Table of Contents:

Table of Contents:

==========

Step -1 : Update the ubuntu index by using update command

$ sudo apt-get update

$ sudo apt install fontconfig openjdk-17-jre : Install the JAVA 17 version per Jenkins Document.

```
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$ sudo apt-get update
Hit:1 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Hit:3 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:5 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [297 kB]
Fetched 423 kB in 1s (645 kB/s)
Reading package lists... Done
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$ sudo apt install fontconfig openjdk-17-jre
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
```

Jenkins Document Link

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

$ java –version : To check the JAVA version.

```
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$ java --version
openjdk 17.0.11 2024-04-16
OpenJDK Runtime Environment (build 17.0.11+9-Ubuntu-1)
OpenJDK 64-Bit Server VM (build 17.0.11+9-Ubuntu-1, mixed mode, sharing)
ubuntu@ip-172-31-24-254:~$
```

==========

Step -2 : Use the Long term Support Release Command to install the Jenkins.

Long Term Support release

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

```
sudo wget -O /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 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 : Jenkins has been installed successfully, See below Capture.

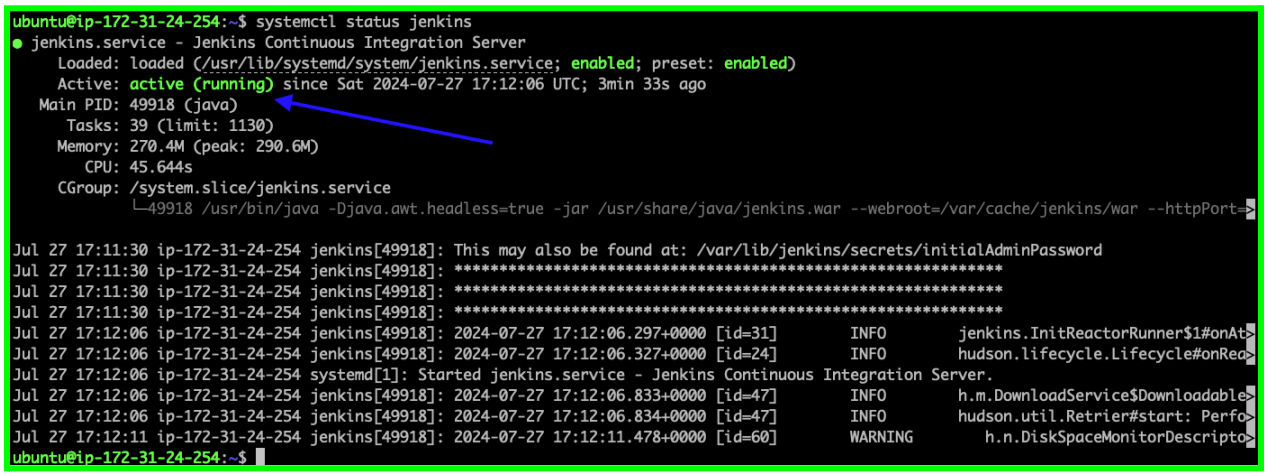
```
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$ sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update
sudo apt-get install jenkins
--2024-07-27 17:11:08--  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
Resolving pkg.jenkins.io (pkg.jenkins.io)... 146.75.122.133, 2a04:4e42:8d::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|146.75.122.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3175 (3.1K) [application/pgp-keys]
Saving to: '/usr/share/keyrings/jenkins-keyring.asc'

/usr/share/keyrings/jenkins-keyrin 100%[=========================================================>]   3.10K  --.-KB/s    in 0s

2024-07-27 17:11:08 (34.0 MB/s) - '/usr/share/keyrings/jenkins-keyring.asc' saved [3175/3175]

Hit:1 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Ign:4 https://pkg.jenkins.io/debian-stable binary/ InRelease
Get:5 https://pkg.jenkins.io/debian-stable binary/ Release [2044 B]
Get:6 https://pkg.jenkins.io/debian-stable binary/ Release.gpg [833 B]
Hit:7 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:8 https://pkg.jenkins.io/debian-stable binary/ Packages [27.3 kB]
```

$ jenkins --version : To Check the Jenkins Version.

