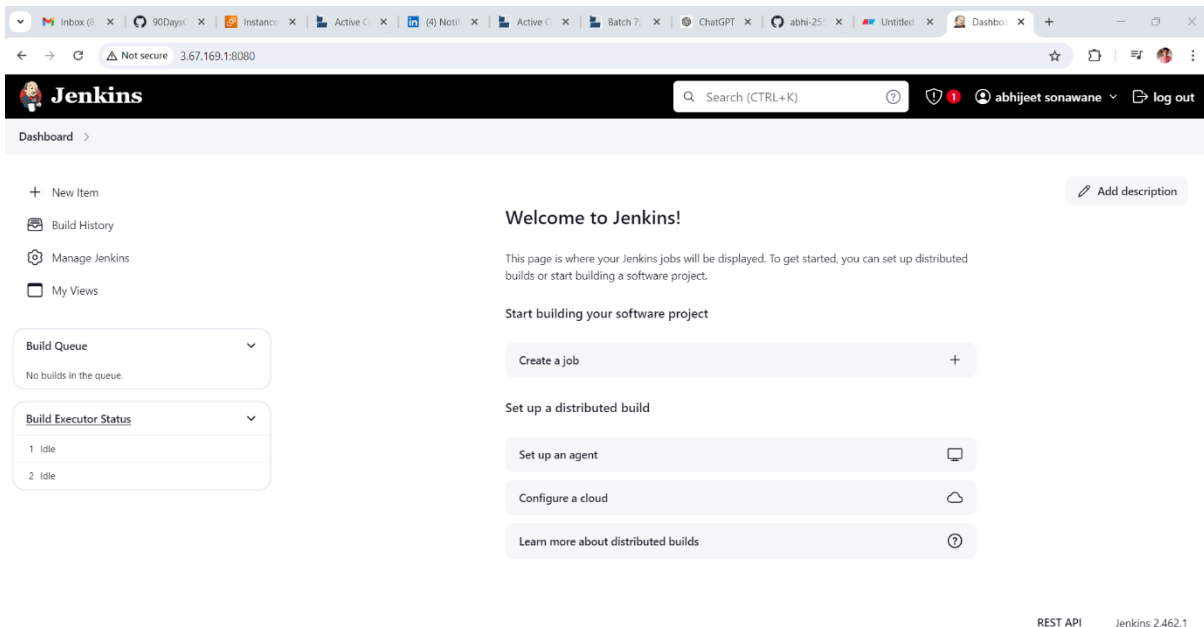
4) Jenkins setup:

a) To access Jenkins dashboard we need to add port 8080 into security group.

b) Serch the ip address of ec2:8080 and login to Jenkins dashboard

Following command will give you the paasword, enter it and login to dashboard
sudo cat /var/lib/Jenkins/initialAdminpassword





Jenkins master setup completed.

Now we have create a agent or worker node where we run our pipeline.

5) Agent Setup:

Launch ec2 instance.

This is our worker node

6) Now we have to create connection between master and worker node.

Master is accessing node so private key should be on master and public key should go to worker node

On Master:

- **ssh-keygen**
- **cd .ssh/**
- **ls**
- **cat id_ed25519.pub >>This public key we have to add in worker node**

```
ubuntu@ip-172-31-18-175:~$ 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:bmJGZPXHGy8CdzwXdYhey7K0oLS2l9cjX0JnND2Vu5w ubuntu@ip-172-31-18-175
The key's randomart image is:
+--[ED25519 256]--+
  .. .o+..+|
  ....=..o.000|
  oo ..*oo ++|
  o  . ..+ ooo|
  . S.   oo =|
  . + . ..o E|
  = + ooo + .|
  + o....o +|
  .. ..+
+----[SHA256]-----+
ubuntu@ip-172-31-18-175:~$ cd .ssh/
ubuntu@ip-172-31-18-175:~/.ssh$ ls
authorized_keys  id_ed25519  id_ed25519.pub
ubuntu@ip-172-31-18-175:~/.ssh$ cat id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIPW5jd002m0nYEkwhDl13r6CAp/7yijLLKk2utE2zj/8 ubuntu@ip-172-31-18-175
ubuntu@ip-172-31-18-175:~/.ssh$
```

Now go to Worker Node:

Follow the commands:

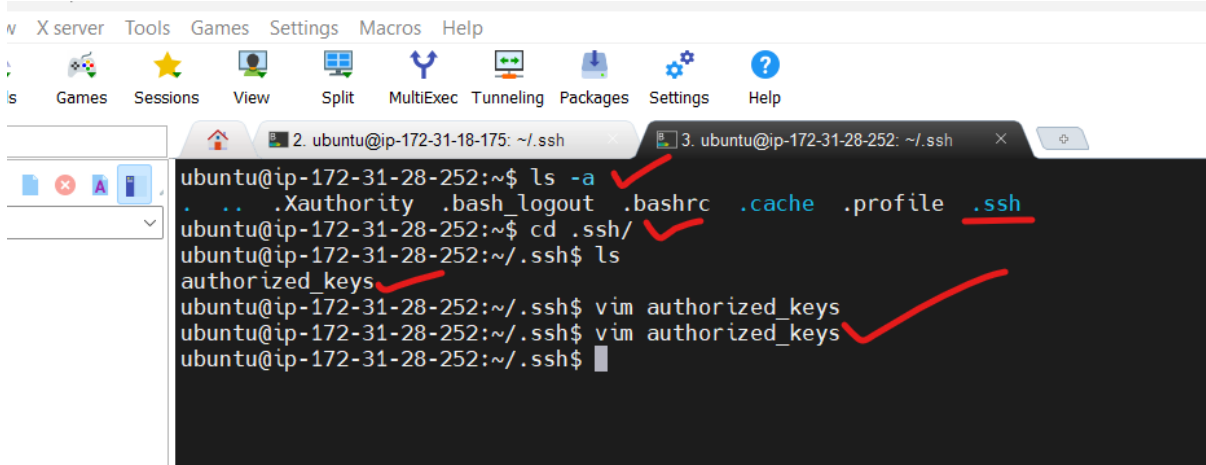
cd

ls -a

cd .ssh

ls

vim authorized_keys



```
ubuntu@ip-172-31-28-252:~$ ls -a
.  ..  .Xauthority  .bash_logout  .bashrc  .cache  .profile  .ssh
ubuntu@ip-172-31-28-252:~$ cd .ssh/
ubuntu@ip-172-31-28-252:~/.ssh$ ls
authorized_keys
ubuntu@ip-172-31-28-252:~/.ssh$ vim authorized_keys
ubuntu@ip-172-31-28-252:~/.ssh$ vim authorized_keys
ubuntu@ip-172-31-28-252:~/.ssh$
```

Add the public key of master here:



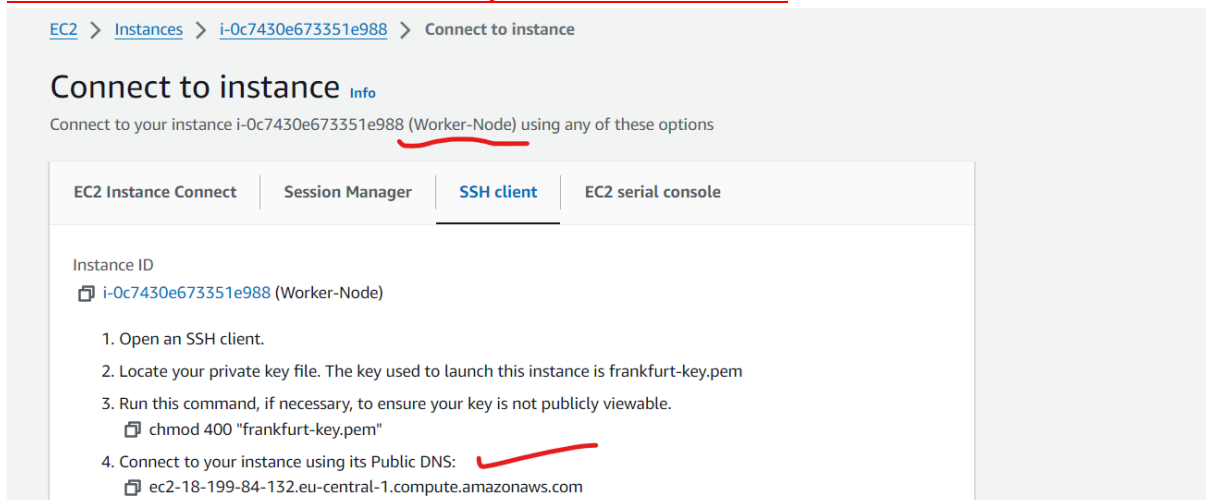
```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCSyBm6uszBmk705L61L1e9TS1uFCfMfMee80:cg4js8M6/mqYm84gd24nX7fuhhTirDyylpB8tD17n6qa9SInM9toYmoXhUgeUcCocL/2098vR9mg9vtinU85CoTtECcsEqp+50r2wKcJLmmWucsmMxeA399sjfg/K1G51ov0voVyyxlUK68jV5r2pS1BzL4gq0zpqjg8ajbs/V0kj2oghMqvZVg/s/luUmo77IEojZ30NqfUj115o519ENS24M2p49v5TR1Ra8WdJ9ZE6Bw8pVls81fKTHEzLl9dIm9Z+04JRzz9bKwwo/c+Cbv79/CET7PP1RejmE3L7uafbeo5 frankfurt-key
ubuntu@ip-172-31-28-252:~/.ssh$
```