```
ubuntu@ip-172-31-24-254:~$ jenkins --version
2.452.3
ubuntu@ip-172-31-24-254:~$
```

$ systemctl status jenkins : To check the Jenkins Service Status.
$ systemctl enable jenkins : Service will come up automatically after Linux Server Reboot, else we need to manually make it UP, by using below:

$ systemctl start jenkins : To Start the Jenkins Service in case it's not running.

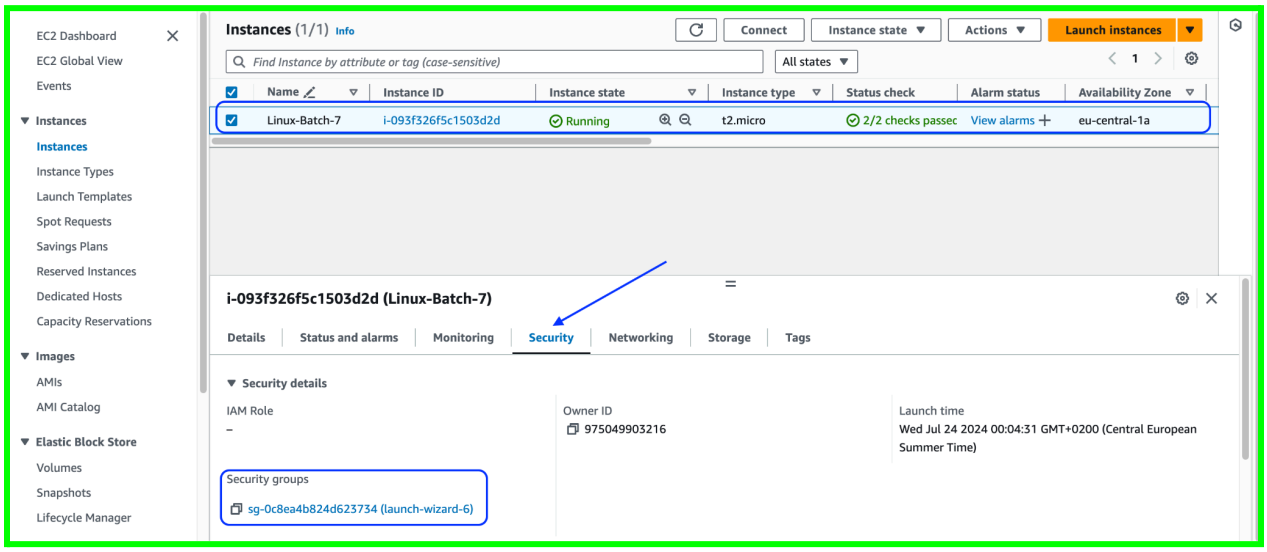


==========

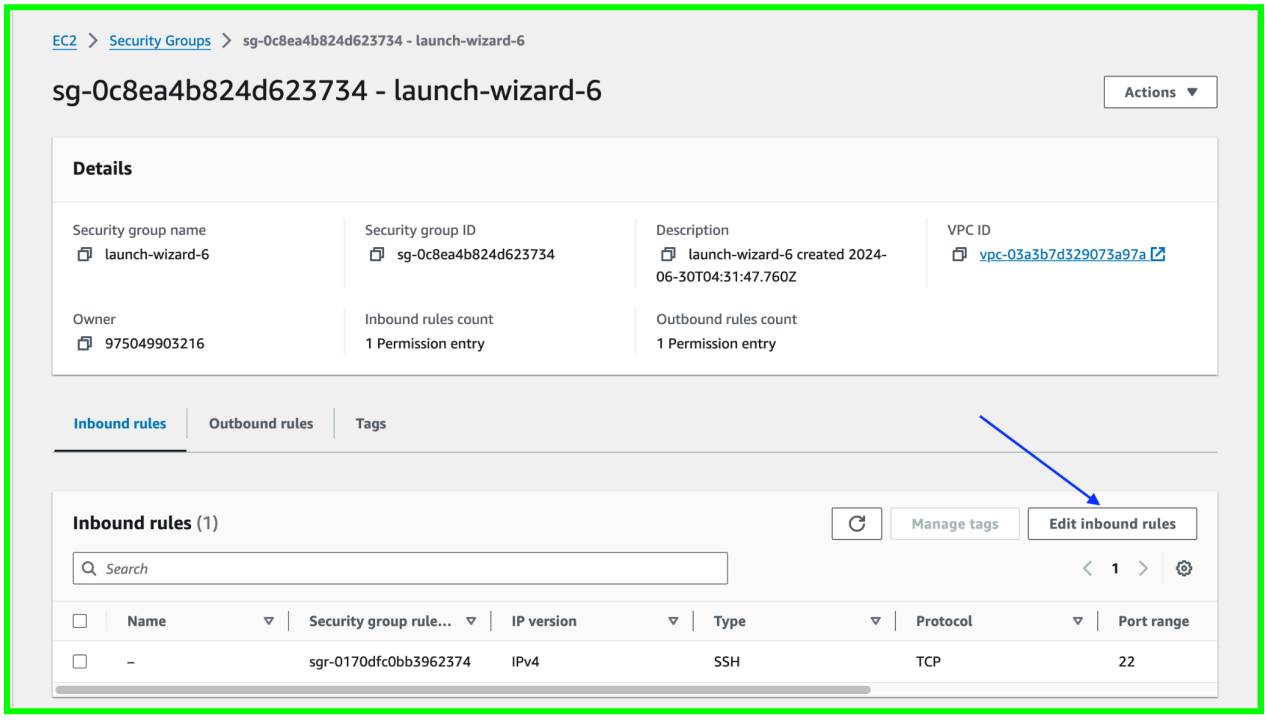
Step -3 : Login to Jenkins URL with Default Port 8080

If you are using EC2 Instance, please open the Port 8080 on Security Group → Inbound Rules

Go to EC2 Instance—>Select the Instance —> Click Security —> Click Security Group hyperLink

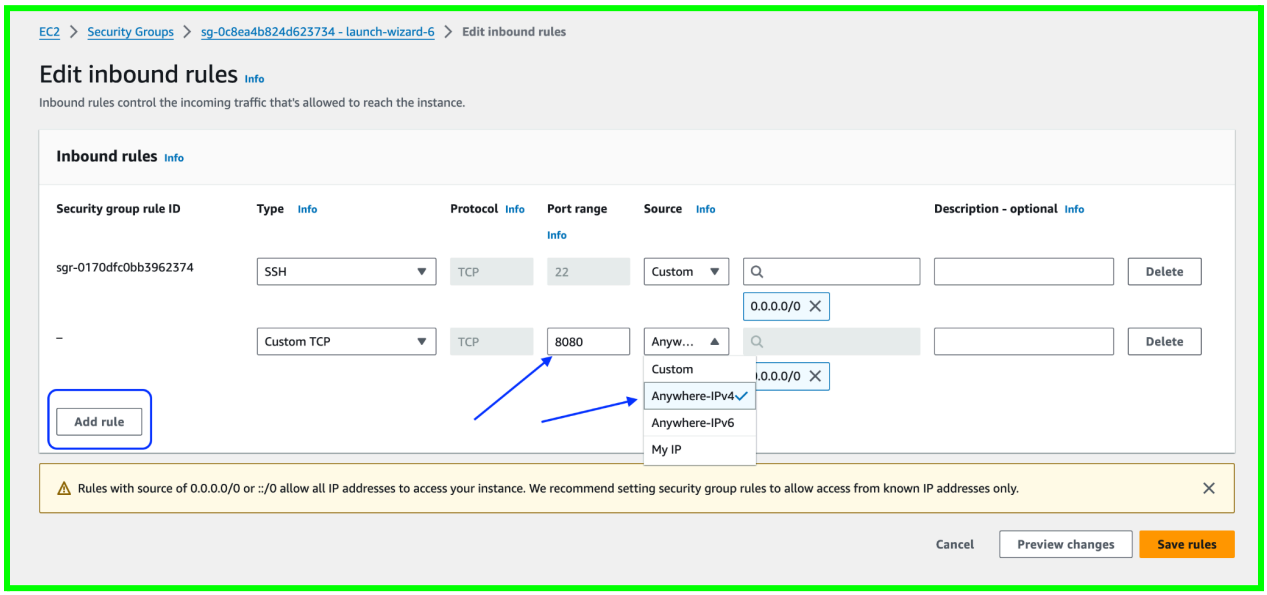


Security Group page will Open, Click —> Edit Inbound rules



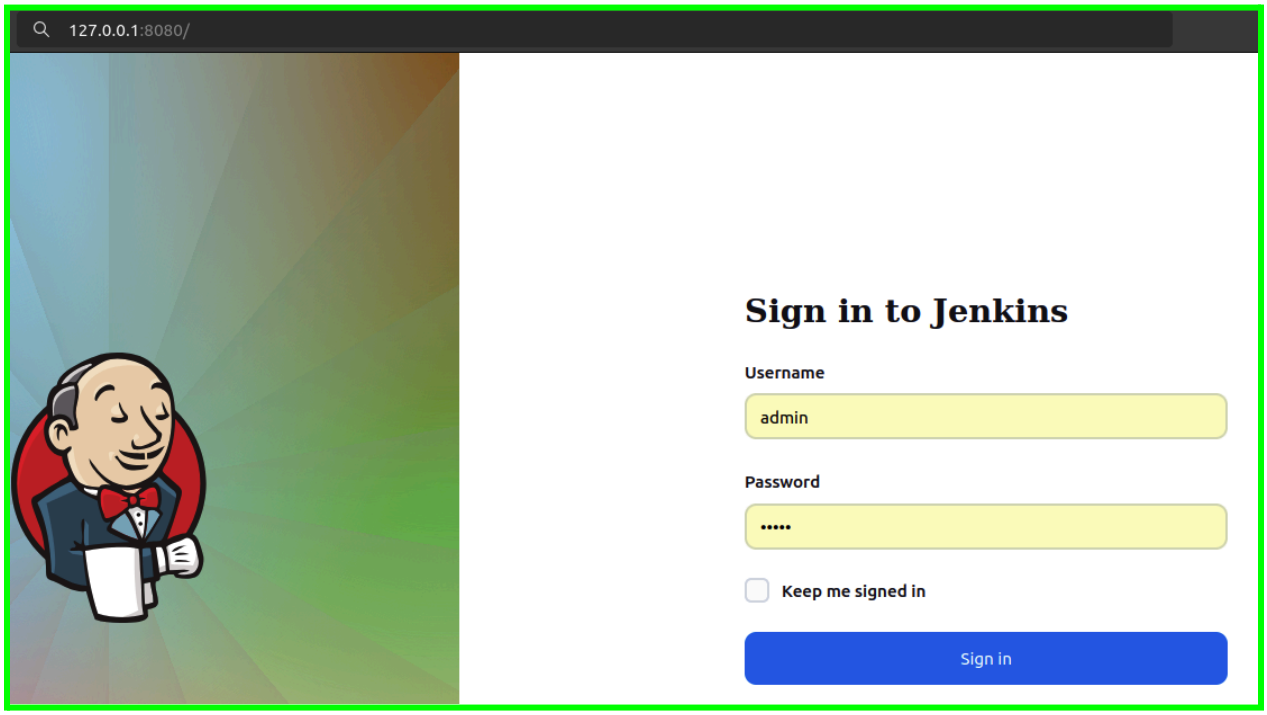
Click → Add Rule —> Custom TCP , Port Range 8080.

Note : Please use **“MyIP”** , for this document purpose i have selected “Anywhere-IPV4”



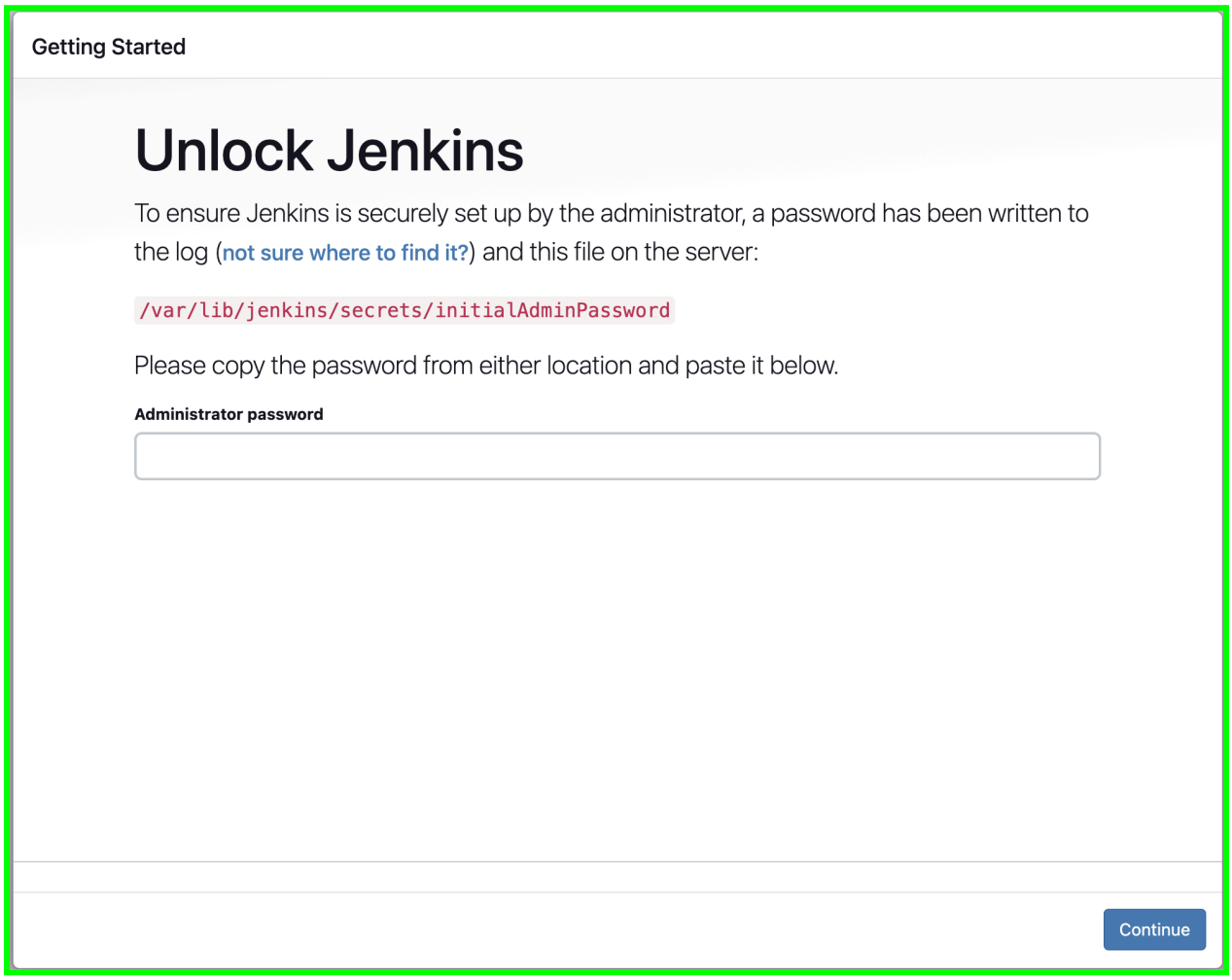
Note : You can install Jenkins on Virtual Box “Ubuntu Linux VM” in your laptop & default url will be in this format —> http://127.0.0.1:8080 **: For Local Ubuntu Linux Machine**

For EC2 Instance it will be in this format—> **http://<EC2PublicIP>:8080**



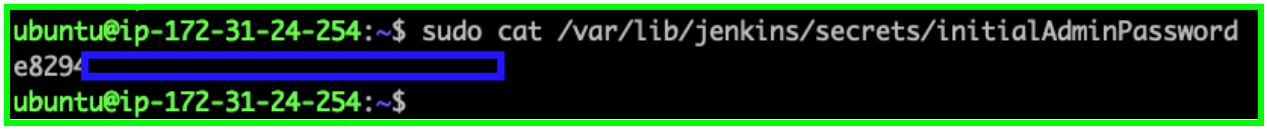
Once the above steps are completed, use your browser with below url
http://<yourEC2PublicIP>:8080

It will Open up with the below “Getting Started” Window.



Jenkins Initial Administrator Password

As per above Jenkin screen shot, please go to the path below in Linux EC2 Instance Machine.
`/var/lib/jenkins/secrets/initialAdminPassword`

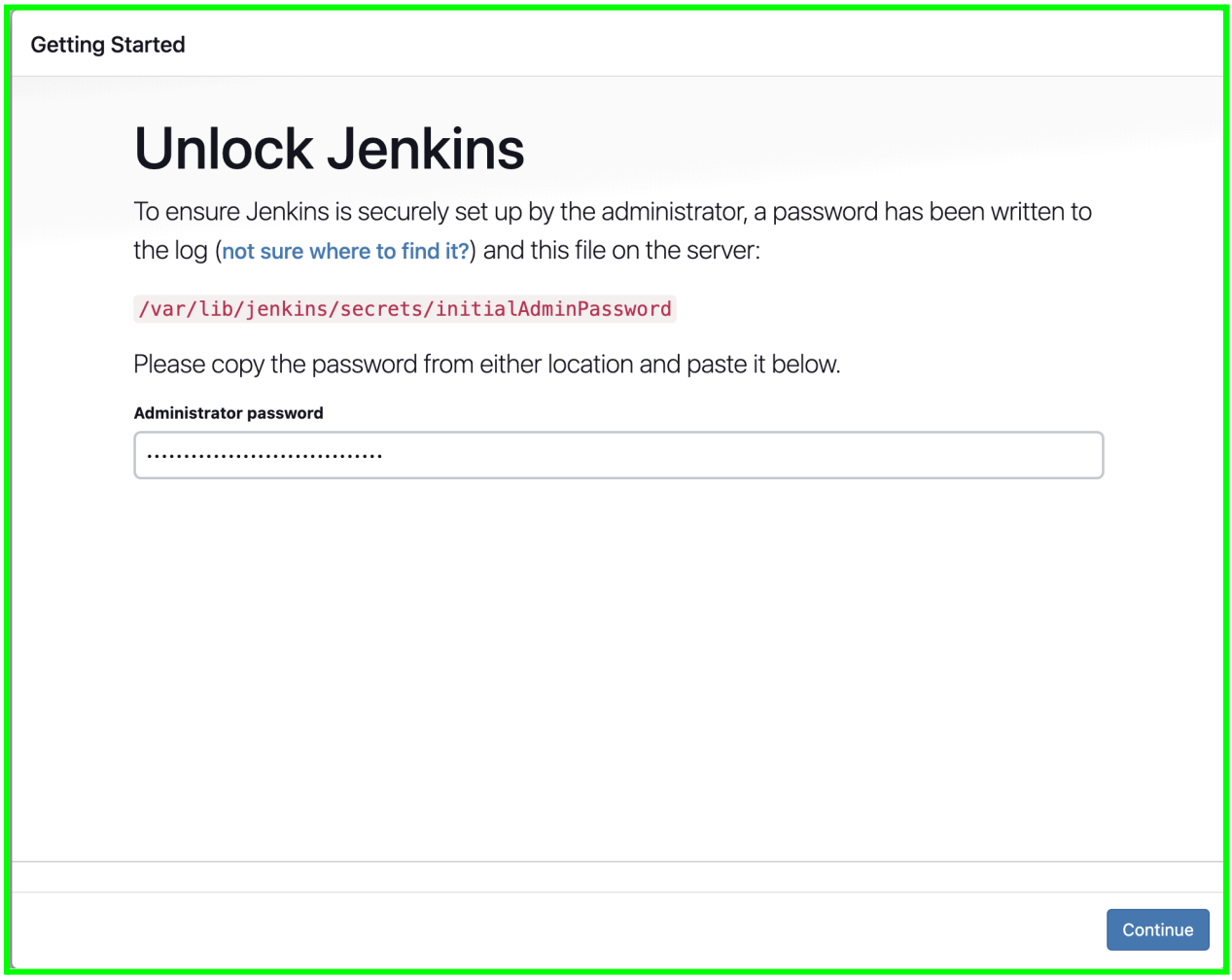


Copy the password & enter into the below step.

==========

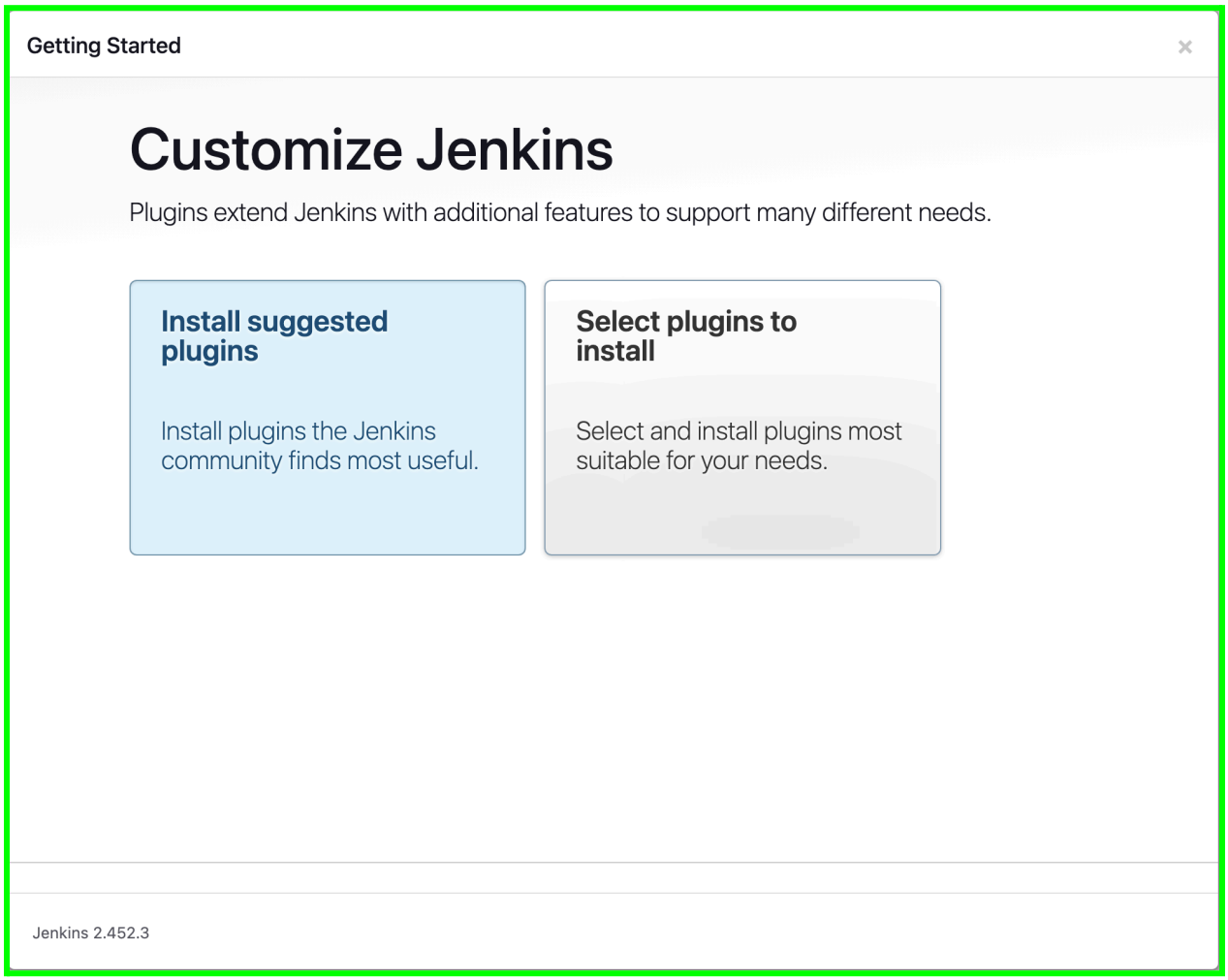
Step -4 : Setting up Jenkins

After Entering the Password, Click Continue



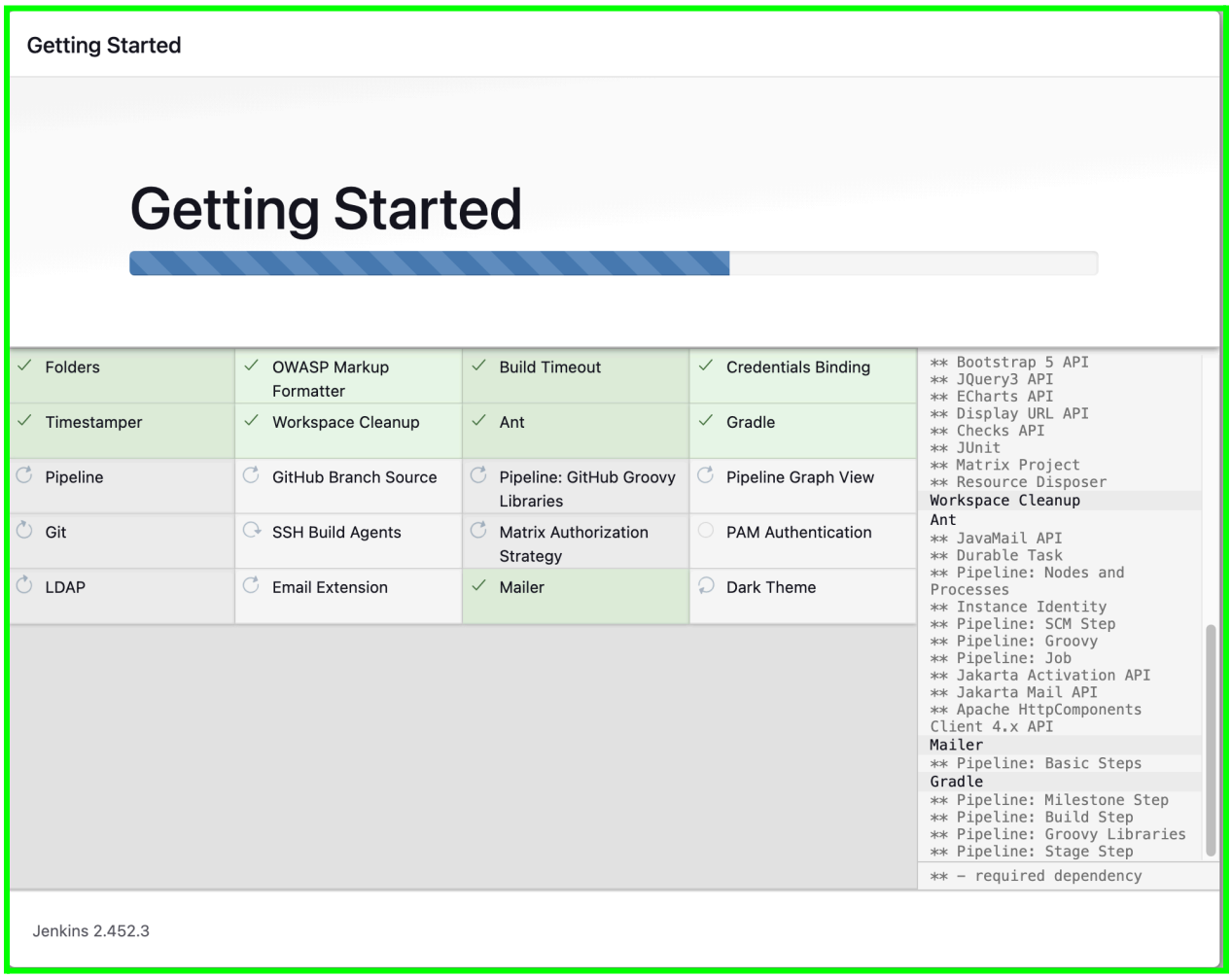
This will be the first screen, after hitting Continue , Please Select **“Install Suggested plugins”**

It will Start the Installation.



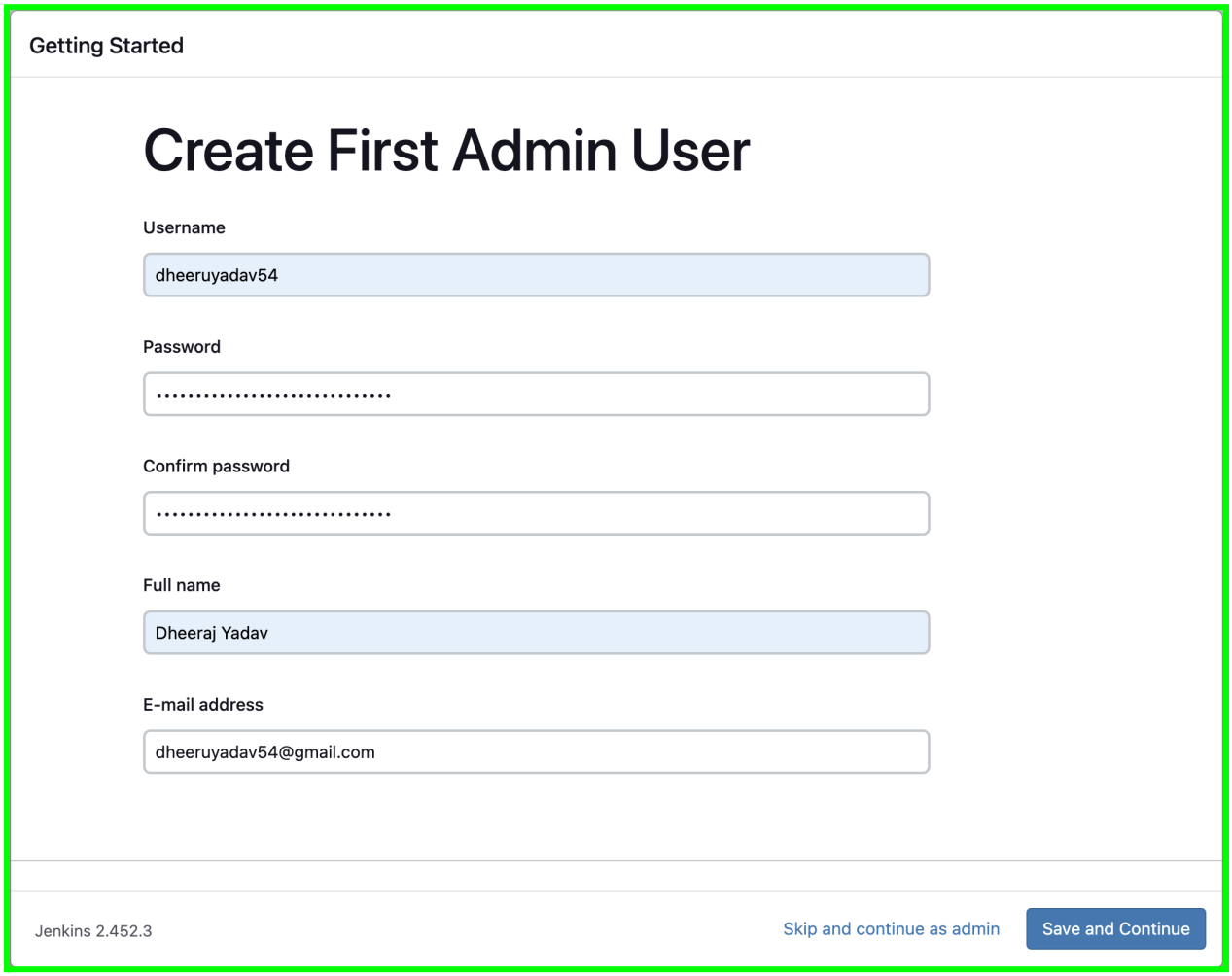
==========

Jenkins Installation has started



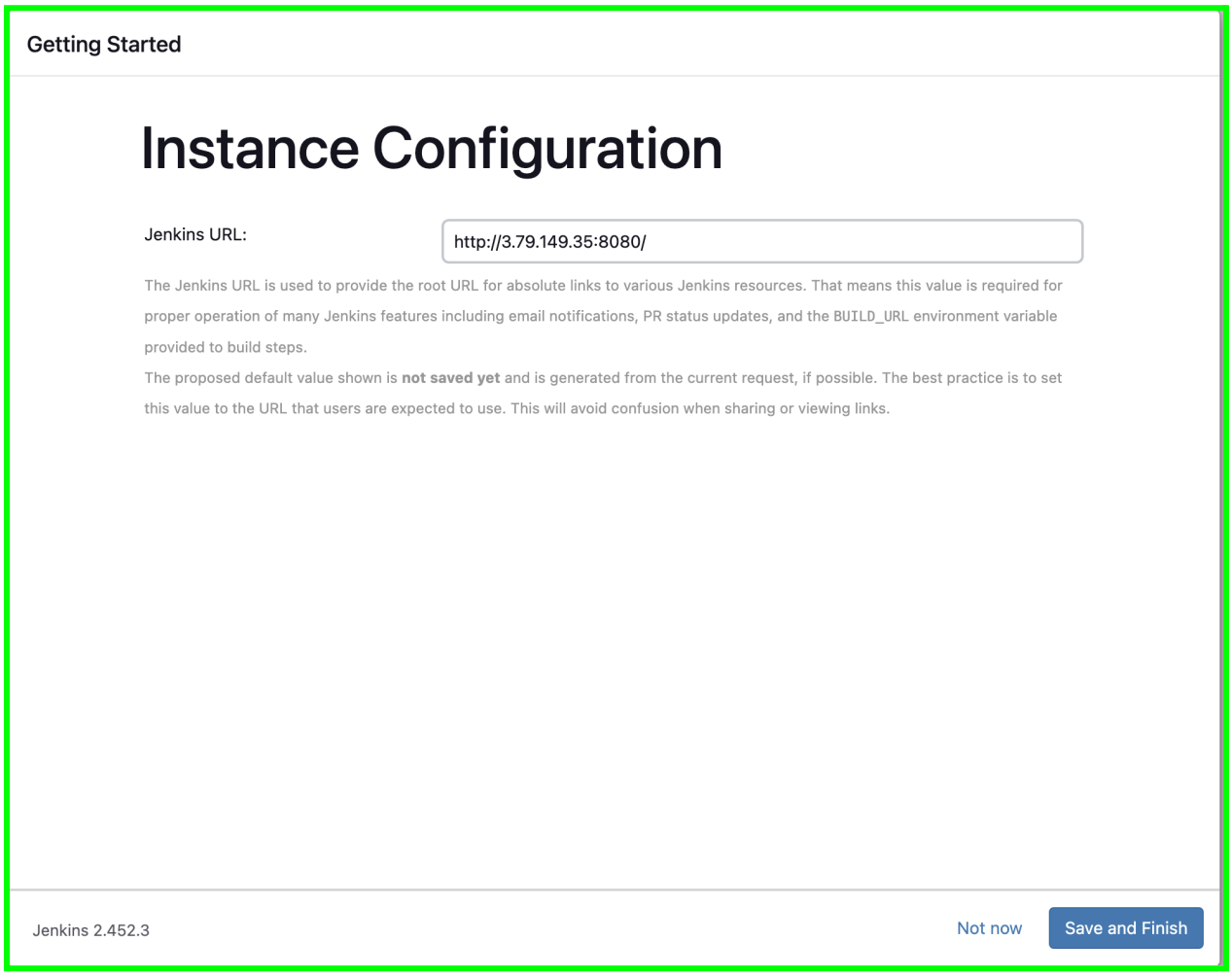
Fill the below details & hit Save and Continue

Username : You can use “admin” as well , but I have used “dheeruyadav54” as my username.

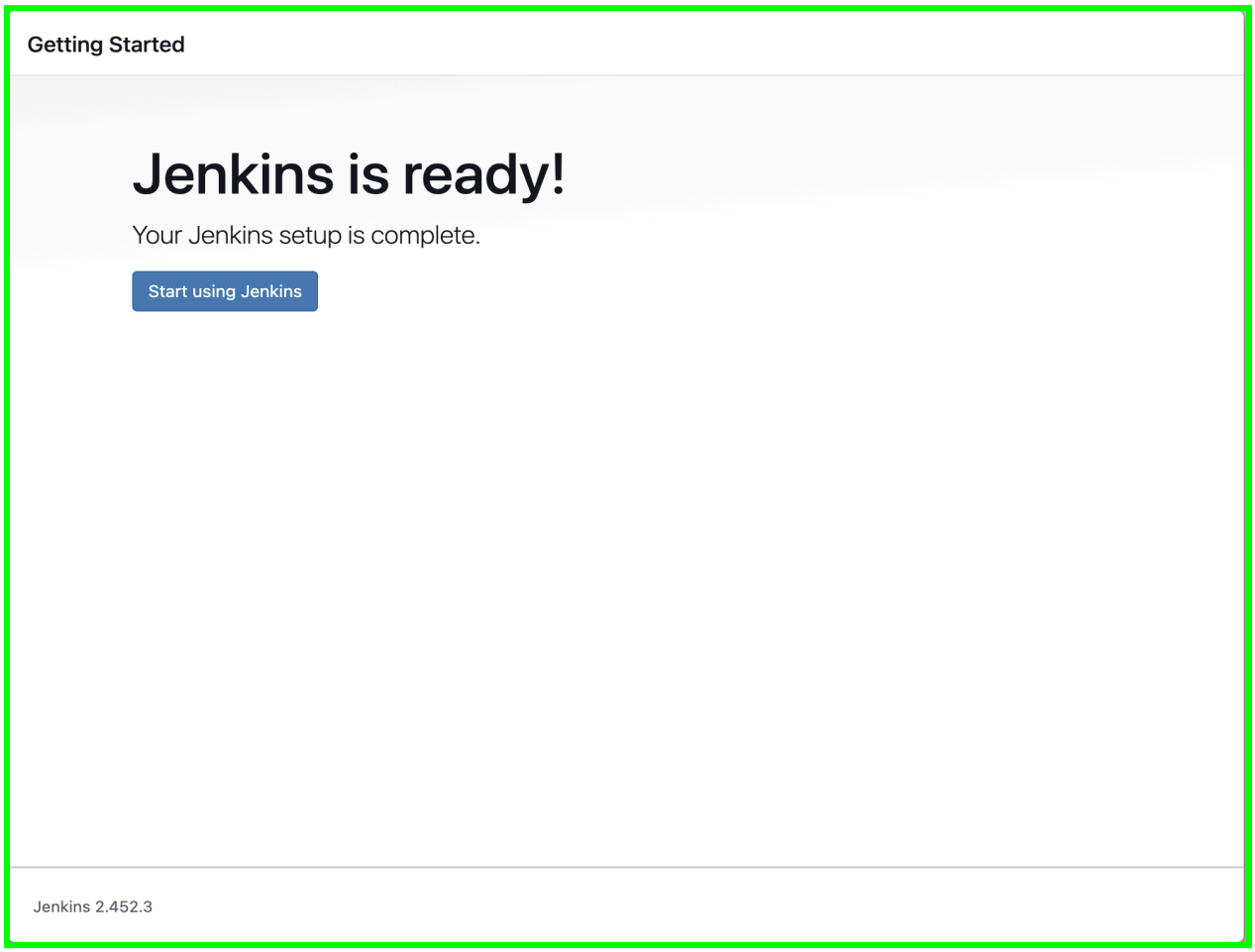


Jenkins URL , you can keep the url or can change it
http://<EC2PublicIPAddress>:8080 → **this is the Default URL.**

Hit Save and Finish



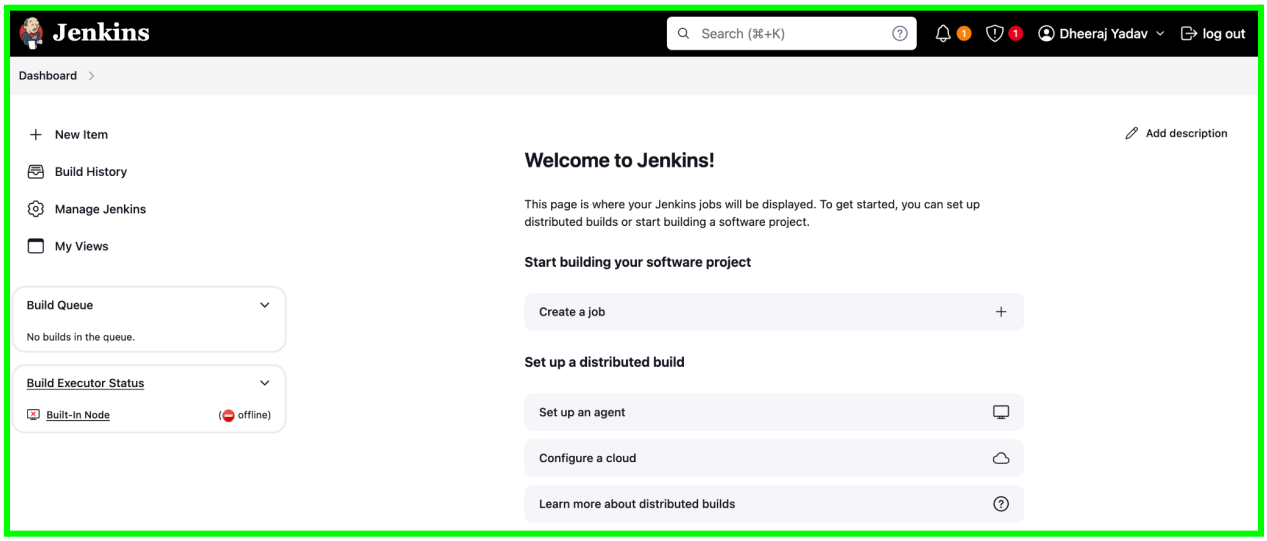
Click on Start using Jenkins:



==========

Jenkins has been installed successfully

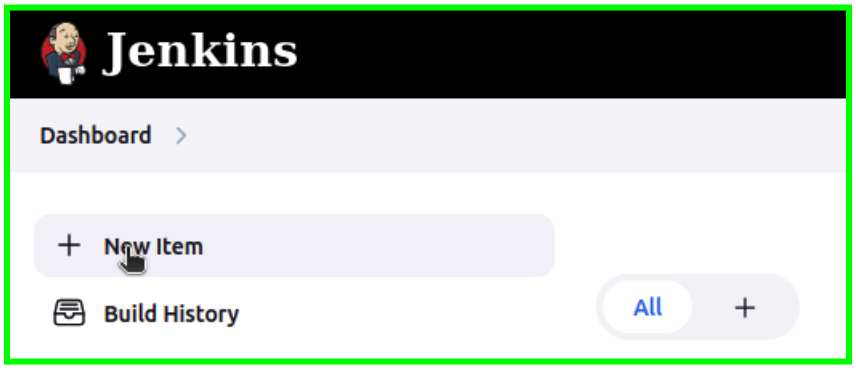
See the below DashBoard:



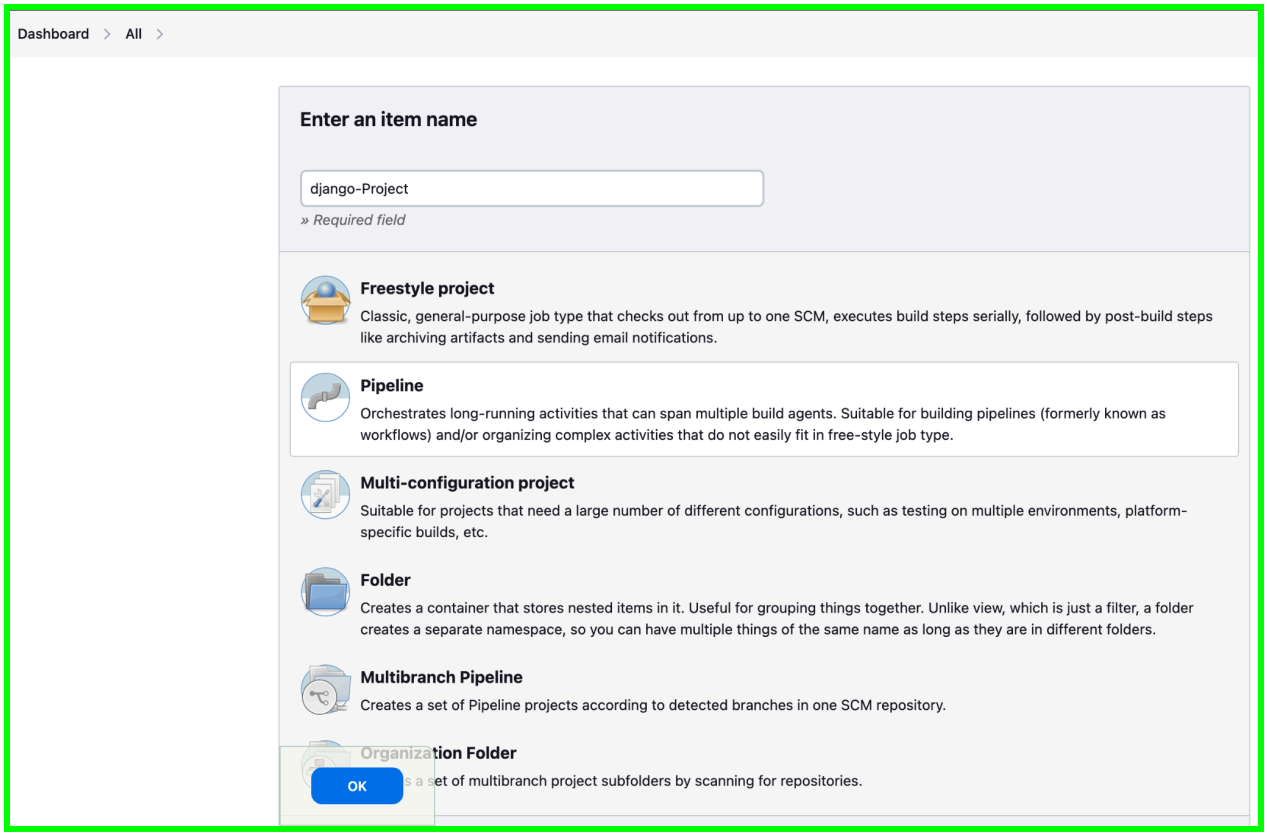
==========

Step -5 : Create a Jenkins Pipeline

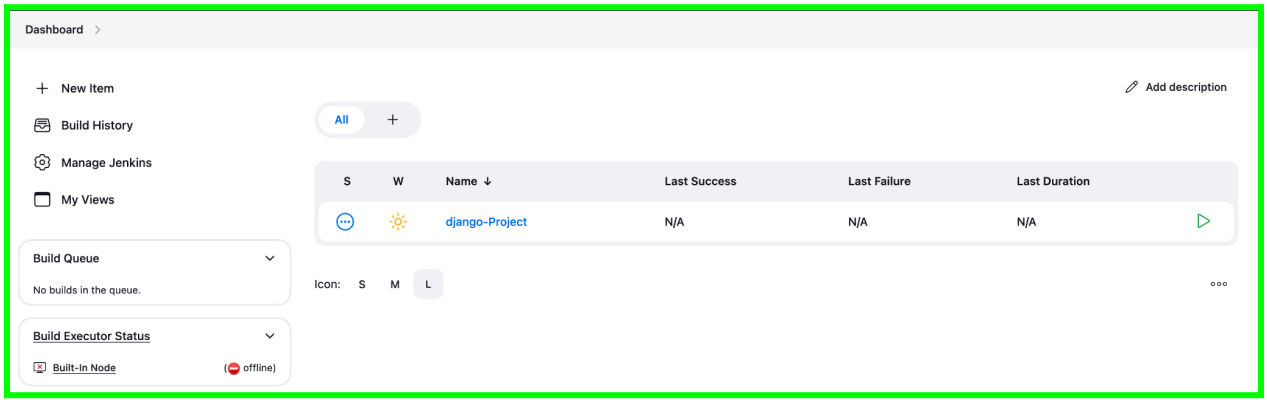
Click on “New Item”



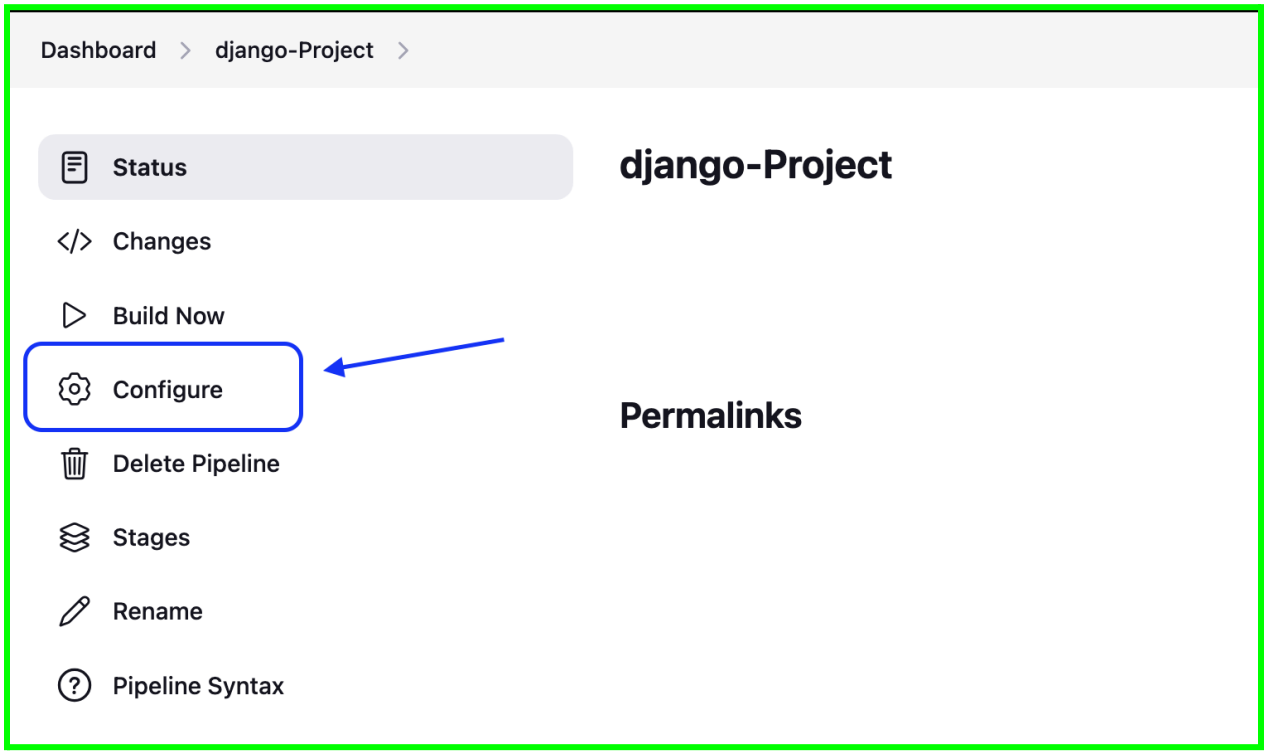
Select Pipeline & give a Project Pipeline Name, Hit OK.



Once you click okay , it will show you like this in your Dashboard.



Select the “django-Project” & Click on **Configure** icon.

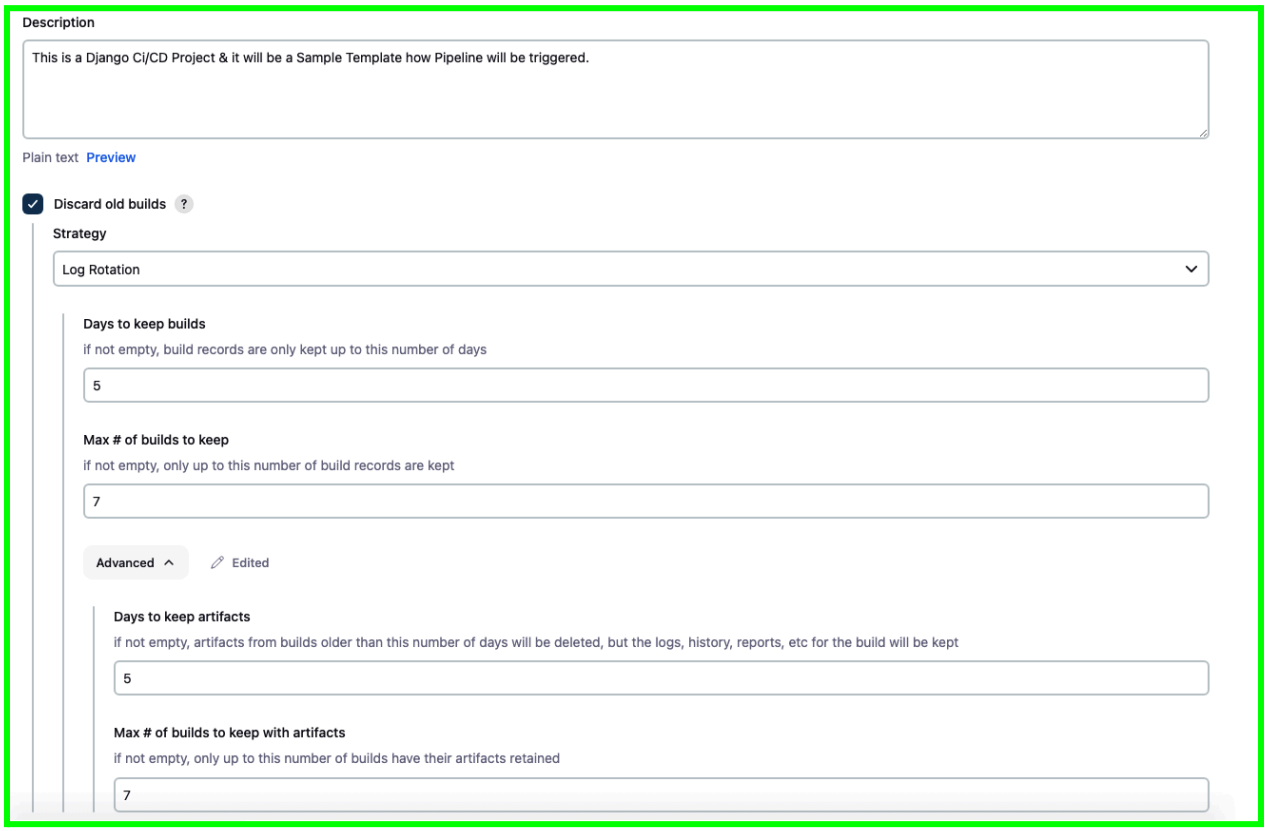


General Template Form

It will Open Up a template Form which needs to be filled. Let me explain all the Option 1 by 1.

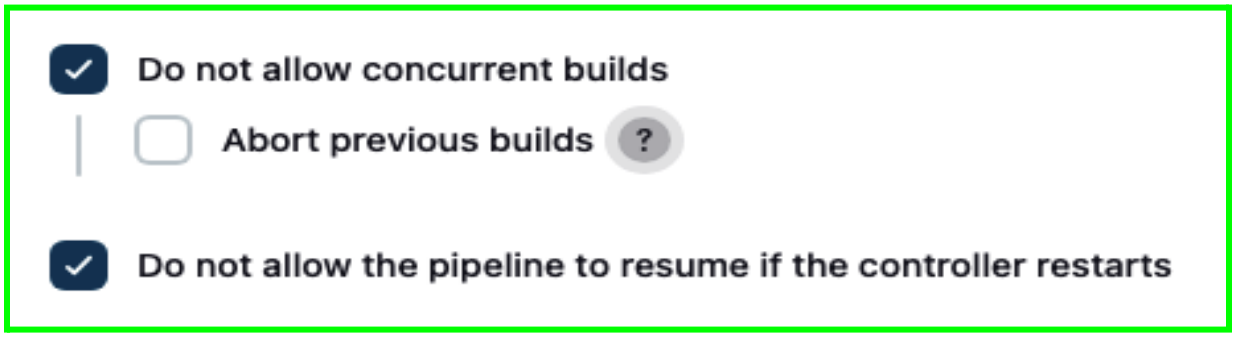
a) **Discard old builds :** You can set up the days to keep your Old Build History before Jenkins start Overriding, here in this example i have configured:

- Days to keep builds = 5
- Max # of builds to keep = 7
- Days to keep artifacts = 5
- Max # of builds to keep with artifacts = 7



b) **Do not allow concurrent builds :** Once you select this option, you can't run a parallel build, meaning it will wait to complete the previous Job.

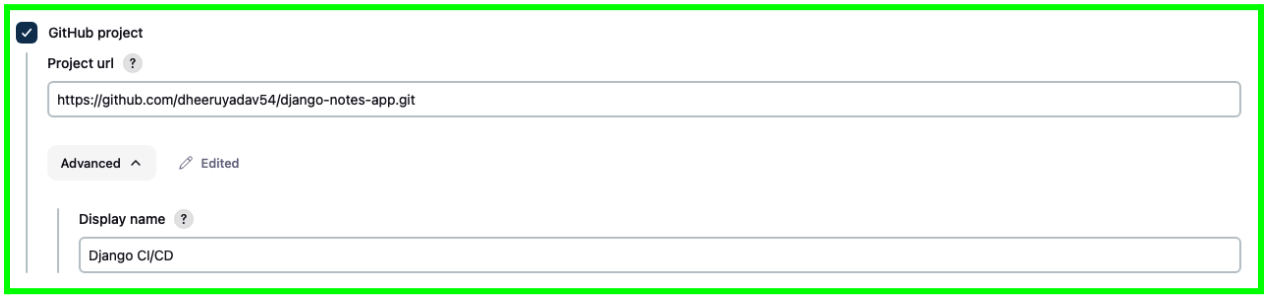
c) **Do not allow the pipeline to resume if the controller restarts :** In case, you restart the Jenkins & don't want to resume the pipeline before restart, use this checkbox,



d) **GitHub project :** Enter the github Project Repo URL here.

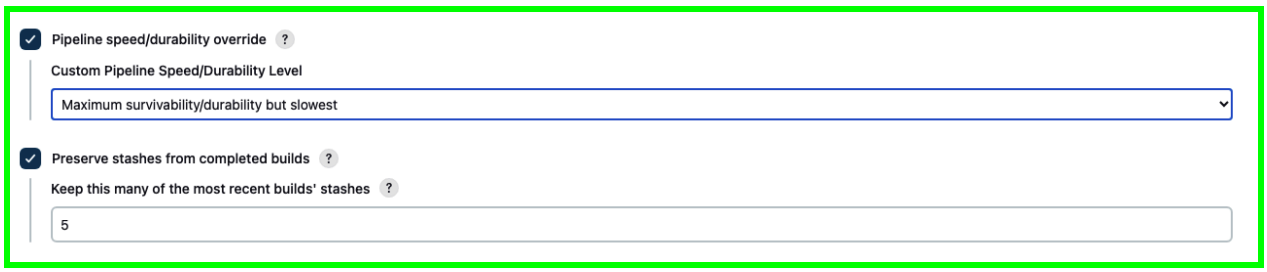
Project URL : https://github.com/dheeruyadav54/django-notes-app.git