Check master can access node or not using:

ssh -i ed25519 ubuntu@ec2-18-199-84-132.eu-central-1.compute.amazonaws.com

ed25519 this is master's private key

ec2-18-199-84-132.eu-central-1.compute.amazonaws.com this is worker node DNS



EC2 > Instances > i-0c7430e673351e988 > Connect to instance

Connect to instance Info

Connect to your instance i-0c7430e673351e988 (Worker-Node) using any of these options

EC2 Instance Connect	Session Manager	SSH client	EC2 serial console
<p>Instance ID</p> <p> i-0c7430e673351e988 (Worker-Node)</p> <ol style="list-style-type: none">1. Open an SSH client.2. Locate your private key file. The key used to launch this instance is frankfurt-key.pem3. Run this command, if necessary, to ensure your key is not publicly viewable. <code>chmod 400 "frankfurt-key.pem"</code>4. Connect to your instance using its Public DNS: <code>ec2-18-199-84-132.eu-central-1.compute.amazonaws.com</code>			

```
ubuntu@ip-172-31-18-175:~/.ssh$ cat id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIPW5jd002m0nYEkwHDL13r6CAp/7yijLLKk2utE2zj/8 ubuntu@ip-172-31-18-175
ubuntu@ip-172-31-18-175:~/.ssh$
ubuntu@ip-172-31-18-175:~/.ssh$
ubuntu@ip-172-31-18-175:~/.ssh$
ubuntu@ip-172-31-18-175:~/.ssh$ ssh -i id_ed25519 ubuntu@ec2-18-199-84-132.eu-central-1.compute.amazonaws.com
The authenticity of host 'ec2-18-199-84-132.eu-central-1.compute.amazonaws.com (172.31.28.252)' can't be established.
ED25519 key fingerprint is SHA256:00Y58iFbkThwt2FuwG2oI11V5klFzKbgukm2LMruHXY.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-18-199-84-132.eu-central-1.compute.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1009-aws x86_64)

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

System information as of Wed Aug 7 18:38:41 UTC 2024

System load:  0.0          Processes:      109
Usage of /:   22.9% of 6.71GB   Users logged in: 1
Memory usage: 21%          IPv4 address for enX0: 172.31.28.252
Swap usage:   0%

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

Last login: Wed Aug 7 18:30:42 2024 from 103.171.186.75
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-28-252:~$
ubuntu@ip-172-31-28-252:~$
```

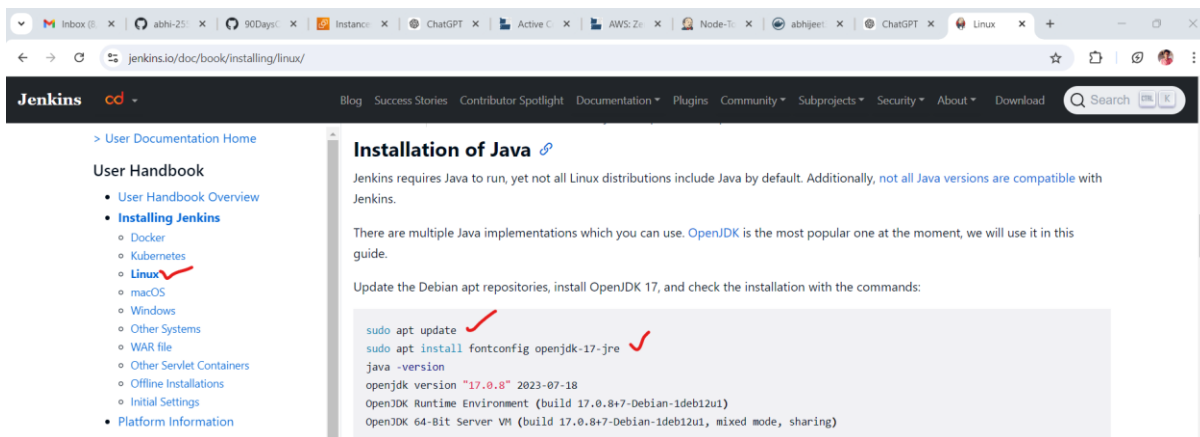
exit

7) On Worker node:

We have to Install java

Search for install Jenkins on ubuntu on google

Follow the commands marked in the image to install java:



sudo apt update

sudo apt install fontconfig openjdk-17-jre

java -version

Till here server to server connection is done.

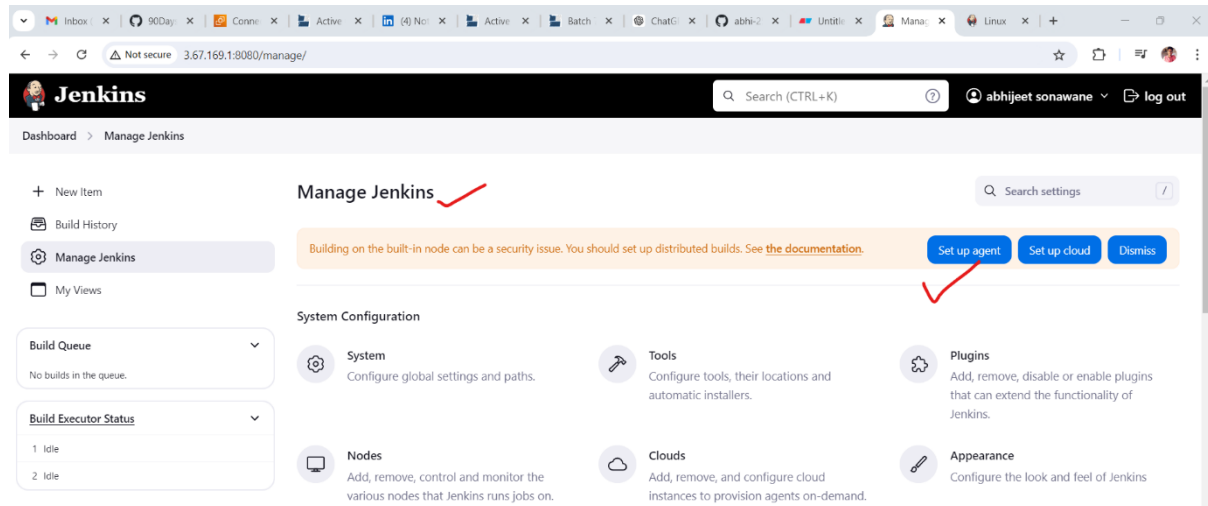
Now we have to do server to Jenkins connection. For that the private key of master needs to be added in Jenkins.

To achieve this we have to create agent on Jenkins dashboard.

Go to Jenkins dashboard.

Follo the path:

Manage Jenkins >> set up agent



Jenkins

Dashboard > Manage Jenkins

+ New Item

Build History

Manage Jenkins

My Views

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Manage Jenkins

Building on the built-in node can be a security issue. You should set up distributed builds. See the documentation.

Set up agent

Set up cloud

Dismiss

System Configuration

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.