Display name : It will be visible during Commit Status.

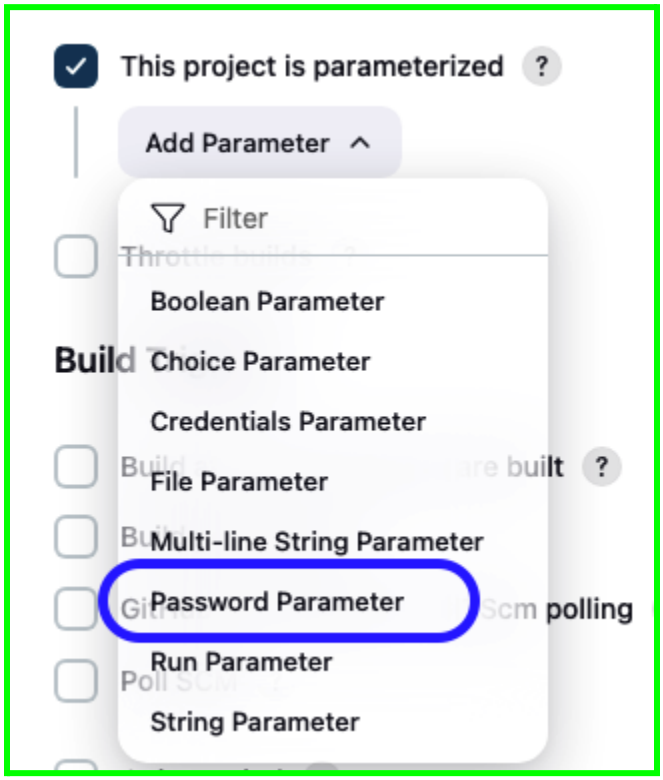


e) **Pipeline speed/durability override :** You can configure the Pipeline speed based on the need , we will be using the default option : Maximum Survivability/Durability but Slowest.

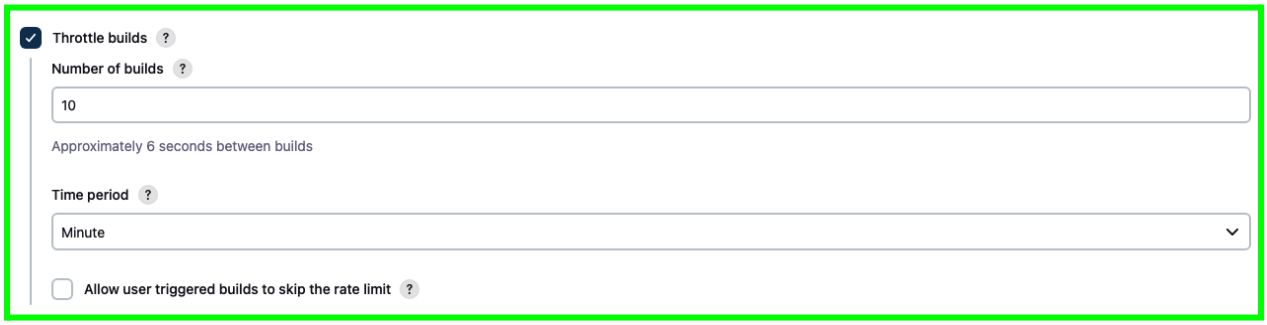
f) **Preserve stashes from completed builds :** If you check this box, meaning you can keep this Project Artifacts up-to 5 latest Pipeline, before Jenkins starts Overriding it.



g) **This project is parameterized :** We can use the below Parameter to insert into pipeline like Password parameter



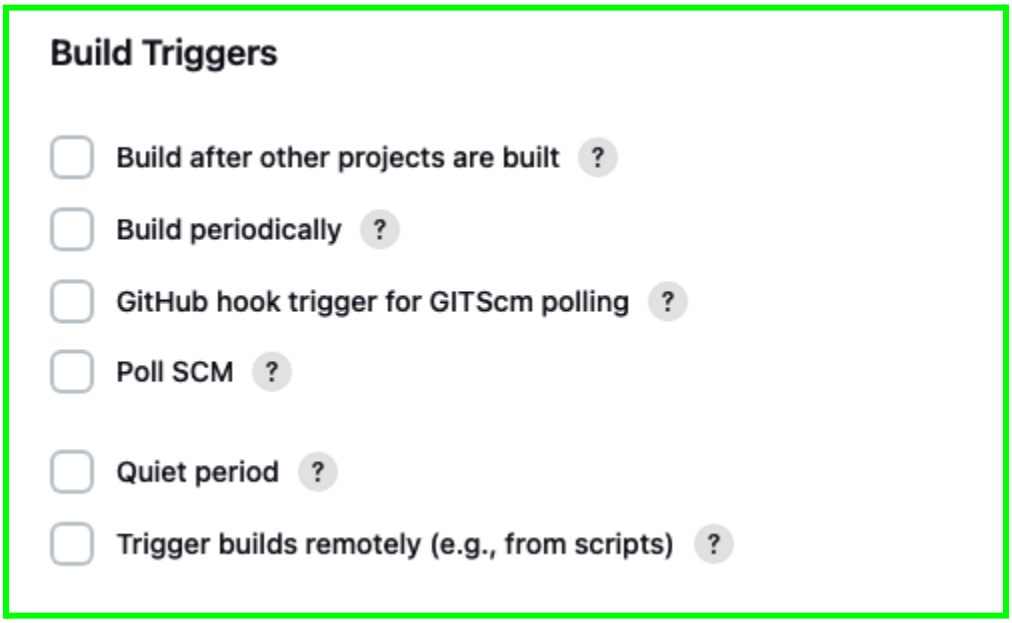
h) Throttle builds : We can define the Number of builds per Second, minute, hour, Day, Week, Month, Year etc.
⇒ **We have defined 10 Builds per minute**



==========

Build Triggers

a) Build after other projects are built :
b) Build periodically
c) GitHub hook trigger for GITSCM polling: Whenever you make changes in the github repository Jenkins receives a GitHub push hook, GitHub Plugin checks to see whether the hook came from a GitHub repository.
d) Poll SCM
e) Quiet period
f) Trigger builds remotely (Script)



Advanced Project Options:

Display name :

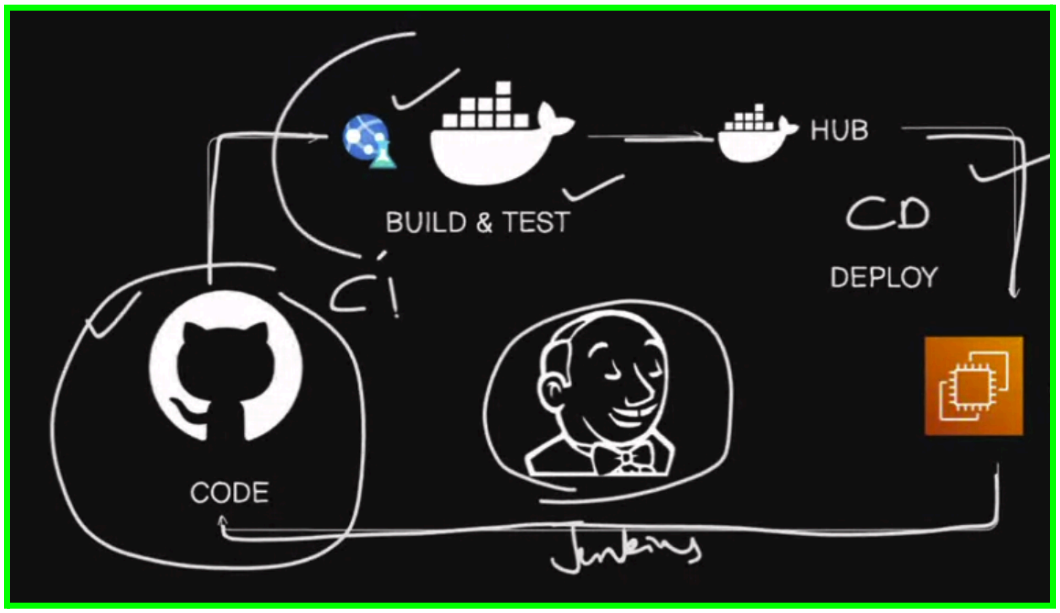
If set, the optional display name is shown for the project throughout the Jenkins web GUI. As it is for display purposes only, the display name is not required to be unique amongst projects.

If the display name is not set, the Jenkins web GUI will default to showing the project name.

==========

Pipeline

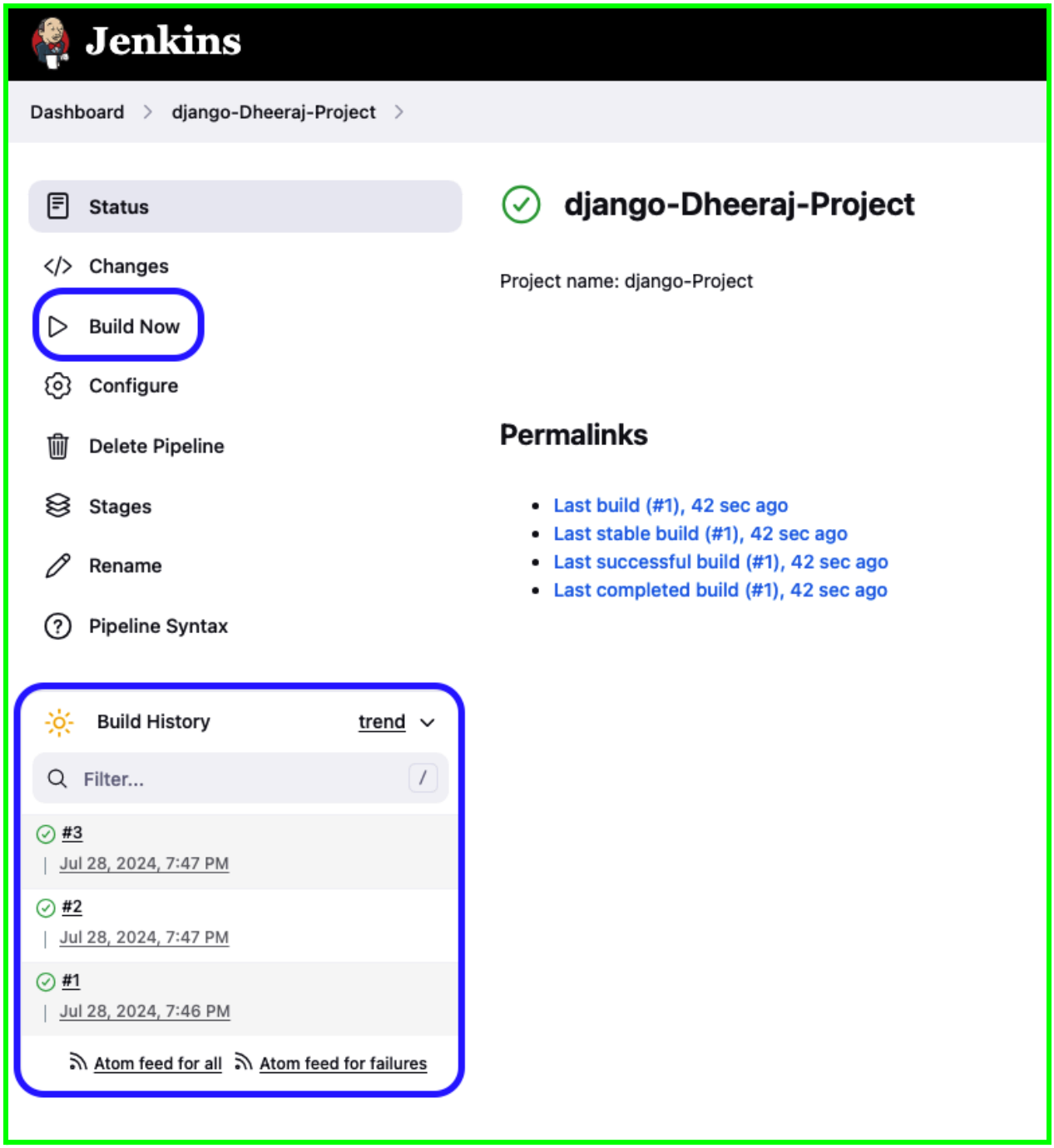
Pipeline Flow



===> Created this Simple Pipeline where we have created 4 stages, Click Save now.

Once you Save the pipeline, now the next option is to build it.

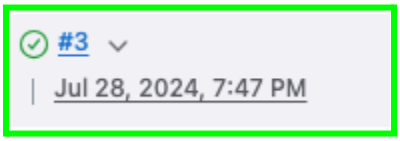
Click on **"Build Now"**, you will see the Build History below.


Jenkins
Dashboard
django-Dheeraj-Project
Status
Changes
Build Now
Configure
Delete Pipeline
Stages
Rename
Pipeline Syntax
Build History
trend
Filter...
#3
Jul 28, 2024, 7:47 PM
#2
Jul 28, 2024, 7:47 PM
#1
Jul 28, 2024, 7:46 PM
Atom feed for all
Atom feed for failures
django-Dheeraj-Project
Project name: django-Project
Permalinks
Last build (#1), 42 sec ago
Last stable build (#1), 42 sec ago
Last successful build (#1), 42 sec ago
Last completed build (#1), 42 sec ago

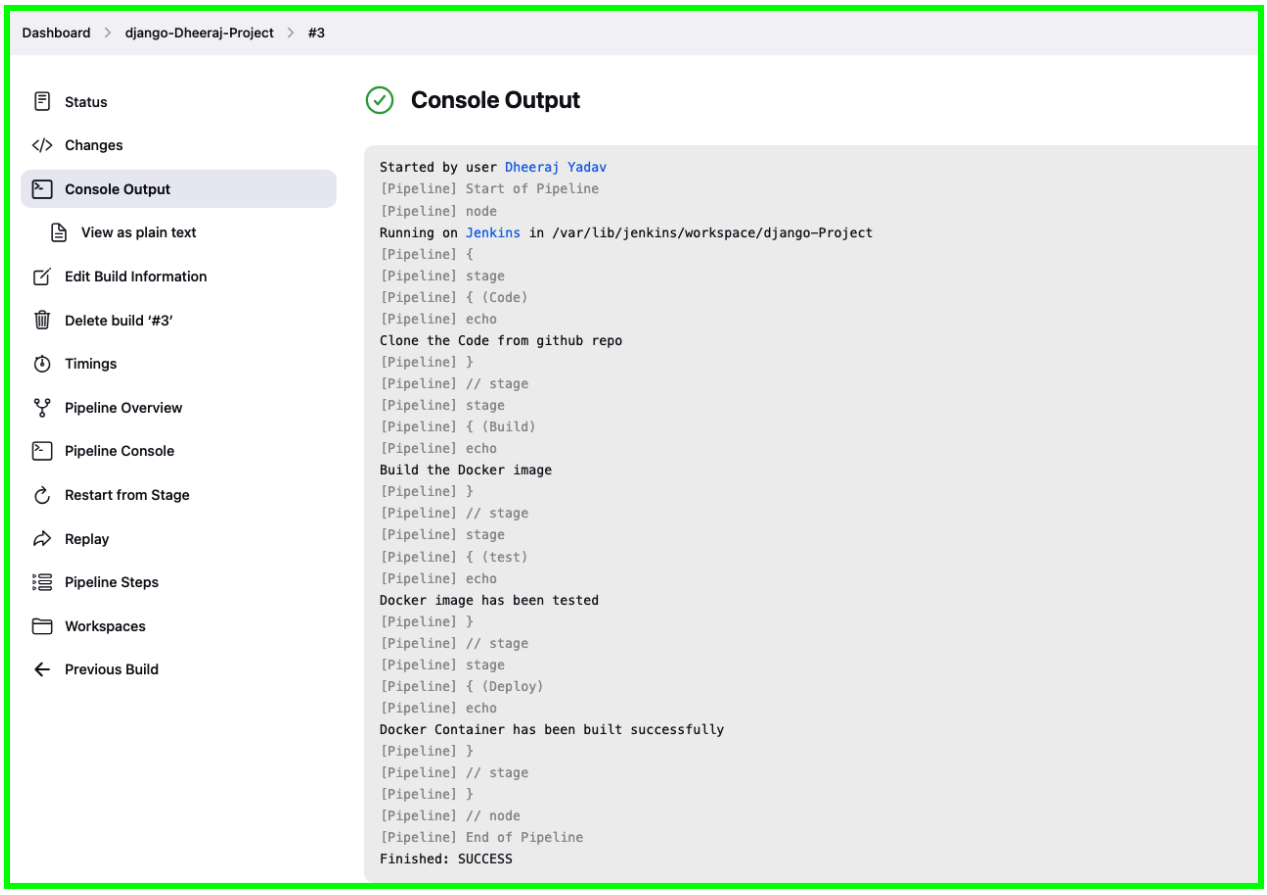
Click on successful pipeline, here in this example click on #03

==========

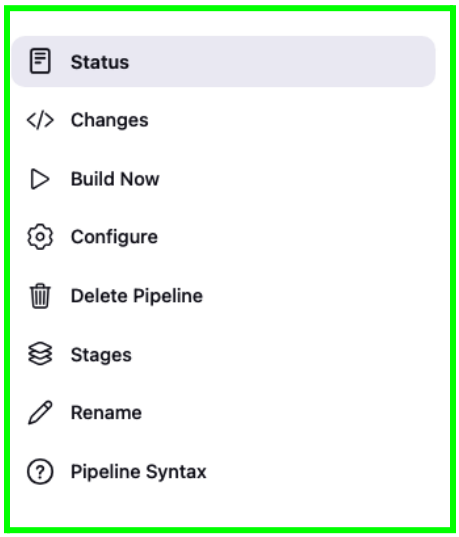
Success



Click on Console Output to see more details about pipeline steps



Click on Stages tab to see each stage status.



All the Pipelines were successful.

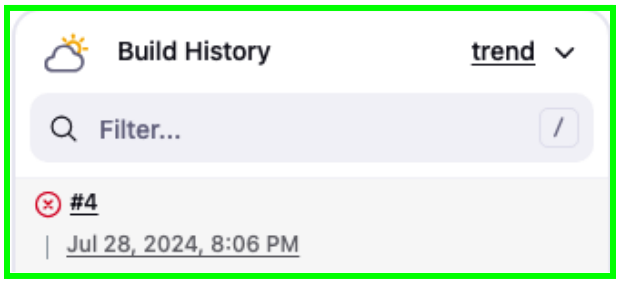
Let's make a mistake to see pipeline fail status.

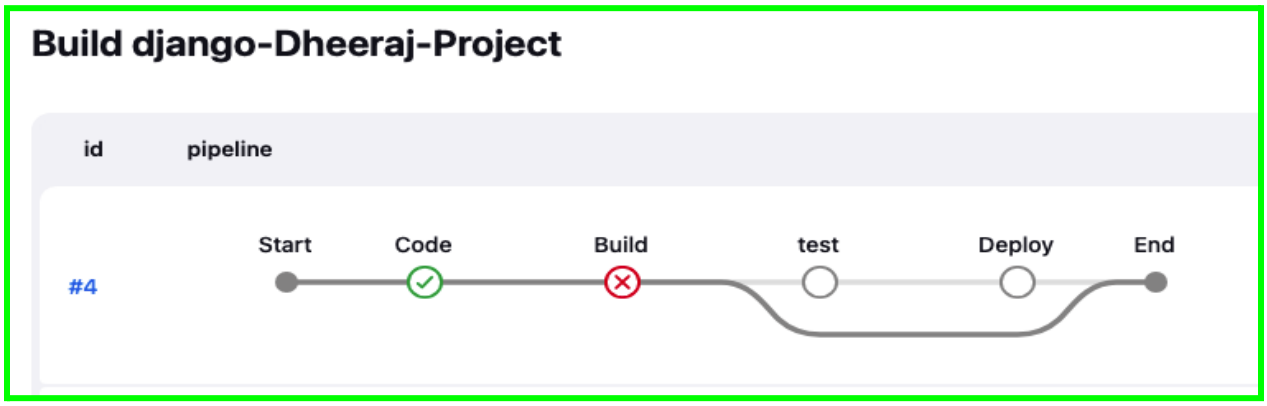
==========

Failure

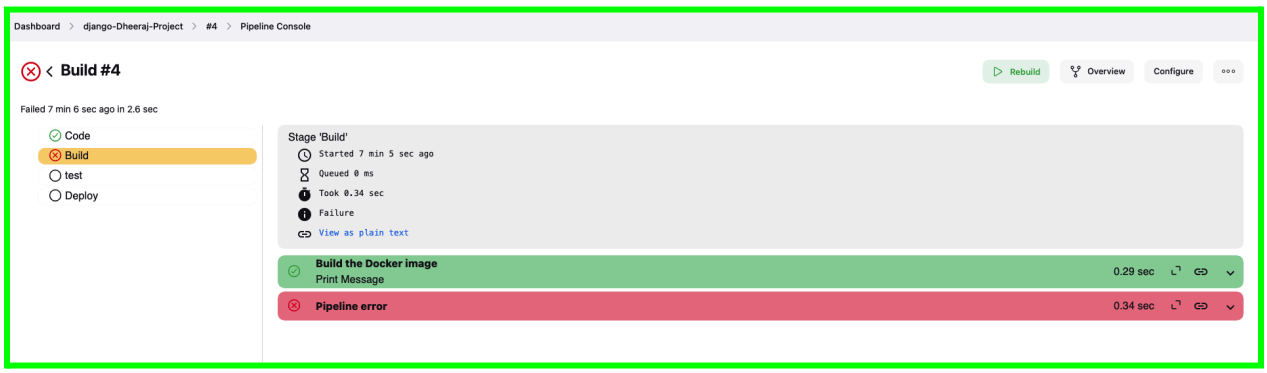
Make a small mistake, now the pipeline will fail, Just to show you how fail pipeline looks.

Build number #4 failed, Let's check the stages Output.





Click on Build Failed step to see more details about Failure.



==========

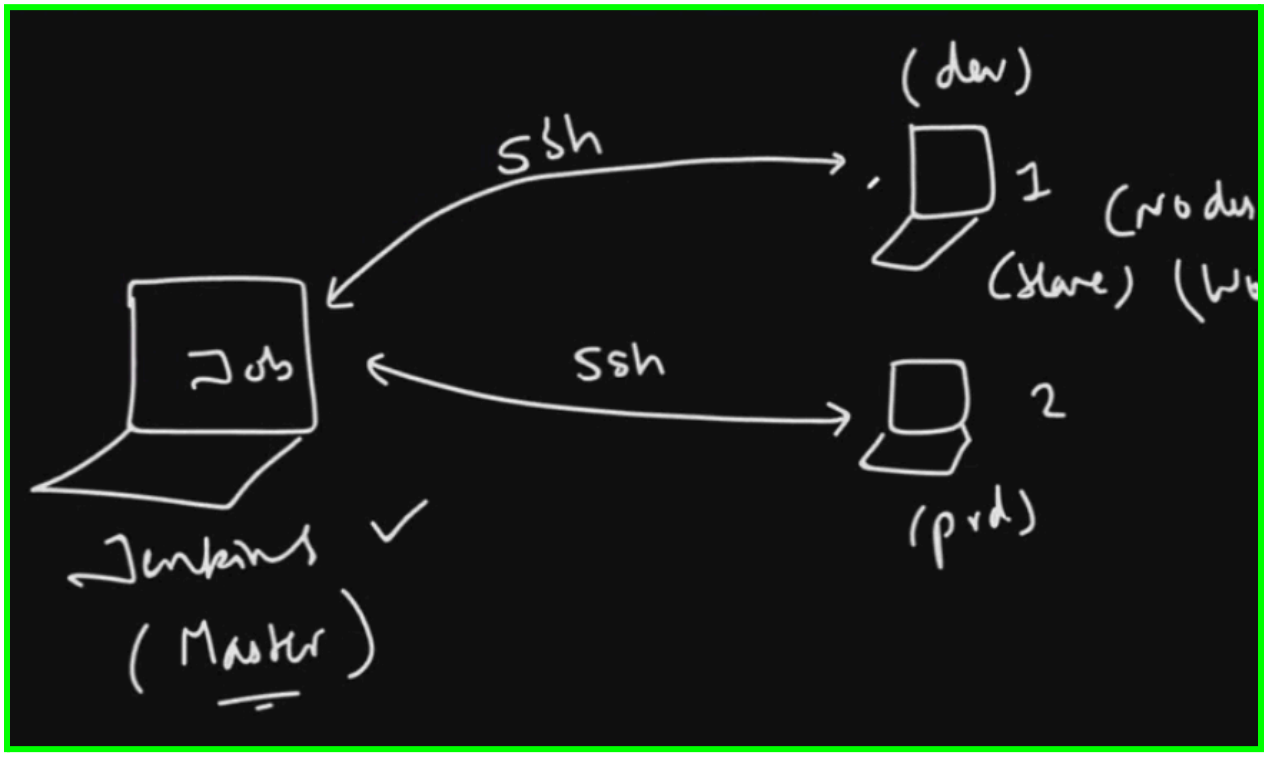
Step -6 : Setting Up the Agent

===> We will Create one more EC2 instance to be running as AGENT.

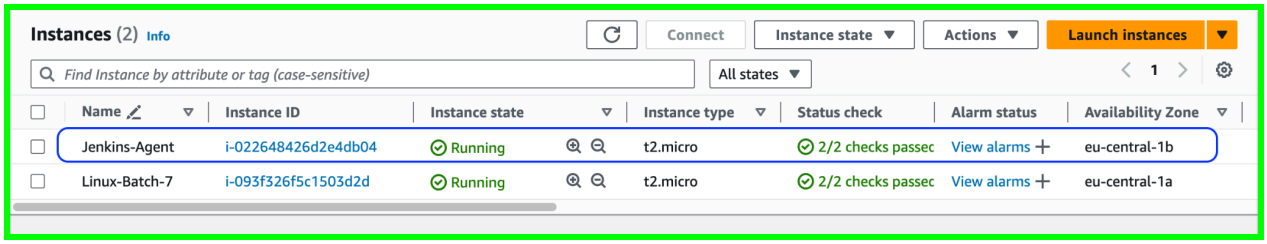
Note : Agent doesn't need Jenkins to be running, if it does then it is running as **MASTER**.

Prerequisites : Need to install JAVA only on Agent Machine.

This is how Master → Agent will be communicating “through Private , Public key”



===> Created a New EC2 Instance with Name **"Jenkins-Agent"**



==========

Generate SSH Key

Let's Generate SSH Key so jenkin's Agent can connect to Jenkins Master Server with Public key.

$ ssh-keygen : Run this command to generate the SSH Private & Public key on Jenkins Master Server.

Note : if you keep the default name id_ed25519 for your Private key then Public Key Name would be id_ed25519.pub & both will be saved to default location /home/ubuntu/.ssh

In this example, I have changed the name hence the key saved into my /home/ubuntu not into .ssh.

```
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_ed25519): id_jenkins
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in id_jenkins
Your public key has been saved in id_jenkins.pub
The key fingerprint is:
SHA256:B97Q5EQ5xDQuUb9YrjY2wlMSSH5XiTJ9r4+8UiSugZ8 ubuntu@ip-172-31-24-254
The key's randomart image is:
+--[ED25519 256]--+
|      . .BOo..   |
|     o .oB*+o    |
|      o *o=oo.   |
|       o B.+...  |
|       .S.=oo.   |
|      ...+..o    |
|       .++*o o   |
|        E=.oo .  |
|           ...   |
+----[SHA256]-----+
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$
```

```
ubuntu@ip-172-31-24-254:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-24-254:~$ ls -ltr id_jenkins*
-rw-r--r-- 1 ubuntu ubuntu 105 Jul 28 20:34 id_jenkins.pub
-rw------- 1 ubuntu ubuntu 419 Jul 28 20:34 id_jenkins
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$
```

===> Copy this Public key & paste into Jenkins-Agent under **authorized_keys file.**

```
ubuntu@ip-172-31-24-254:~$ cat id_jenkins.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIDGn1vsy3aNTfCBGvvIu+PWpZgnBui4JWyH6bTSCueAy ubuntu@ip-172-31-24-254
ubuntu@ip-172-31-24-254:~$
```

```
ubuntu@ip-172-31-35-243:~/.ssh$ pwd
/home/ubuntu/.ssh
ubuntu@ip-172-31-35-243:~/.ssh$
ubuntu@ip-172-31-35-243:~/.ssh$ ls
authorized_keys
ubuntu@ip-172-31-35-243:~/.ssh$
```

Paste the Pub Key , See below:

```
ubuntu@ip-172-31-35-243:~/.ssh$ cat authorized_keys
ssh-rsa '
myd7d8PP
M8xZ1DZH
b3gX2iEOK0B Batch-7-Key
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIDGn1vsy3aNTfCBGvvIu+PWpZgnBui4JWyH6bTSCueAy ubuntu@ip-172-31-24-254
ubuntu@ip-172-31-35-243:~/.ssh$
```

From Master i did the SSH to Jenken's Agent using the id_jenkins Private key.
Master Jenkins is able to connect Jenkins Agent through SSH Key.