Nodes

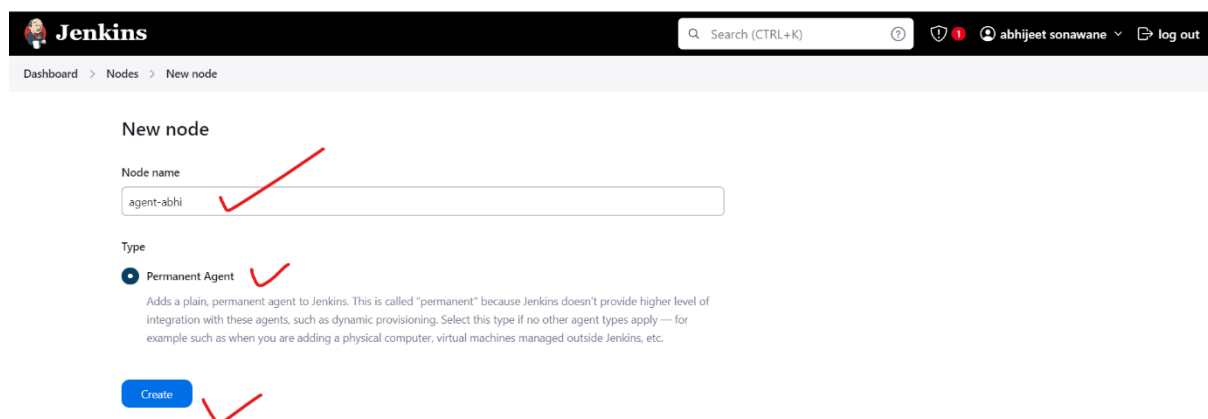
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

Clouds

Add, remove, and configure cloud instances to provision agents on-demand.

Appearance

Configure the look and feel of Jenkins



Jenkins

Dashboard > Nodes > New node

New node

Node name

agent-abhi

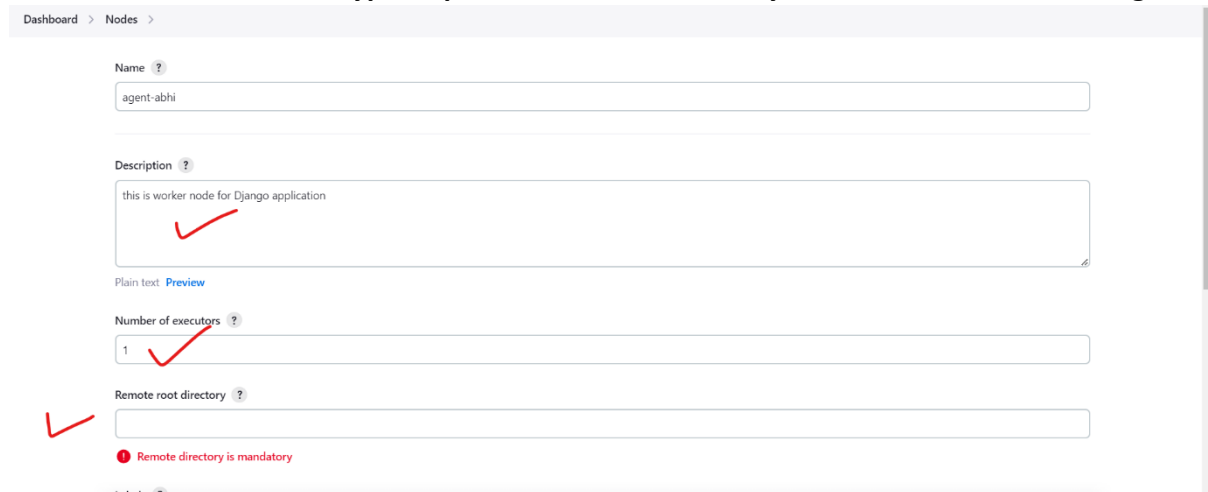
Type

☒ Permanent Agent

Adds a plain, permanent agent to Jenkins. This is called "permanent" because Jenkins doesn't provide higher level of integration with these agents, such as dynamic provisioning. Select this type if no other agent types apply — for example such as when you are adding a physical computer, virtual machines managed outside Jenkins, etc.

Create

Provide name and select type as permanent then create, you will redirect to following:



Dashboard > Nodes >

Name

agent-abhi

Description

this is worker node for Django application

Plain text Preview

Number of executors

1

Remote root directory

Remote directory is mandatory

For remote root directory path:

We need to create a directory on worker node.

mkdir apps

cd apps

pwd

/home/ubuntu/apps -----this path we have to add in above section

Dashboard > Nodes >

Remote root directory ?
/home/ubuntu/apps

Labels ?
dev **IMP**

Usage ?
Use this node as much as possible

Launch method ?
Launch agents via SSH

Host ? **node public ip**
18.199.84.132

Credentials ?
- none -
+ Add

The selected credentials cannot be found

Add credentials so that Jenkins can connect to this server:

Kind
SSH Username with private key

Scope ?
Global (Jenkins, nodes, items, all child items, etc)

ID ?
node-abhi-ssh-from-ubuntu

Description ?
this is a key to connect agent-abhi

Username
ubuntu

☐ Treat username as secret ?