```
Jenkins-EC2-Master ⌘1 | Jenkins-EC2-Agent ⌘2
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$ ssh -i id_jenkins ubuntu@ec2-3-127-68-109.eu-central-1.compute.amazonaws.com
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

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

 System information as of Sat Aug  3 15:22:12 UTC 2024

  System load:  0.0               Processes:             103
  Usage of /:   37.3% of 6.71GB   Users logged in:       0
  Memory usage: 19%               IPv4 address for enX0: 172.31.35.243
  Swap usage:   0%

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

   https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

27 updates can be applied immediately.
6 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

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


Last login: Sat Aug  3 15:22:13 2024 from 172.31.24.254
ubuntu@ip-172-31-35-243:~$
ubuntu@ip-172-31-35-243:~$
```

==========

Install JAVA

Install the JAVA into Jenkin's Agent Machine.

$ sudo apt-get update

$ sudo apt install fontconfig openjdk-17-jre : Install the JAVA 17 version per Jenkins Document.

```
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$ sudo apt-get update
Hit:1 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Hit:3 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:5 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [297 kB]
Fetched 423 kB in 1s (645 kB/s)
Reading package lists... Done
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$ sudo apt install fontconfig openjdk-17-jre
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
```

$ java –version : To check the JAVA version.

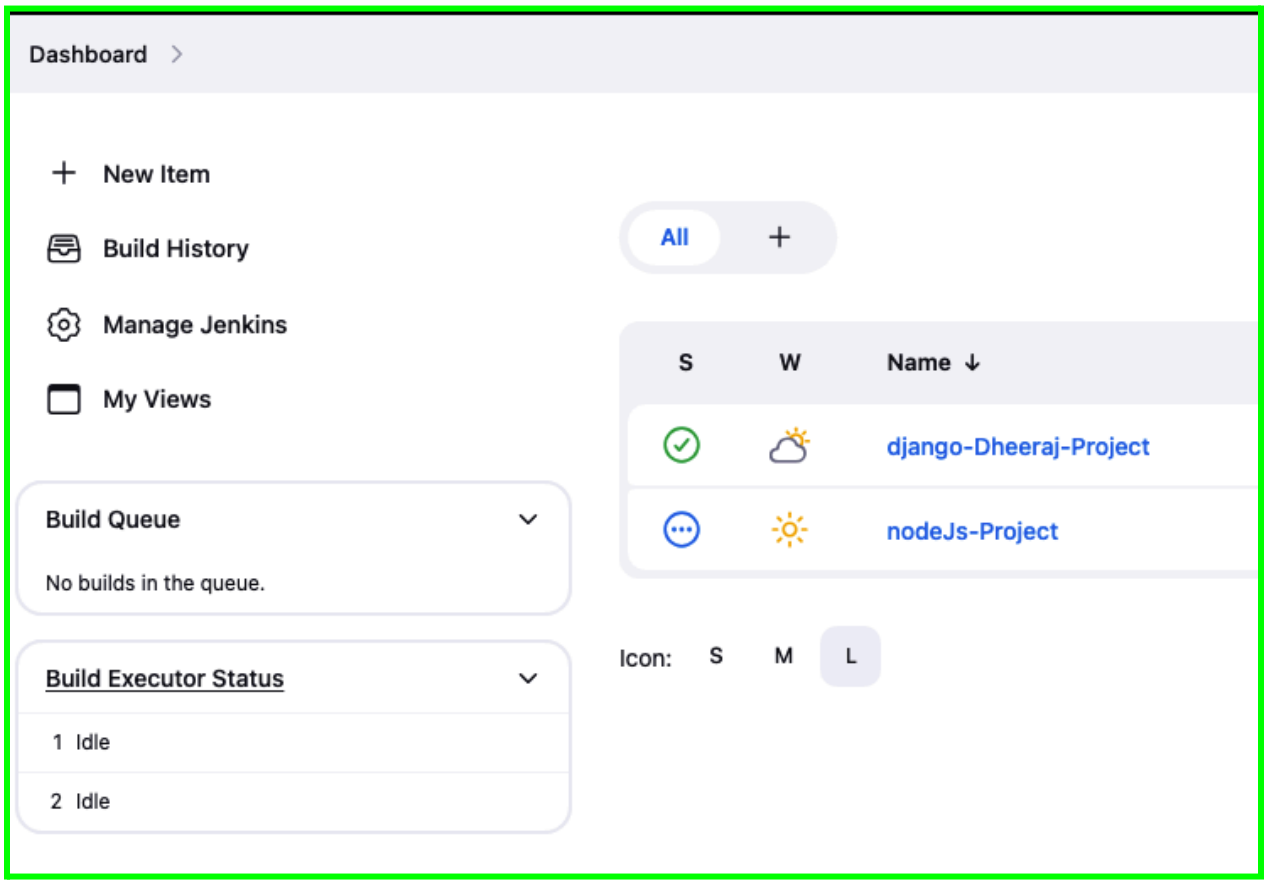
```
ubuntu@ip-172-31-24-254:~$
ubuntu@ip-172-31-24-254:~$ java --version
openjdk 17.0.11 2024-04-16
OpenJDK Runtime Environment (build 17.0.11+9-Ubuntu-1)
OpenJDK 64-Bit Server VM (build 17.0.11+9-Ubuntu-1, mixed mode, sharing)
ubuntu@ip-172-31-24-254:~$
```

Note : This SSH Connection is a EC2-EC2 instance connection , we need to add the Private Key into Jenkin's Master GUI Page as well.

==========

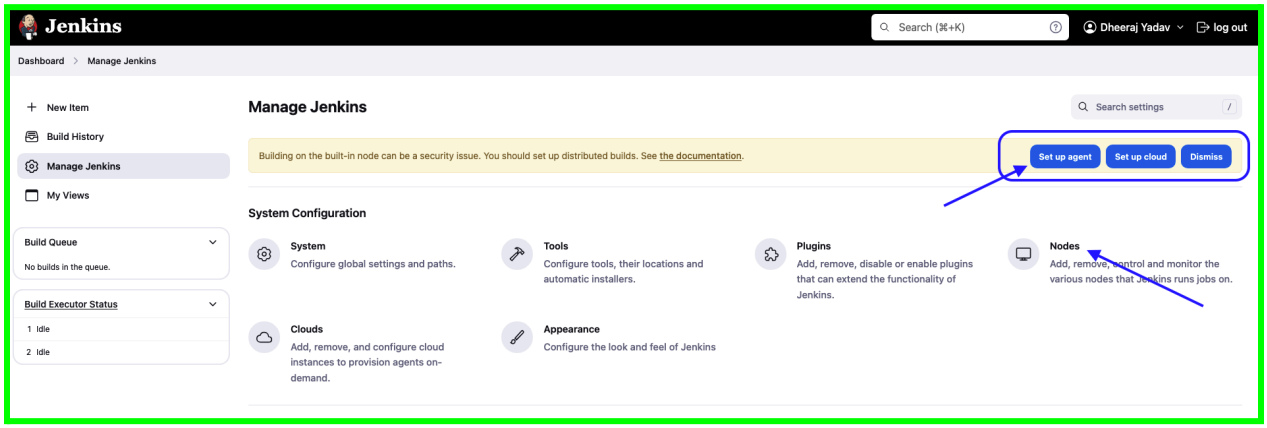
Add the Agent

Click on **Manage Jenkins**:

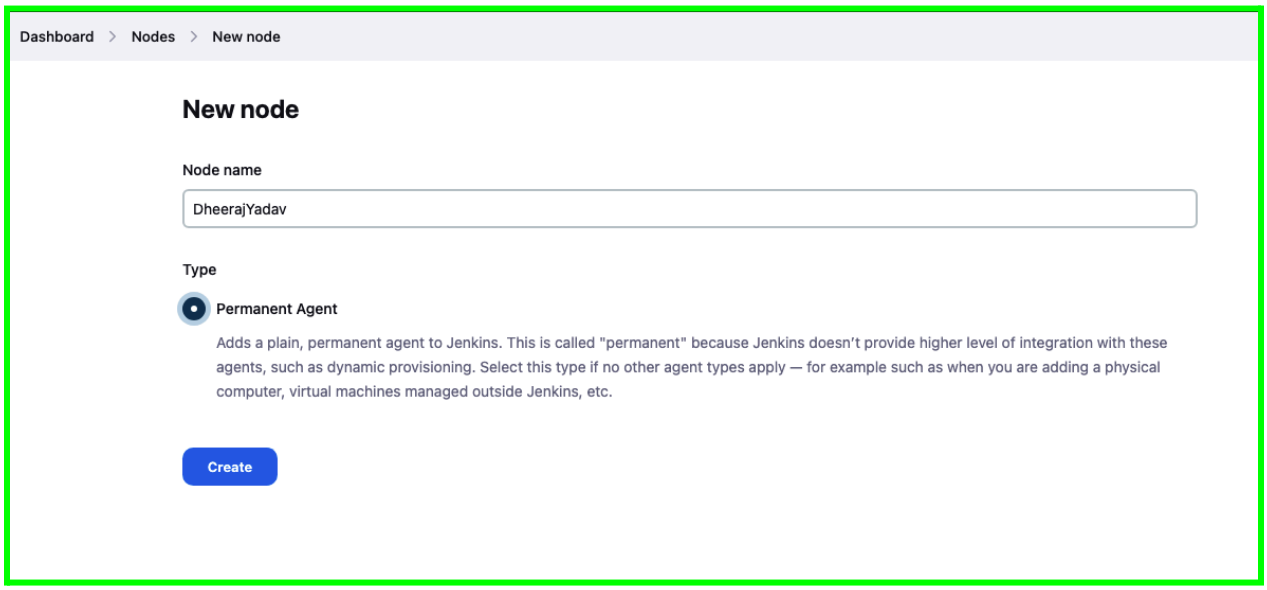


Manage Jenkins—> Set Up Agent

In case if you don't see this "Set Up agent" then click on —> Nodes—> New Nodes

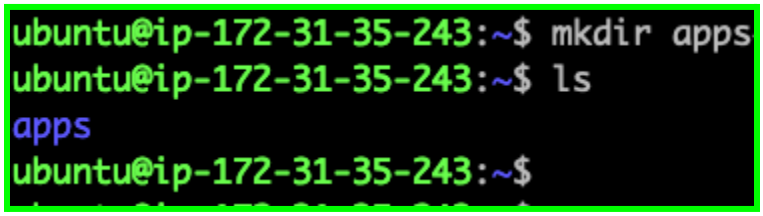


New Node —> Name —> Create



====> **Number of executors :** 1 "Only 1 Pipeline can run at a time"

Remote Root Directory : Create a Directory on Agent's machine, using **"mkdir apps"** command

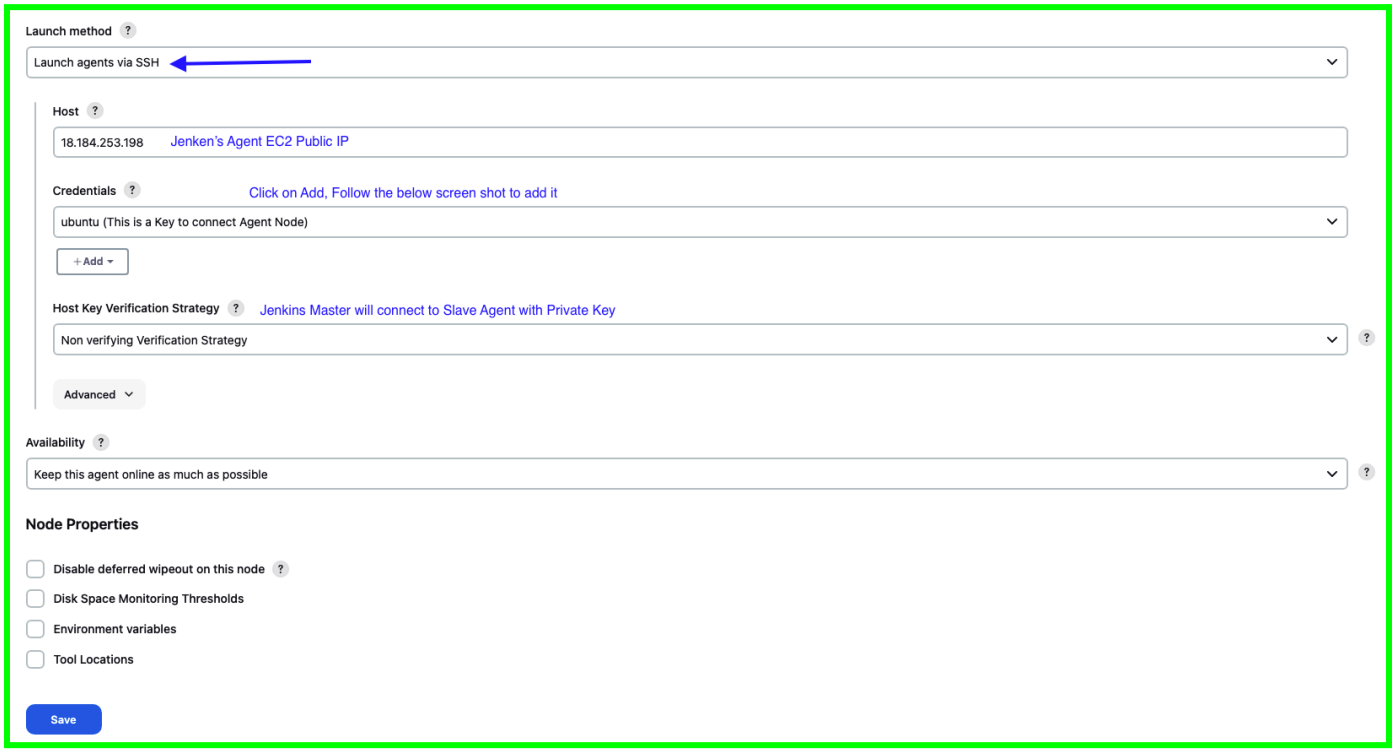


Labels : Agent name “dev” Script will use this name to find the agent to build the pipeline. We can give the same Name to multiple Agents to increase the efficiency. More Agents means stability & speed to execute the pipeline.

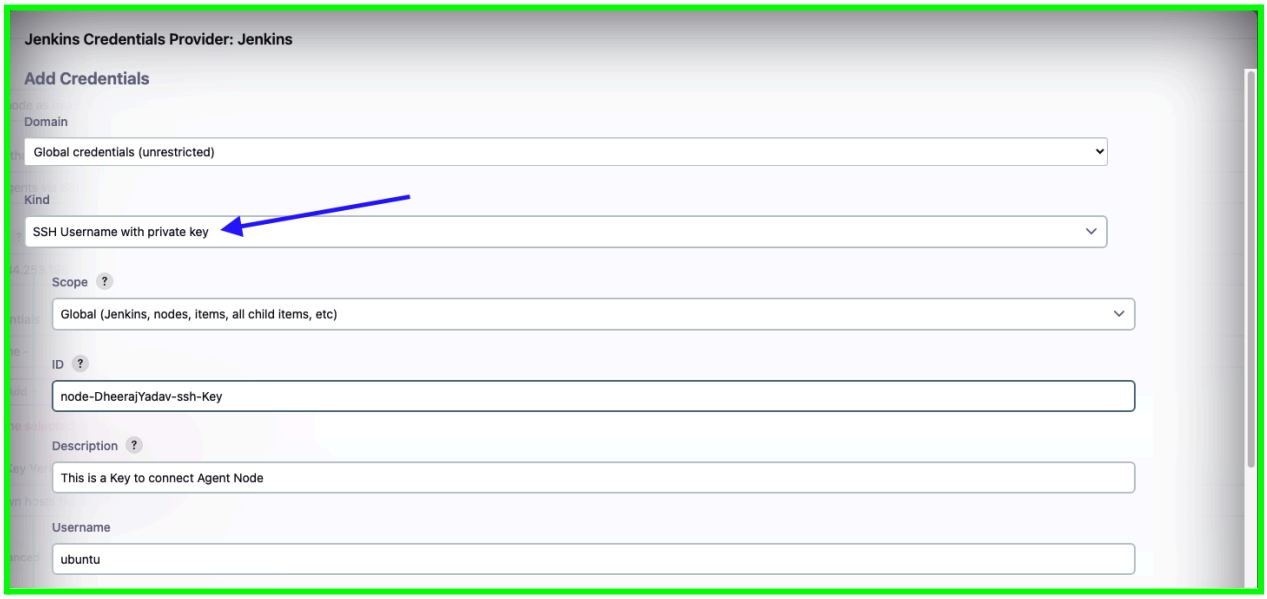
=====>**Launch Method :** Please use **“agents with SSH”**

Host : IP address of **Jenkin’s Agent**

Credentials: User→ ubuntu , use **Jenkins Master Private key**



===> **SSH Username with Private key :** Select this option



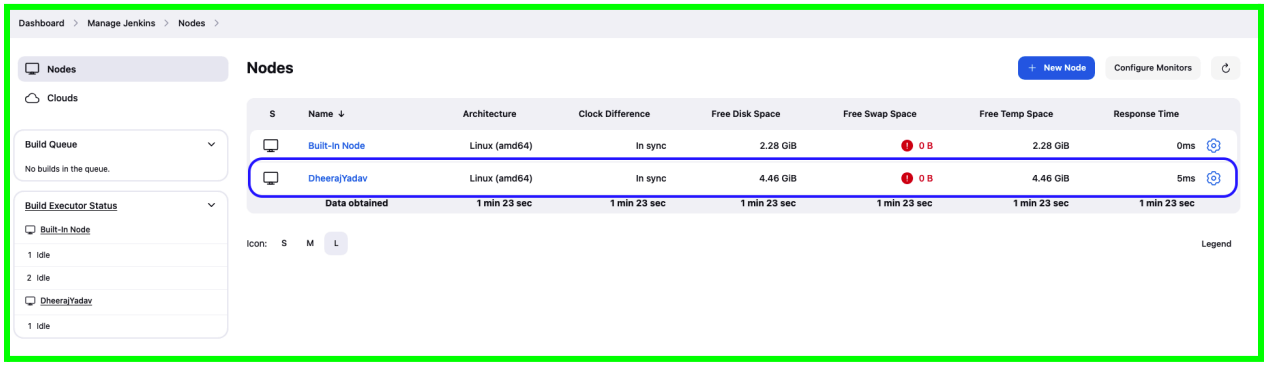
====> Enter the Master Jenkin Private key here from

cat /home/ubuntu/.ssh/id_ed25519

==========

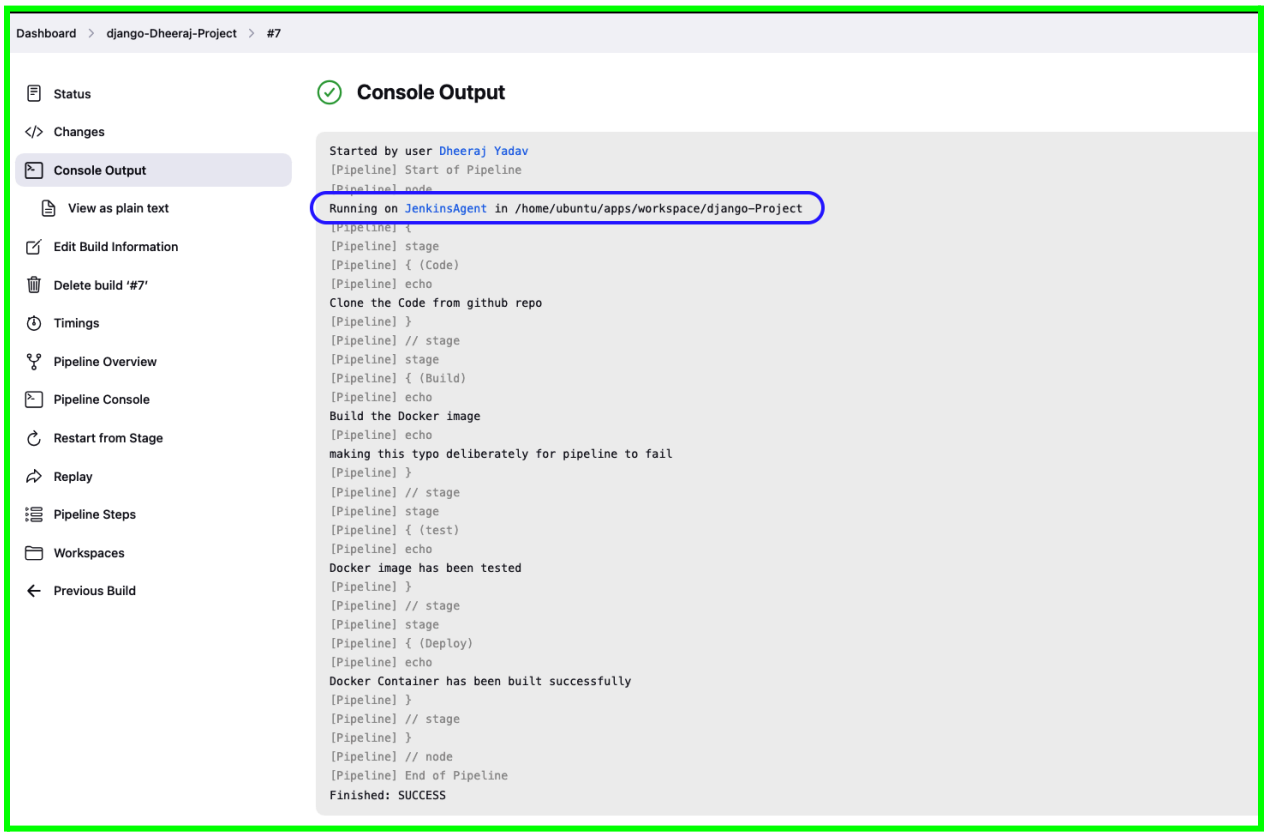
Agent Status

Agent-Node is in-sync means it's UP & running.



===> Add a Label “dev” & test the pipeline.

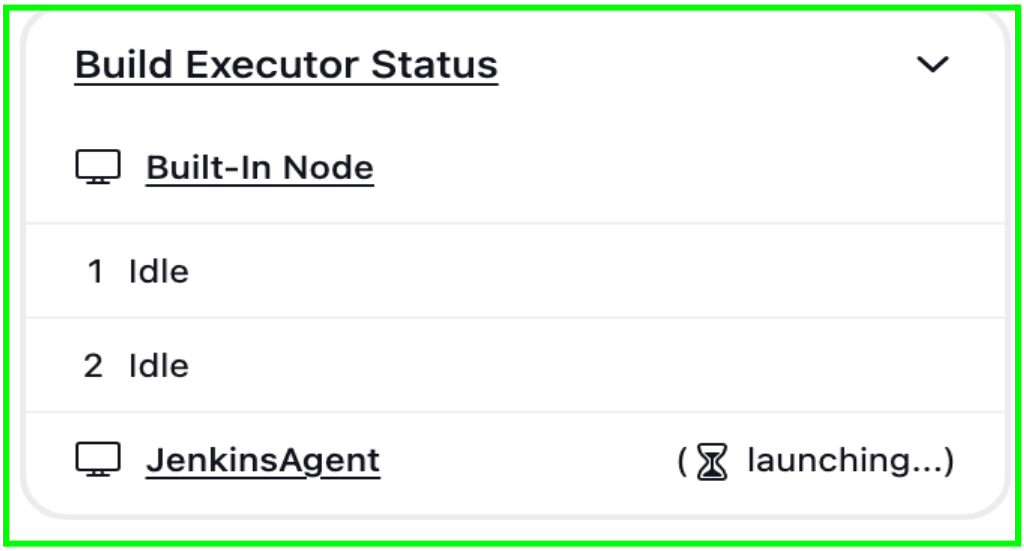
===> Jenkin’s Agent is running & executing the pipeline with Agent Node



==========

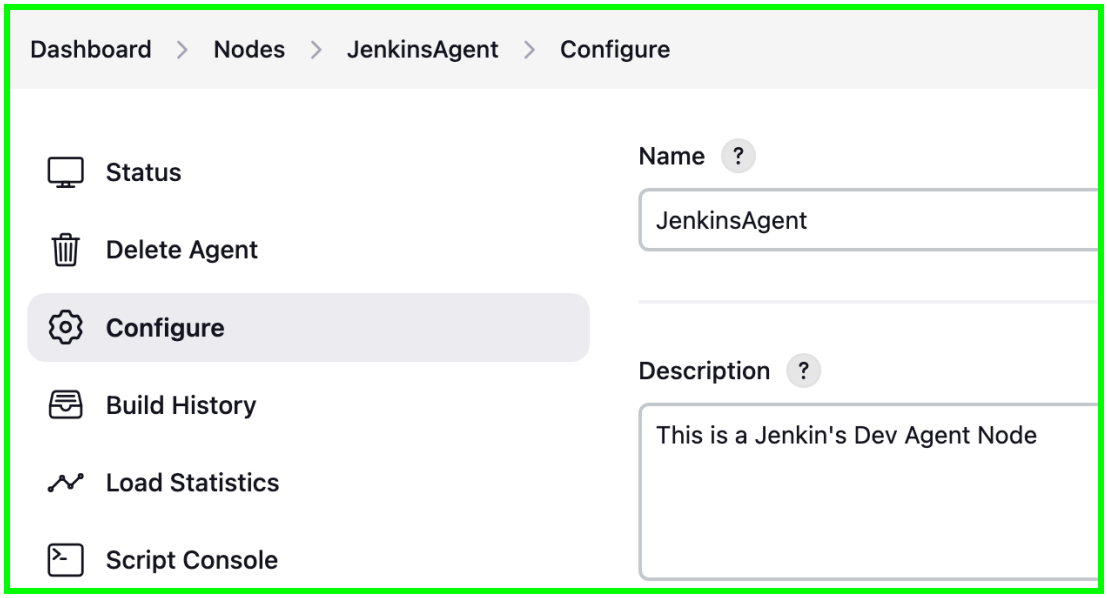
Agent Offline

In case, you stop your Agent instance, Public IP will be changed & Either Jenkin's Agent Status will be showing **"Offline" Or "Launching"**

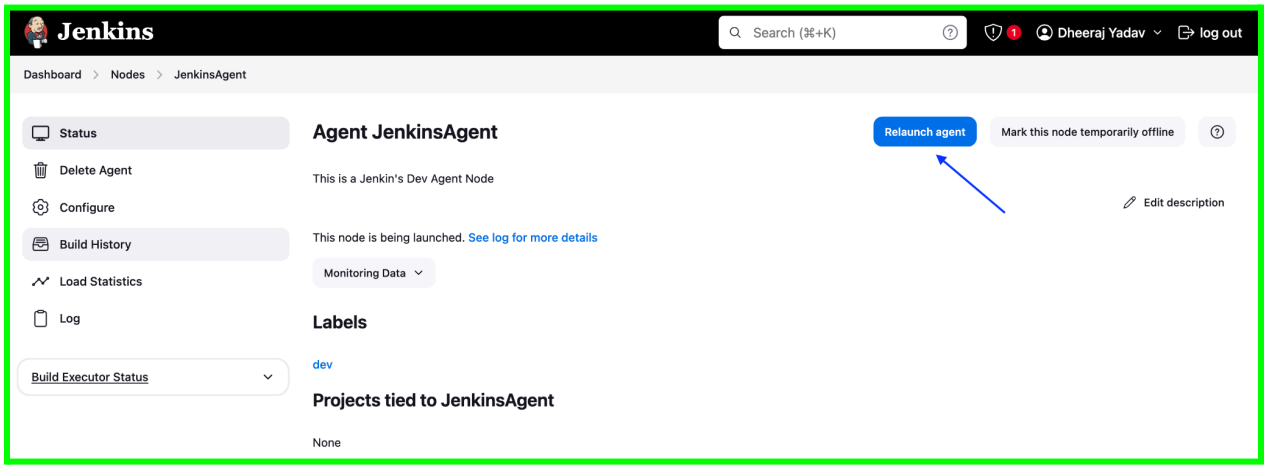


Click on JenkinsAgent PC icon showing above —> It will redirect you on the status page —> Select Configure & Update the Host IP with New IP address.

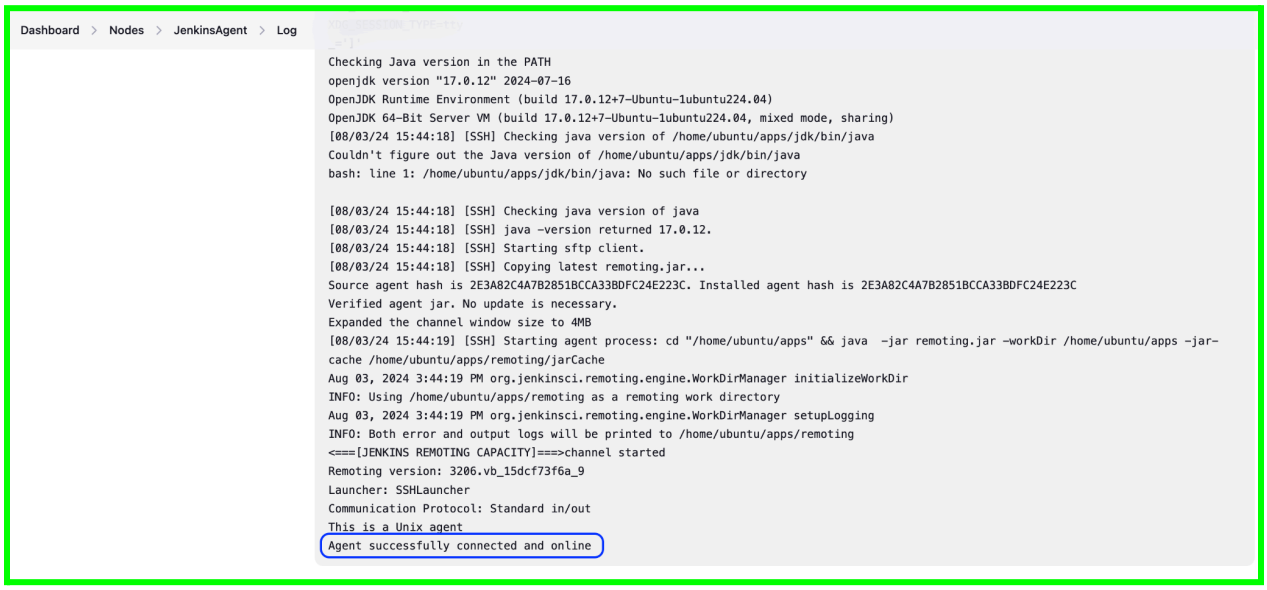
Host IP : Change with New Public IP Address.



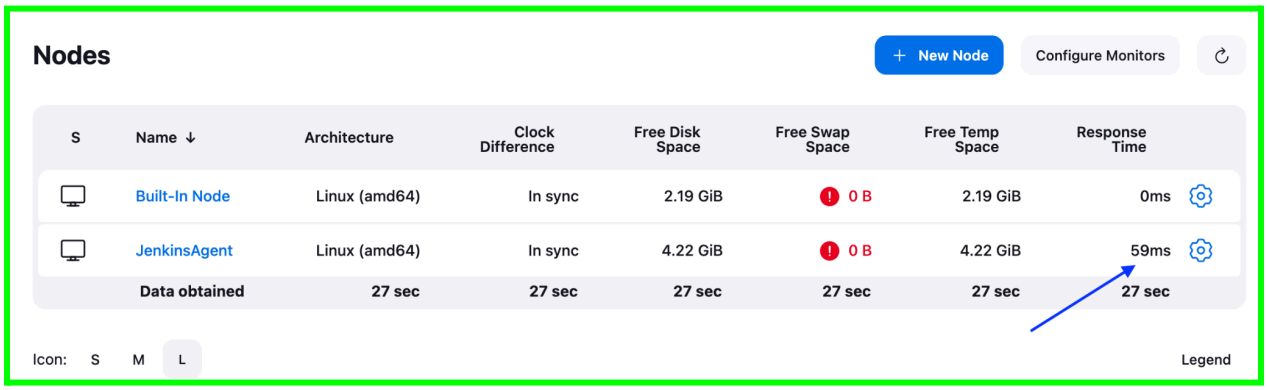
Once you update the IP address, Click on the **"Relaunch agent"** icon, it will try to re-establish the connection.



Agent Connection has been Connected and online.



Agent is Online now.



Jenkins Agent is showing **Idle** now.

==========

Step -7 : Build More Pipelines

===> Created the below pipeline to build the notes-app docker image.

Pipeline Structure

Script ?

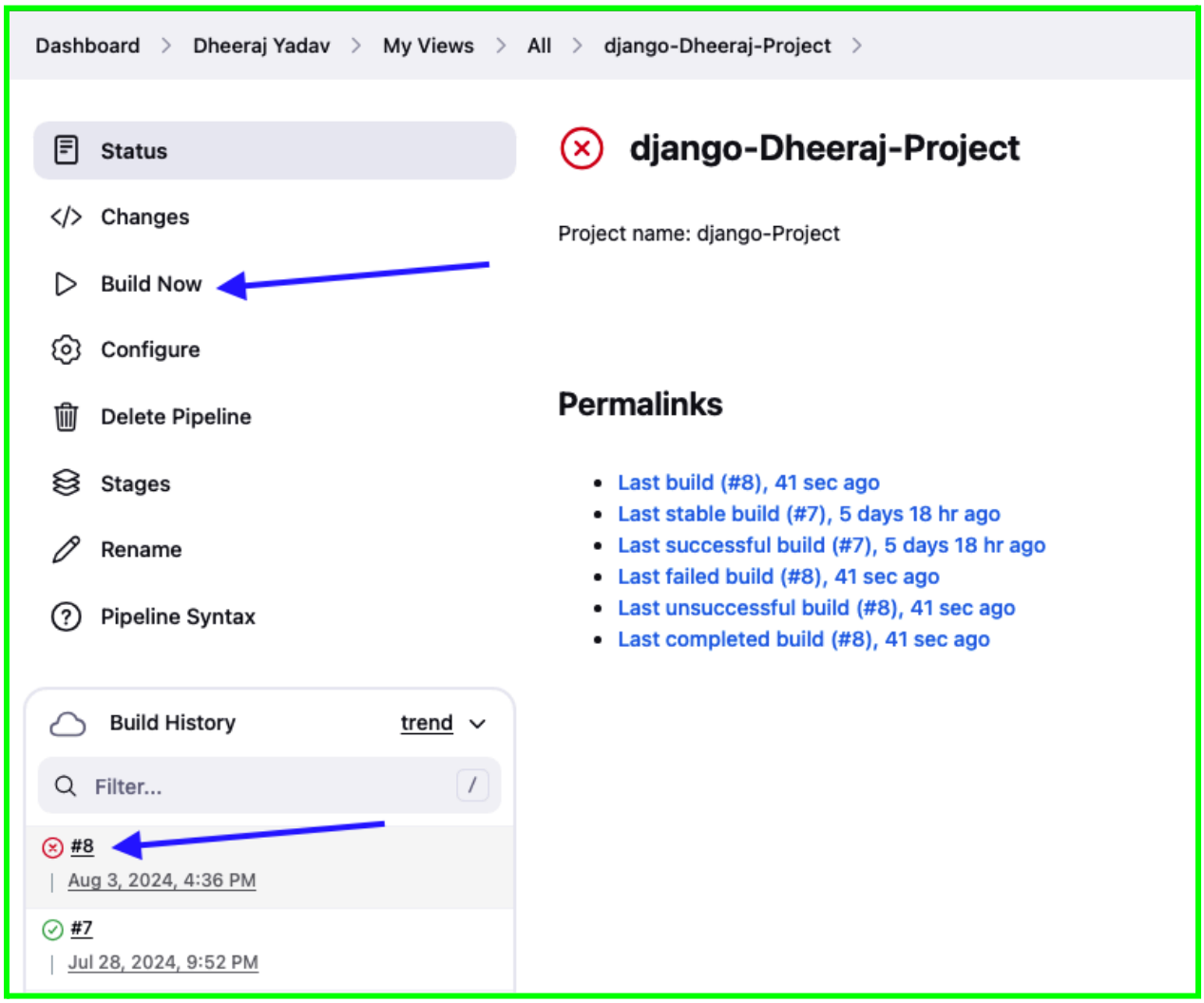
```
pipeline{

    agent{
        node{
            label "dev"
        }
    }

    stages{
        stage("Code"){
            steps{
                git url: "https://github.com/dheeruyadav54/django-notes-app.git", branch: "main"
                echo "Clone the Code from github repo"
            }
        }
        stage("Build"){
            steps{
                echo "Build the Docker image"
                sh "docker build -t notes-app:latest ."
                echo "making this typo deliberately for pipeline to fail"
            }
        }
        stage("test"){
            steps{
                echo "We don't have any test steps avaialble, it will be skip"
            }
        }
        stage("Deploy"){
            steps{
                sh "docker run -d -p 8000:8000 --name notes-app notes-app:latest"
                echo "Docker Container has been built successfully"
            }
        }
    }
}
```

==========

Build the pipeline

Click —< Build Now

I see the New Pipeline has been triggered but failed.

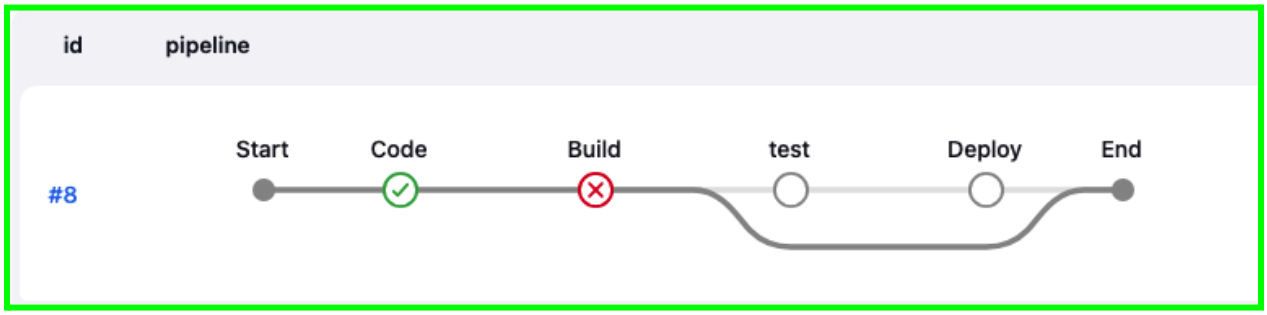


==========

Pipeline Failed-1

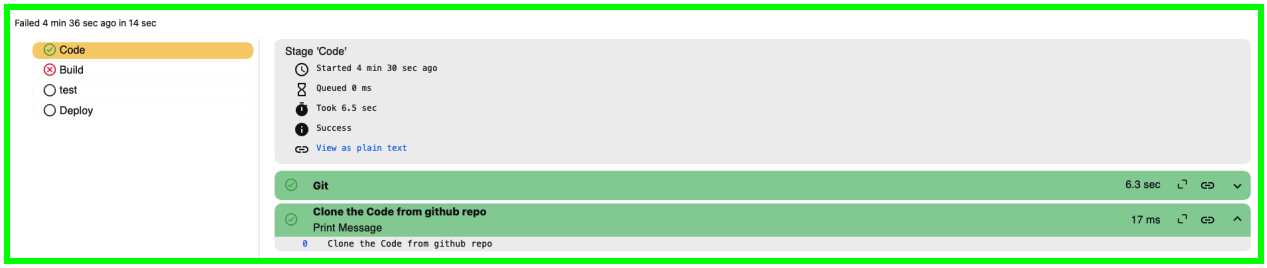
====> Let's check the reason for failure.

Click —> Stages → Build stage has failed.



====> Click on Build Stage, it will open up the Pipeline Console Window.

1. Code has been cloned from github successfully.
2. Build stage has failed due to **docker not found.**



===> Jenkins Agent Machine doesn't have docker installed, let's install & re-run the build.

==========

Solution1

 # Install Docker into Agent-Linux Machine.

$ sudo apt-get update

$ sudo apt-get install docker.io

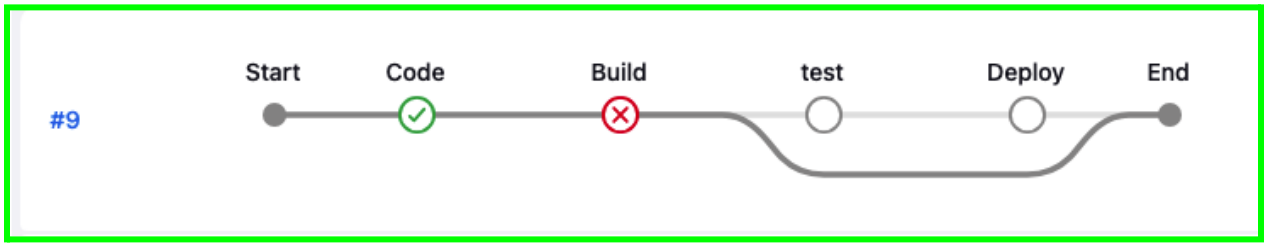
```
ubuntu@ip-172-31-35-243:~$ sudo apt-get update
Hit:1 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://eu-central-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
ubuntu@ip-172-31-35-243:~$
ubuntu@ip-172-31-35-243:~$
ubuntu@ip-172-31-35-243:~$
ubuntu@ip-172-31-35-243:~$ sudo apt-get install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

Since docker has been installed, let's re-run the build stage again.

==========

Pipeline Failed-2

===> The New Pipeline has also failed, let's check it again.

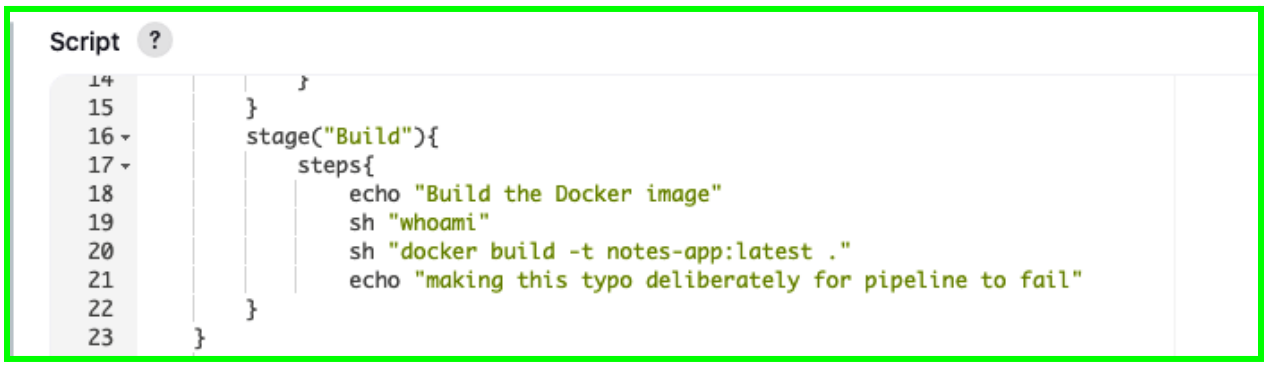


Failure Reason :---> Build has failed due to docker Permission Denied.