Now we have to add private key of master here.

Go to master node and run> cat id_ed25519

```

Connection to ec2-18-199-84-132.eu-central-1.compute.amazonaws.com closed.
ubuntu@ip-172-31-18-175:~/.ssh$ cat id_ed25519
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAABG5vbmUAAAABbm9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZW
QyNTUxOQAAACD1uY3Tjtpjp2BJMIQ5dd6+ggKf+8ooyyyNrrRNs4//AAAAKDELBPqxCT
6gAAAAAtzc2gtZWQyNTUxOQAAACD1uY3Tjtpjp2BJMIQ5dd6+ggKf+8ooyyyNrrRNs4//A
AAED7FG0DFZuKKFpK5EG6FUmADJvf8yIsq0juBzbERRBtVPW5jd002m0nYEkwDl13r6C
Ap/7yijLLKk2utE2zj/8AAAAF3VidW50dUBpcC0xNzItMTZtMTgtMTc1AQIDBAUG
-----END OPENSSH PRIVATE KEY-----
ubuntu@ip-172-31-18-175:~/.ssh$
ubuntu@ip-172-31-18-175:~/.ssh$
ubuntu@ip-172-31-18-175:~/.ssh$

```

Copy the above private key and paste in node configurations:

Jenkins Credentials Provider: Jenkins

☐ Treat username as secret ?

Private Key ✓

☒ Enter directly ✓

Key

```
6gAAAAZC2gtZWQVNIUXOQAAACD1uY3IjTpj2BjMlQ5dd6+ggKf+8ooyypNHrRHS4 / A
AAED7FG0DFZuKkFpk5EG6FumaDjvf8yIsq0juBzbERRBTVPW5jd002m0NYEkwhD113r6C
Ap/7yiJLLKk2utEzZj/8AAAAF3V1dw50dUBpcC0xNzITMzETMTgTMTc1AQIDBAUG
-----END OPENSSH PRIVATE KEY-----
```

Passphrase

Cancel Add ✓

Select the credentials now:

Dashboard > Nodes >

Credentials ?

- none -

- none -

ubuntu (this is a key to connect agent-abhi)

❌ The selected credentials cannot be found

Dashboard > Nodes >

Credentials ?

ubuntu (this is a key to connect agent-abhi)

+ Add

Host Key Verification Strategy ?

Non verifying Verification Strategy

Advanced

Availability ?

Keep this agent online as much as possible

Node Properties

☐ Disable deferred wipeout on this node ?

☐ Disk Space Monitoring Thresholds

☐ Environment variables

☐ Tool Locations

Save ✓

Refresh and check:

Search (CTRL+K) ?

! 1

abhiyeet sonawane

log out

Nodes

+ New Node Configure Monitors

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	agent-abhi ✓	Linux (amd64)	✓ In sync	4.46 GiB	❌ 0 B	4.46 GiB	26ms
	Built-In Node	Linux (amd64)	In sync	3.92 GiB	❌ 0 B	3.92 GiB	0ms
	last checked	0.6 sec	0.6 sec	0.61 sec	0.6 sec	0.6 sec	0.6 sec

8) On Worker node:

We have to Install docker and docker compose:

- `sudo apt-get install docker.io`
- `sudo apt-get install docker-compose-v2`
- `docker ps`

after running `docker ps` command there will be error as permission denied so we need to add user into docker group using following command:

- `sudo usermod -aG docker $USER`
- `sudo chown ubuntu /var/run/docker.sock`

`sudo reboot` and connect to server after few minutes

Now check `docker ps` command, there will be no error.

9) Create a job as pipeline

Stages will be:

`stage("clone code")`

`stage("build and test")`

`stage("push image to dockerhub")`

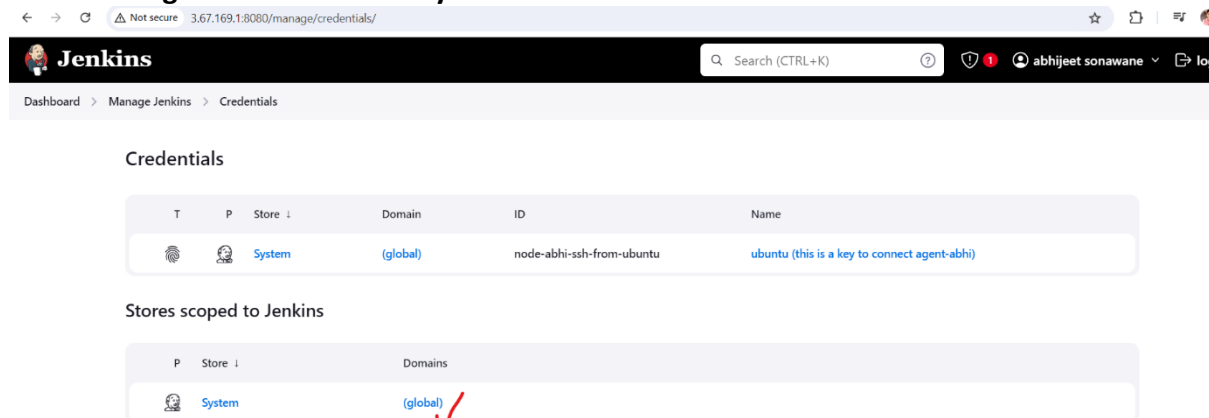
`stage("deploy")`

The first screenshot shows the Jenkins 'New Item' page. The 'Enter an item name' field contains 'Django-application'. Under 'Select an item type', the 'Pipeline' option is selected. The second screenshot shows the 'Configure' page for the 'Django-application' job. The 'General' tab is selected. The 'Description' field contains 'This is Django CI-CD project'. The 'GitHub project' checkbox is checked, and the 'Project url' field contains 'https://github.com/abhi-255/django-notes-app.git'.

Save the job

As we are building a image and pushing it to dockerhub we have to add docker hub credentials also.

Go to manage Jenkins >> security >> credentials

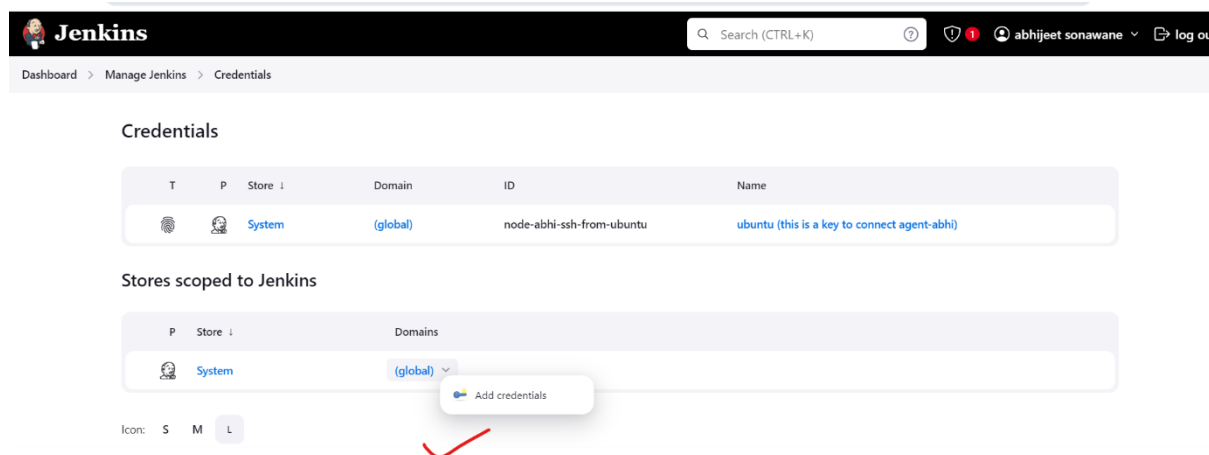


The screenshot shows the Jenkins 'Credentials' page. At the top, there's a navigation bar with 'Dashboard > Manage Jenkins > Credentials'. Below this, a table lists existing credentials. The table has columns: 'T' (Type), 'P' (Provider), 'Store' (Scope), 'Domain', 'ID', and 'Name'. One credential is listed: 'ubuntu (this is a key to connect agent-abhi)' with ID 'node-abhi-ssh-from-ubuntu' and Domain '(global)'. Below the table, there's a section 'Stores scoped to Jenkins' with a table showing 'System' as the provider and '(global)' as the domain. A red checkmark is placed next to the '(global)' domain in this section.