==========

Solution2

1. We will check which user is having a permission denied.
2. Add a shell command in the build stage “whoami” in the script pipeline.



Jenkins Agent User Ubuntu is getting the Permission denied, Let’s add Ubuntu User into Docker Group.

$ cat /etc/group : It will show you if an ubuntu user has docker Group access.

```
ubuntu@ip-172-31-35-243:~$ cat /etc/group
```

Ubuntu User does not have access , **Let's add into Docker Group.**

$ sudo usermod -aG docker ubuntu : This Command will add User into Docker Group.

```
ubuntu:x:1000:
docker:x:113:
ubuntu@ip-172-31-35-243:~$ sudo usermod -aG docker ubuntu
```

$ cat /etc/group : Ubuntu user has access now.

```
docker:x:113:ubuntu
```

$ docker ps : tried running this command , still getting permission denied, which is kind of a common issue. **Let's fix this.**

```
ubuntu@ip-172-31-35-243:~$ docker ps
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get "http://%2Fvar%2Frun%
2Fdocker.sock/v1.24/containers/json": dial unix /var/run/docker.sock: connect: permission denied
ubuntu@ip-172-31-35-243:~$
```

$ newgrp docker : This command will Activate the changes to groups. Now the **"docker ps"** command is running,

```
ubuntu@ip-172-31-35-243:~$ newgrp docker
ubuntu@ip-172-31-35-243:~$
ubuntu@ip-172-31-35-243:~$
ubuntu@ip-172-31-35-243:~$
ubuntu@ip-172-31-35-243:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@ip-172-31-35-243:~$
```

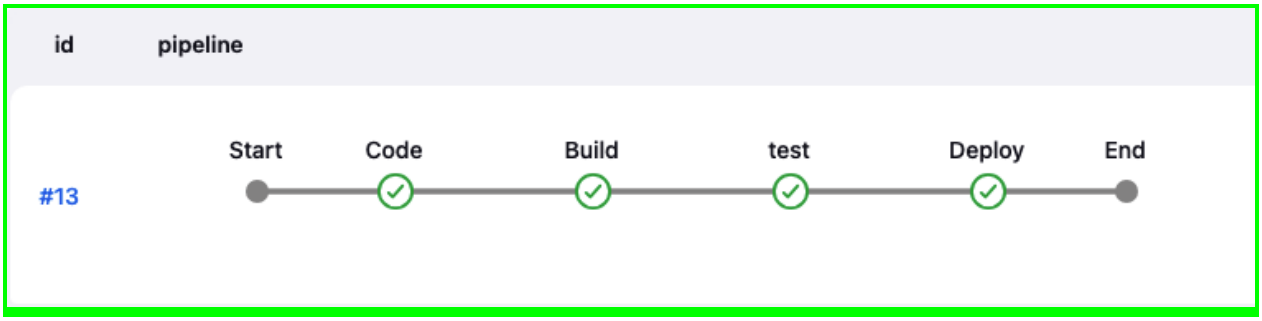
Note : if you still get the permission error, we can reboot the EC2 machine once.

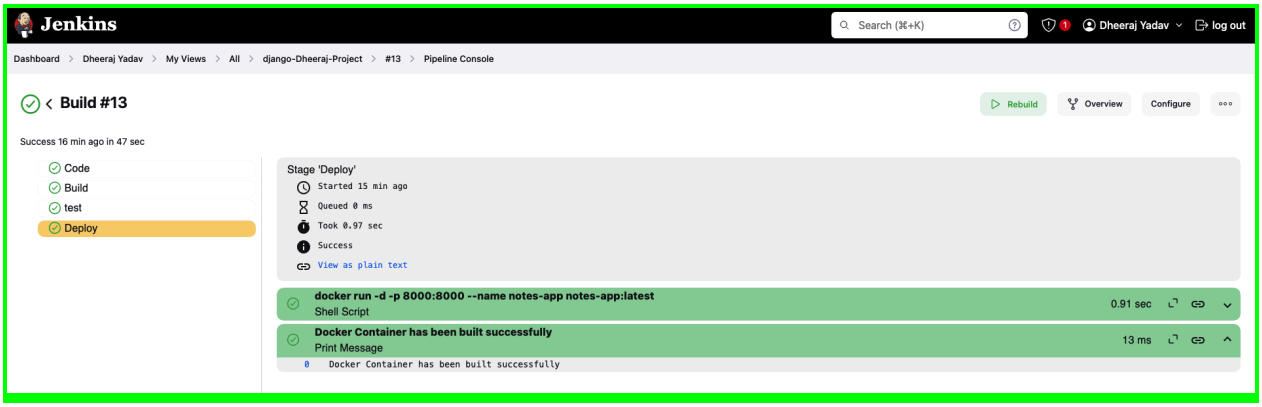
```
ubuntu@ip-172-31-35-243:~$ sudo reboot

Broadcast message from root@ip-172-31-35-243 on pts/0 (Sat 2024-08-03 19:37:25 UTC):

The system will reboot now!
```

Let's build our pipeline again

Nice, it has worked this time, Container has been deployed.

Let's Verify it.





==========

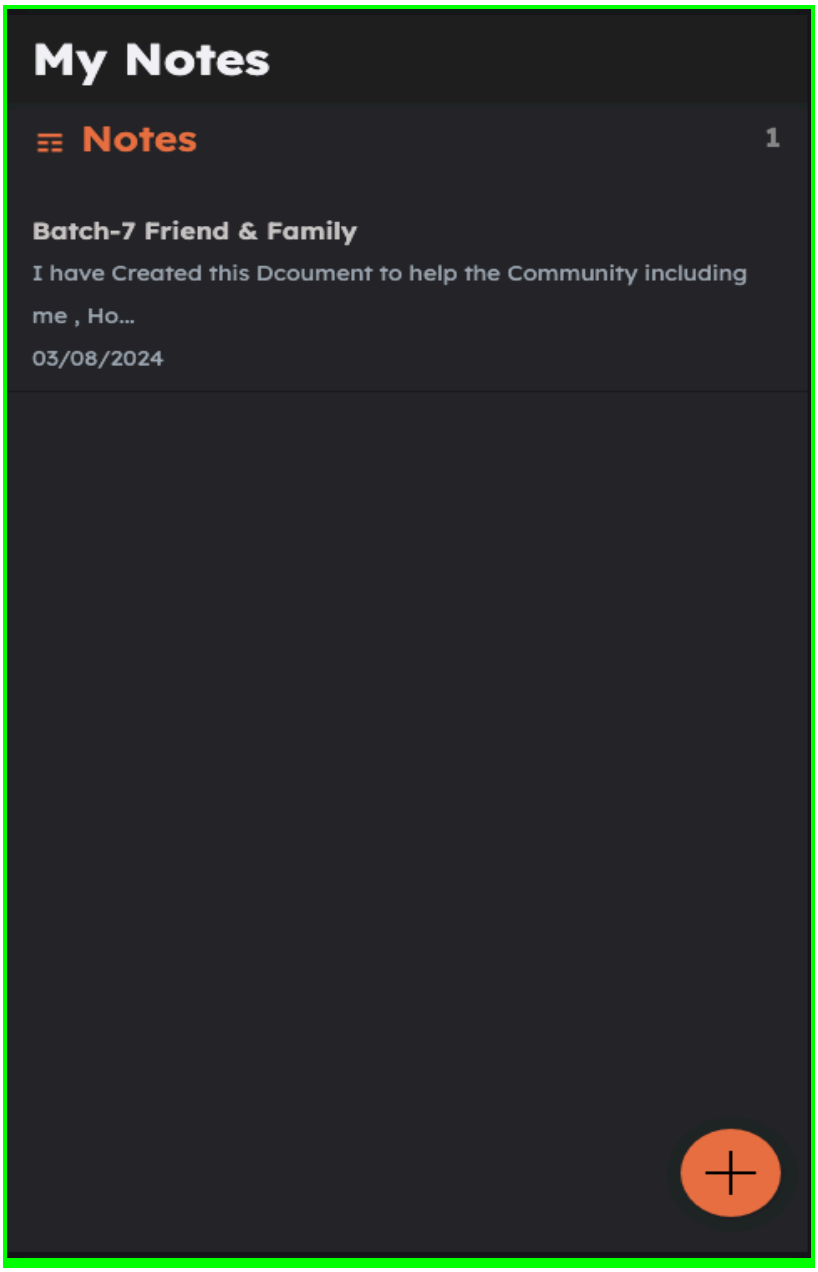
Verify the Build

$ docker ps : It is showing that Container has been Deployed Successfully.

```
ubuntu@ip-172-31-35-243:~$
ubuntu@ip-172-31-35-243:~$ docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED         STATUS         PORTS                                       NAMES
8eac590723ff   notes-app:latest   "/bin/sh -c 'python …"   16 minutes ago  Up 16 minutes  0.0.0.0:8000->8000/tcp, :::8000->8000/tcp   notes-app
ubuntu@ip-172-31-35-243:~$
```

Note: Notes application is running on port 8000, please make sure port 8000 has been opened under **"inbound rules"** Jenkins Agent EC2 Machine.

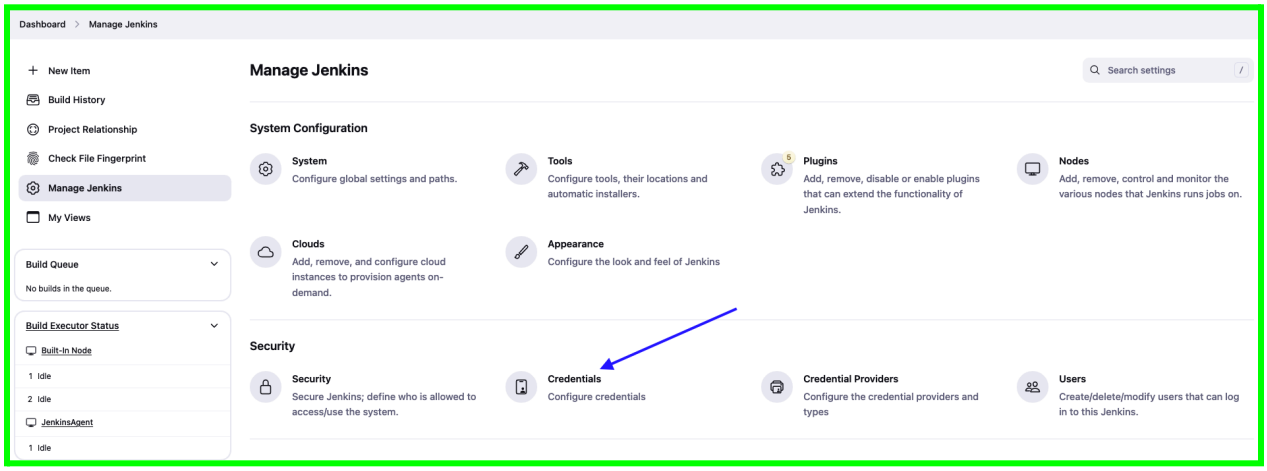
Notes app Project has been deployed through Jenkins Pipeline.



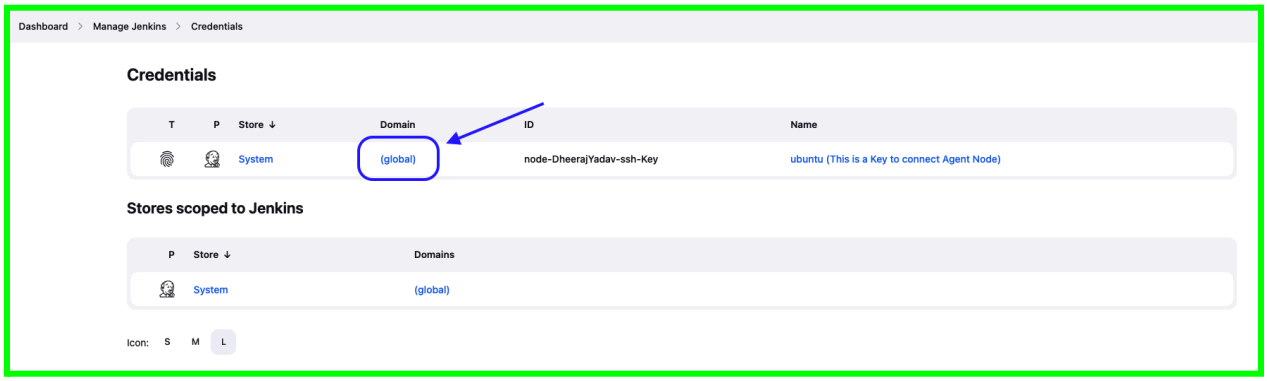
==========

Step -8 : Credential Binding & Image Pushed to DockerHub

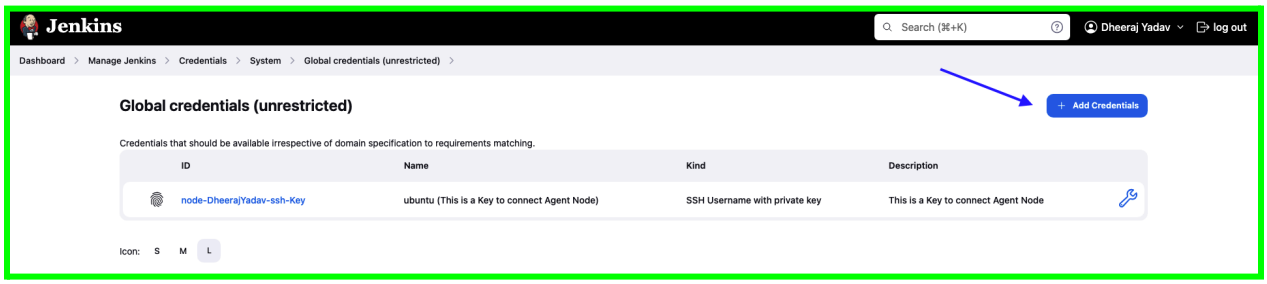
Manage Jenkins—-> Security —> Credentials : To Add the user & DockerHub Token here.



Click on the global icon.



Click on **Add Credentials**

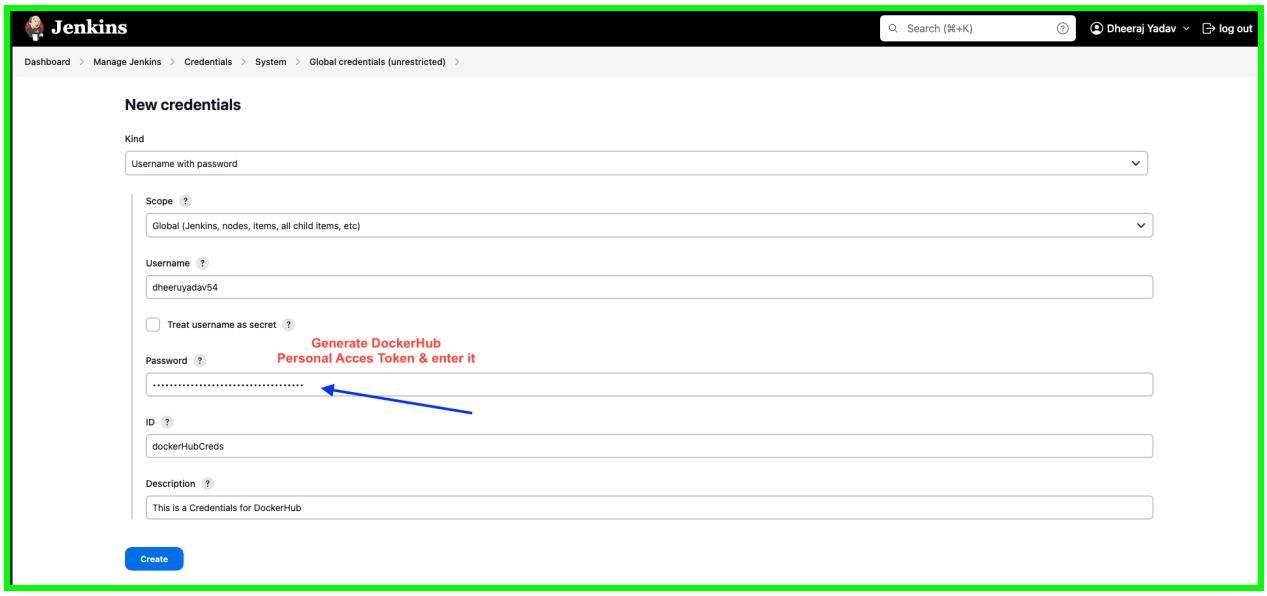


Kind : Username with password

Username : dockerHub Username

Password : generate the Personal Access token per below steps.

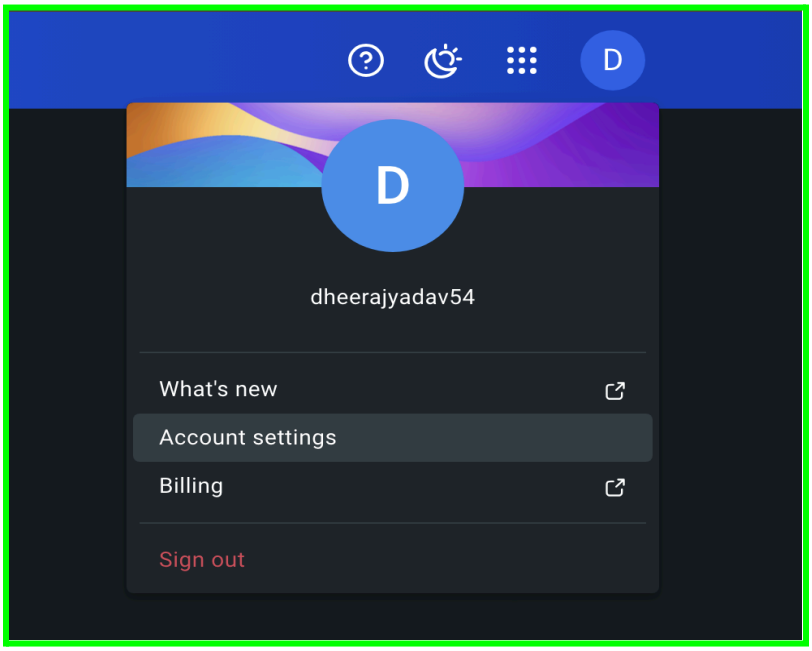
ID : It will be representing username & password so in the script pipeline we will use this ID.



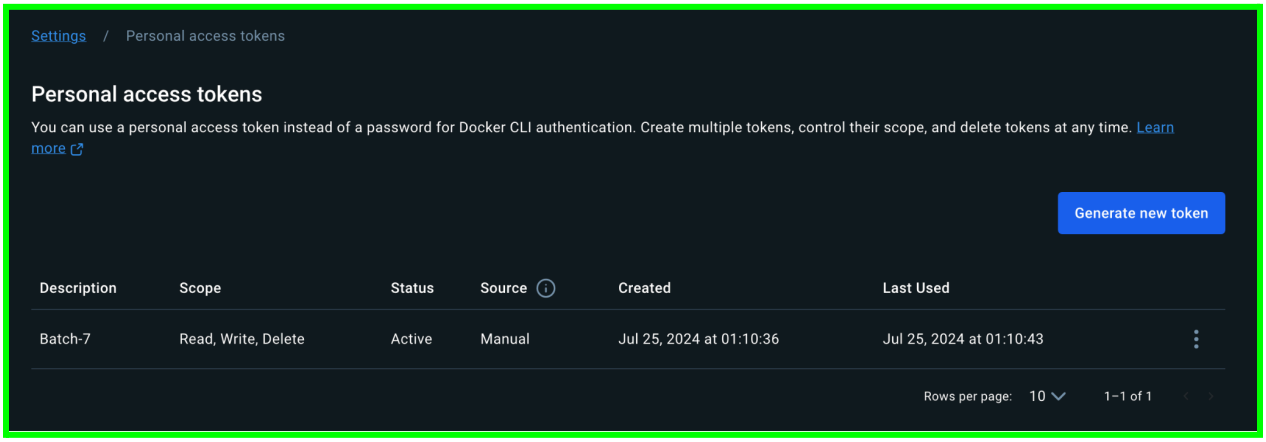
==========

Generate the Personal Access Token on Docker Hub

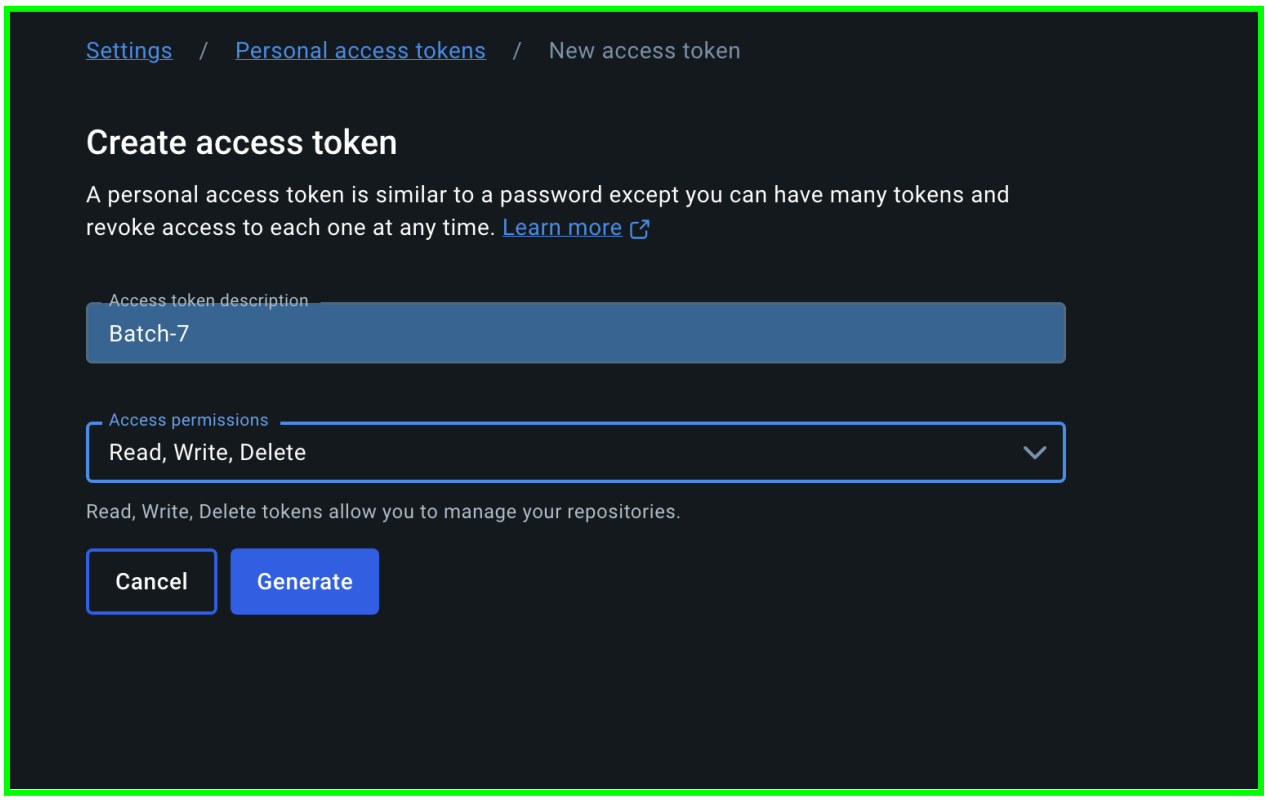
a) Docker Hub Account —-> Account Setting



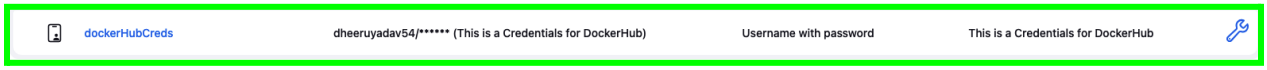
b) Personal access tokens → Generate new token



Click on Generate & Copy the token. Once you Copied the Token, go to Jenkins Credentials page & paste under Password Section.