T	P	Store	Domain	ID	Name
	System		(global)	node-abhi-ssh-from-ubuntu	ubuntu (this is a key to connect agent-abhi)

P	Store	Domains
	System	(global) ✓

Click on global option and then add credentials



This screenshot is similar to the previous one, but it shows a dropdown menu that appears when the '(global)' domain is clicked in the 'Stores scoped to Jenkins' section. The dropdown menu contains the text 'Add credentials'. A red checkmark is placed below the dropdown menu.

P	Store	Domains
	System	(global) ✓ Add credentials

Generate a personal access token on dockerhub

Create access token

A personal access token is similar to a password except you can have many tokens and revoke access to each one at any time. [Learn more](#)

Access token description

jenkins-token

Access permissions

Read, Write, Delete

Read, Write, Delete tokens allow you to manage your repositories.

Cancel Generate

Access token description
jenkins-token

Access permissions
Read, Write, Delete

To use the access token from your Docker CLI client:

1. Run

```
$ docker login -u abhijeet311
```

Copy

2. At the password prompt, enter the personal access token.

```
docker_login_password: 5315M8qD1ZcLdU0Pfg
```

Copy

[Back to access tokens](#)

Copy the token

Scope ?
Global (Jenkins, nodes, items, all child items, etc) ▼

Username ?
abhijeet311

☐ Treat username as secret ?

Password ?
.....

ID ?
dockercred

Description ?
credentials for dockerhub





Create

Use the docker hub username,
Password= personal access token
Id= dockercred (Remember your id)
Click on create

Global credentials (unrestricted)

[+ Add Credentials](#)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description	
 node-abhi-ssh-from-ubuntu	ubuntu (this is a key to connect agent-abhi)	SSH Username with private key	this is a key to connect agent-abhi	
 dockercred	abhijeet311/***** (credentials for dockerhub)	Username with password	credentials for dockerhub	

Icon: S M L

Install stage view plugin for pipeline

Now configure the job again using pipeline attached in this document

You will get the output:

Dashboard > Django-application >

Jenkins

Search (CTRL+K) | abhijeet sonawane | log out

Django-application [Edit description](#)

This is Django CI-CD project

Stage View

Average stage times: (Average full run time: ~1min 10s)

	clone code	build and test	push image to dockerhub	deploy
Aug 08 01:30	6s	42s	13s	1s

Permalinks

Build History [trend](#)

Filter...

7 Aug 2024, 20:00

Search ip_addres:8000

18.199.84.132:8000/#/

My Notes

Notes 6

Hello Dosto
8/8/2024

Note #5
Does the time not update or something? tests
1/21/2023

Note #4
Looks like everything is working so far This is a test update ok ev...
1/20/2023

Note #3
This a new note now REACT is integrated with Django!
1/20/2023

Note #2

Check dockerhub registry:

hub.docker.com

dockerhub Explore **Repositories** Organizations

Search Docker Hub [ctrl+k](#)

abhijeet311 Search by repository name All Content [Create repository](#)

abhijeet311 / **notes-app-jenkins** [Contains: Image](#) Last pushed: 4 minutes ago [0](#) [7](#) [Public](#) [Scout inactive](#)

Dashboard > Django-application > Configuration

Configure

General

Advanced Project Options

Pipeline

```
1 pipeline {
2   agent {
3     node {
4       label "dev"
5     }
6   }
7   stages {
8     stage("clone code"){
9       steps{
10         git url: "https://github.com/abhi-255/django-notes-app.git", branch: "main"
11       }
12     }
13     stage("build and test"){
14       steps{
15         sh "whoami"
16         sh "docker build -t notes-app-jenkins:latest ."
17       }
18     }
19     stage("push image to dockerhub"){
20       steps{
21         withCredentials([
22           usernamePassword(
23             credentialsId: "dockercred",
24             passwordVariable: "dockerHubPass",
25             usernameVariable: "dockerHubUser"
26           )
27         ]){
28           sh "docker image tag notes-app-jenkins:latest ${env.dockerHubUser}/notes-app-jenkins:latest"
29           sh "docker login -u ${env.dockerHubUser} -p ${env.dockerHubPass}"
30           sh "docker push ${env.dockerHubUser}/notes-app-jenkins:latest"
31         }
32       }
33     }
34   }
35 }
36
37 stage("deploy"){
38   steps{
39     sh "docker compose up -d"
40   }
41 }
42
43 }
```