Now we have Credentials configured, let's build the Pipeline & push the image to docker HUB.



==========

Build the Docker Compose pipeline

Note : we need to have docker compose installed, I have added the command in the pipeline itself. Also stopping & removing the existing docker container, Marked in RED Box.

Script ?

```
pipeline{

    agent{
        node{
            label "dev"
        }
    }

    stages{
        stage("Code"){
            steps{
                git url: "https://github.com/dheeruyadav54/django-notes-app.git", branch: "main"
                echo "Clone the Code from github repo"
            }
        }
        stage("Build"){
            steps{
                echo "Build the Docker image"
                sh "whoami"
                sh "docker build -t notes-app-jenkins:latest ."
                echo "making this typo deliberately for pipeline to fail"
            }
        }
        stage("Push to DockerHub"){
            steps{
                echo "Install the Docker Compose package into Jenkins Agent Machine"
                sh "sudo apt-get install docker-compose-v2"
                echo "Push the image to DockerHub"
                withCredentials(
                    [usernamePassword(
                        credentialsId:"dockerHubCreds",
                        passwordVariable:"dockerHubpass",
                        usernameVariable:"dockerHubuser"
                    )
                ]
                ){
                    sh "docker image tag notes-app-jenkins:latest ${env.dockerHubuser}/notes-app-jenkins:latest"
                    sh "docker login -u ${env.dockerHubuser} -p ${env.dockerHubpass}"
                    sh "docker push ${env.dockerHubuser}/notes-app-jenkins:latest"
                }
            }
        }
        stage("Deploy"){
            steps{
                sh "docker stop 8eac590723ff && docker rm 8eac590723ff && docker compose up -d"
                echo "Docker Container has been built successfully"
            }
        }
    }
}
```

Let's Build the Pipeline & push the image to DockerHub:

Pipeline Succeed

Console Output

```
Started by user Dheeraj Yadav
[Pipeline] Start of Pipeline
[Pipeline] node
Running on JenkinsAgent in /home/ubuntu/apps/workspace/django-Project
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Code)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
Fetching changes from the remote Git repository
Checking out Revision d678d955ad1b68a775a5751c1630c7e422cea5c5 (refs/remotes/origin/main)
Commit message: "Added my DockerHub UserName"
 > git rev-parse --resolve-git-dir /home/ubuntu/apps/workspace/django-Project/.git # timeout=10
 > git config remote.origin.url https://github.com/dheeruyadav54/django-notes-app.git # timeout=10
Fetching upstream changes from https://github.com/dheeruyadav54/django-notes-app.git
 > git --version # timeout=10
 > git --version # 'git version 2.43.0'
 > git fetch --tags --force --progress -- https://github.com/dheeruyadav54/django-notes-app.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/main^{commit} # timeout=10
 > git config core.sparsecheckout # timeout=10
 > git checkout -f d678d955ad1b68a775a5751c1630c7e422cea5c5 # timeout=10
 > git branch -a -v --no-abbrev # timeout=10
 > git branch -D main # timeout=10
 > git checkout -b main d678d955ad1b68a775a5751c1630c7e422cea5c5 # timeout=10
 > git rev-list --no-walk d678d955ad1b68a775a5751c1630c7e422cea5c5 # timeout=10
[Pipeline] echo
Clone the Code from github repo
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Build)
[Pipeline] echo
Build the Docker image
[Pipeline] sh
+ whoami
ubuntu
[Pipeline] sh
+ docker build -t notes-app-jenkins:latest .
```

Image has been pushed to DockerHub

```
Push the image to DockerHub
[Pipeline] withCredentials
Masking supported pattern matches of $dockerHubpass
[Pipeline] {
[Pipeline] sh
+ docker image tag notes-app-jenkins:latest dheerajyadav54/notes-app-jenkins:latest
[Pipeline] sh
Warning: A secret was passed to "sh" using Groovy String interpolation, which is insecure.
                 Affected argument(s) used the following variable(s): [dockerHubpass]
                 See https://jenkins.io/redirect/groovy-string-interpolation for details.
+ docker login -u dheerajyadav54 -p ****
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[Pipeline] sh
+ docker push dheerajyadav54/notes-app-jenkins:latest
The push refers to repository [docker.io/dheerajyadav54/notes-app-jenkins]
ab23de4977a9: Preparing
e88bdbea9065: Preparing
```

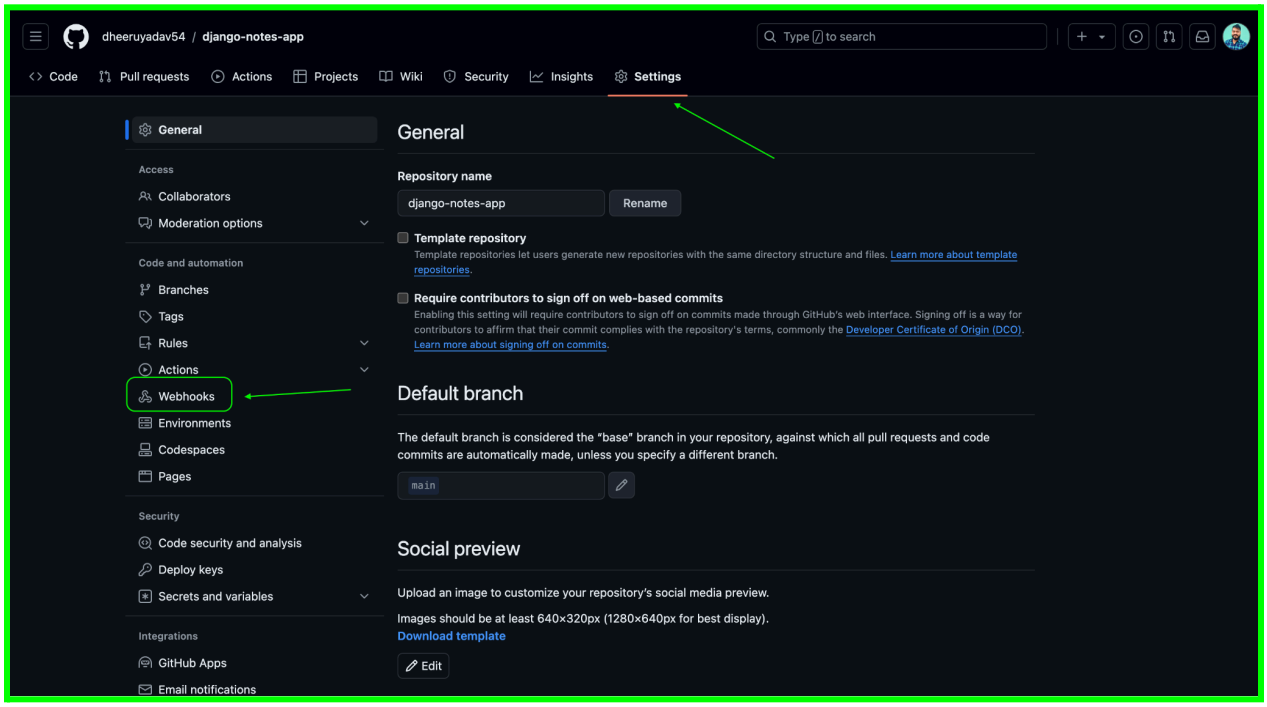
Pipeline Succeed & removed the existing Docker container too.

```
ab23de4977a9: Pushed
latest: digest: sha256:7ef6bff5ece5643a0da29da3fa47932706711dc4d8cefd24cefdc23e10b85d31 size: 2844
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Deploy)
[Pipeline] sh
+ docker stop 8eac590723ff
8eac590723ff
+ docker rm 8eac590723ff
8eac590723ff
+ docker compose up -d
 Network django-project_default  Creating
 Network django-project_default  Created
 Container django-project-web-1  Creating
 Container django-project-web-1  Created
 Container django-project-web-1  Starting
 Container django-project-web-1  Started
[Pipeline] echo
Docker Container has been built successfully
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

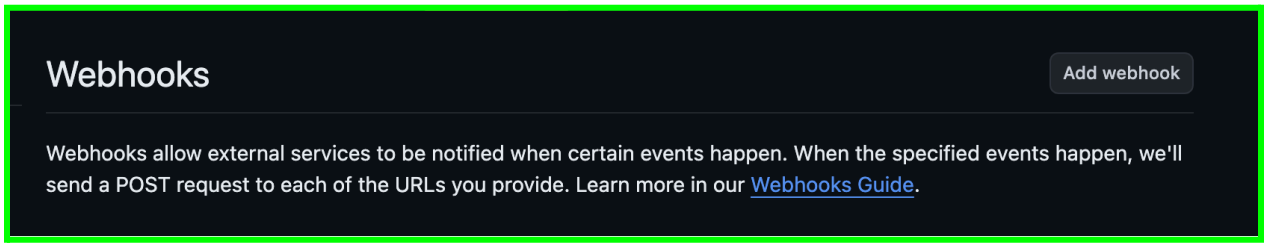
==========

Step -9 : Jenkins Webhook

Go to your Github repository —-> Setting —> webhooks

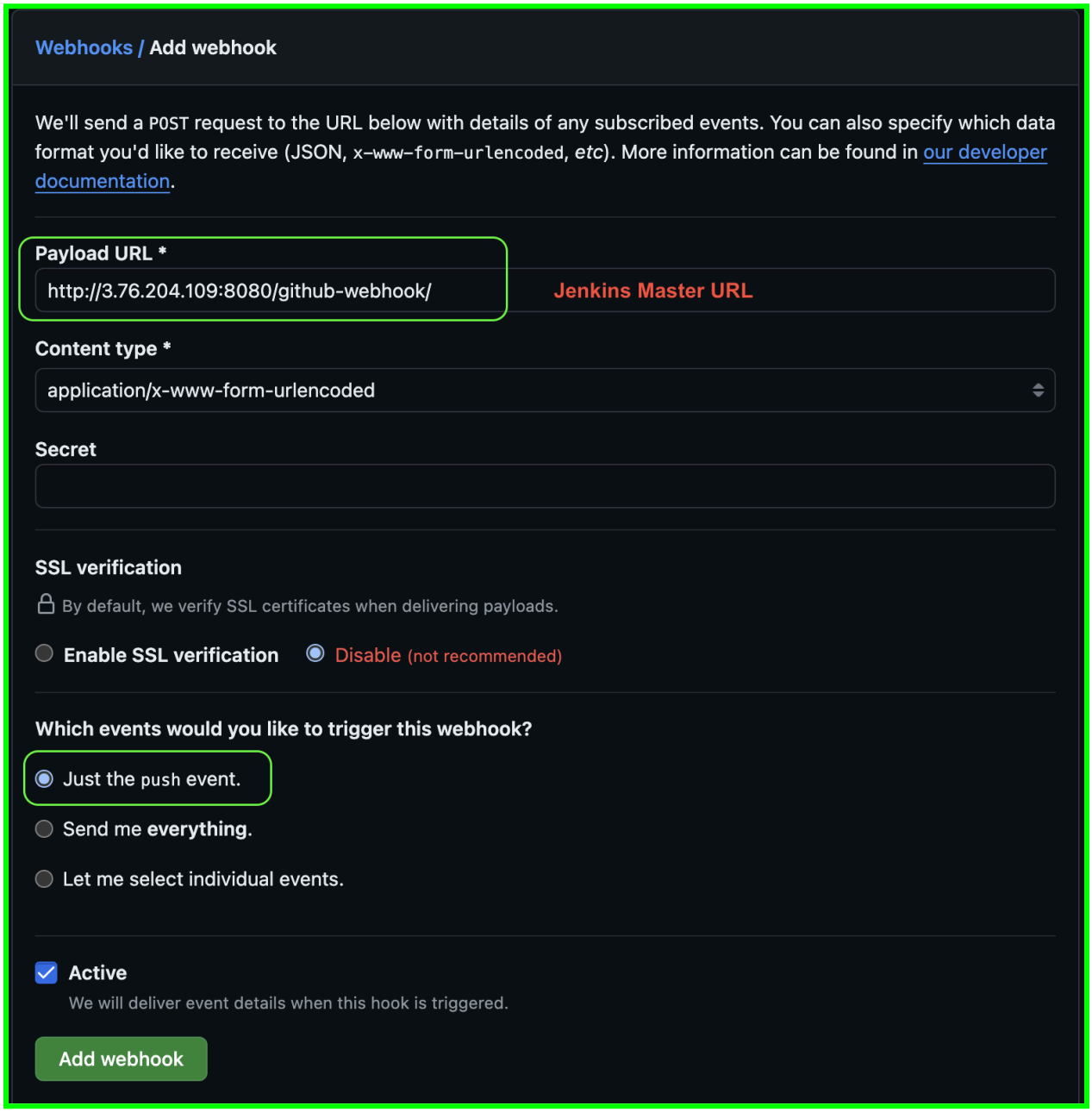


Add Webhook —> it might ask your github password to login again.



 # Payload URL : This is a Jenkins Master URL
http://<EC@PublicIP>:8080/github-webhook

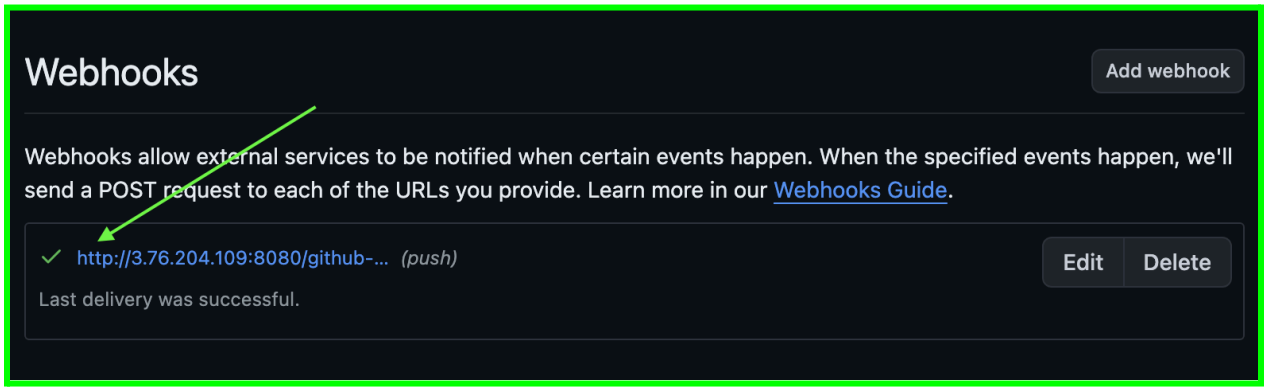
SSL verification : We have disabled it, since it will require certificate handshake.
Just the Push Event : Selected this Option, meaning whenever there is a push in the repository.



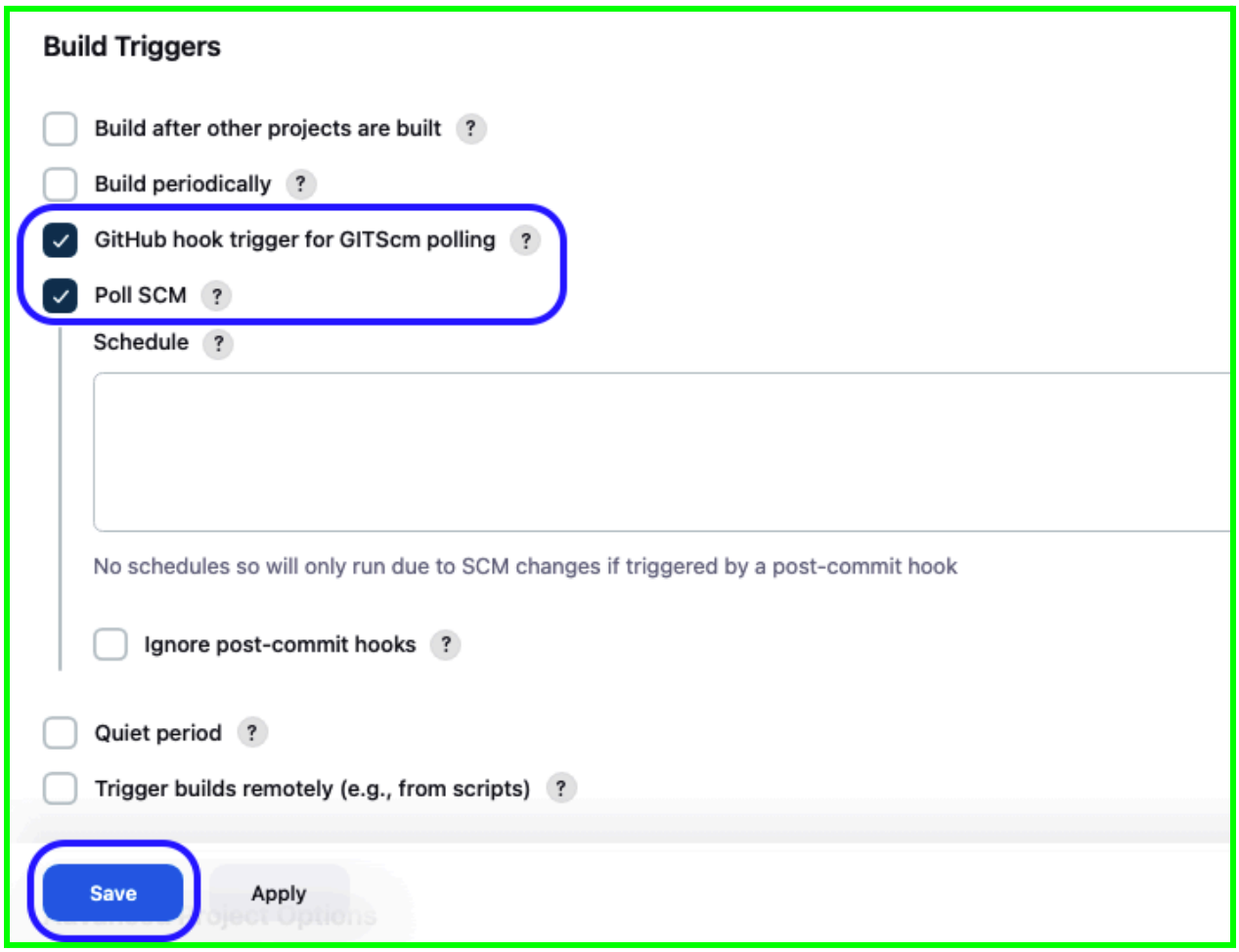
==========

Webhook Connection Status & build Tgigger Config

It will take a few seconds to become Active.



Enable the GitHub hook Trigger & Poll SCM Configuration in Jenkins Project Template.

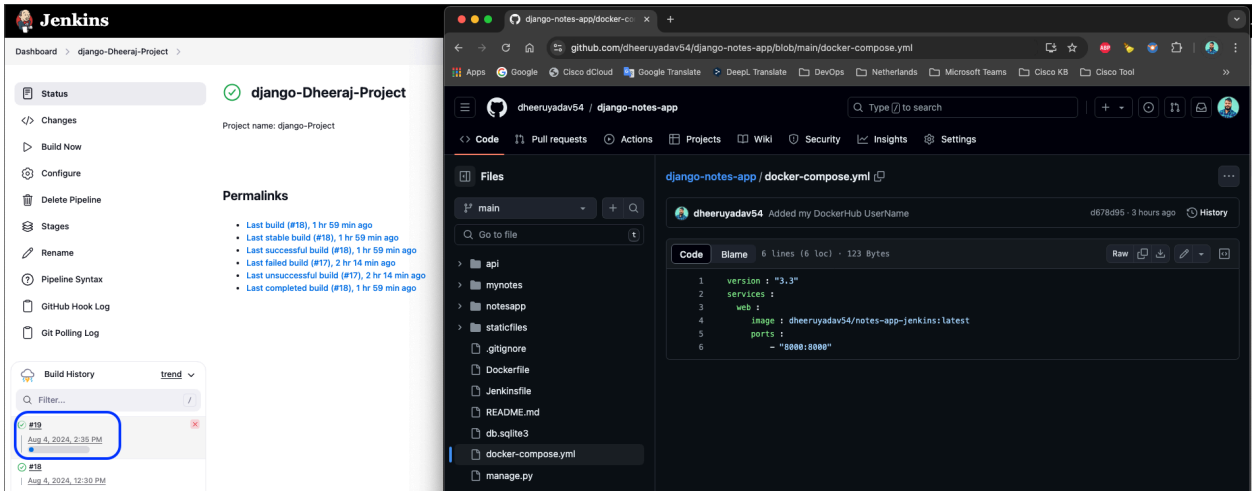


==========

Commit to Trigger Webhook

Now as Soon as you make changes in the github repository it will create a github webhook to jenkins & pipeline will be triggered.

Let's make a small change & commit & see pipeline triggers or not.
As Soon you push a commit it has triggered a new **Pipeline #19**



End of Jenkins Fundamentals