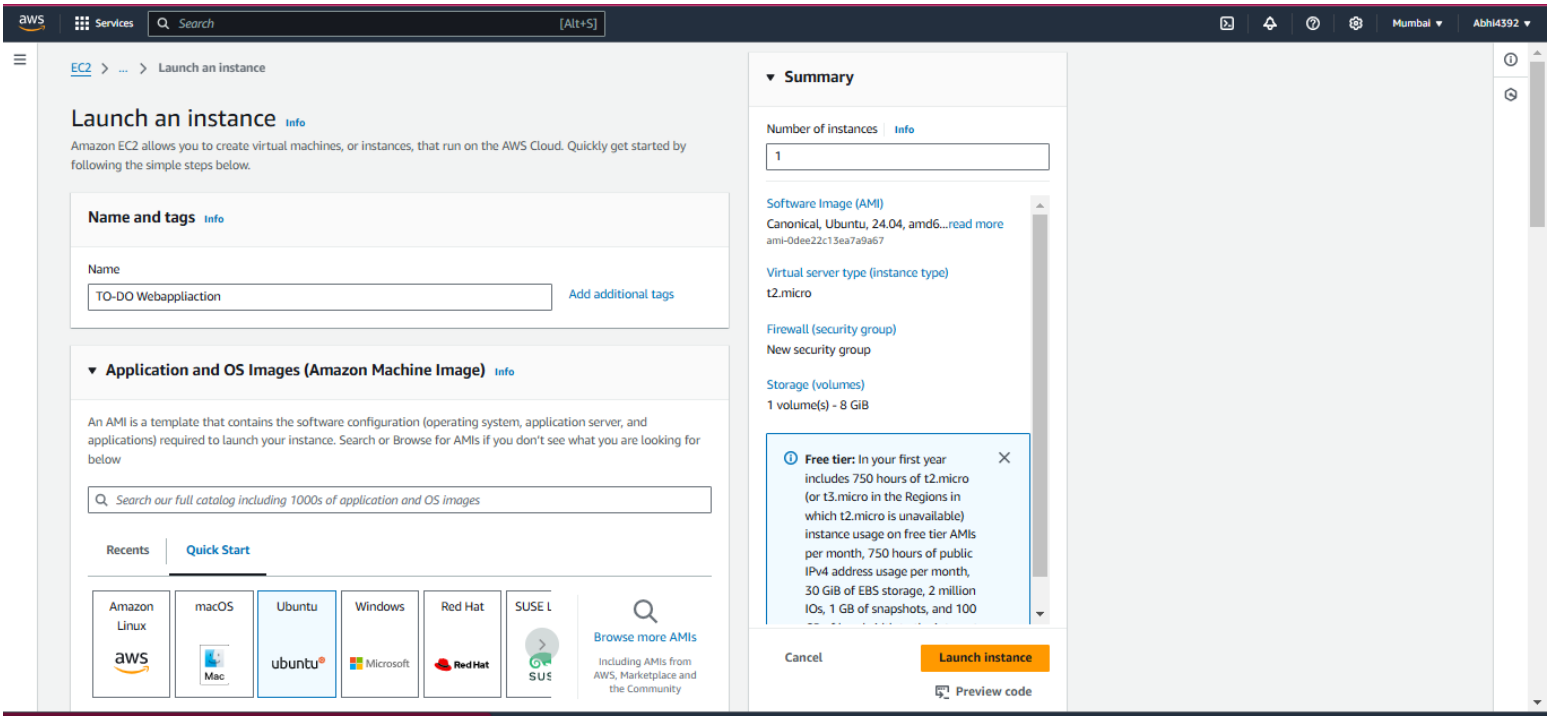
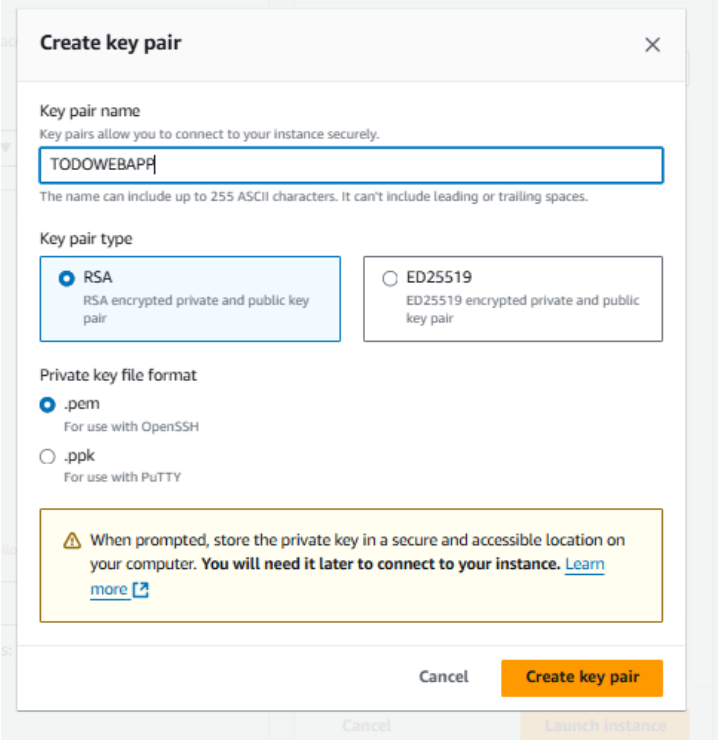
TO-DO WEBAPPLIACTION

Steps:

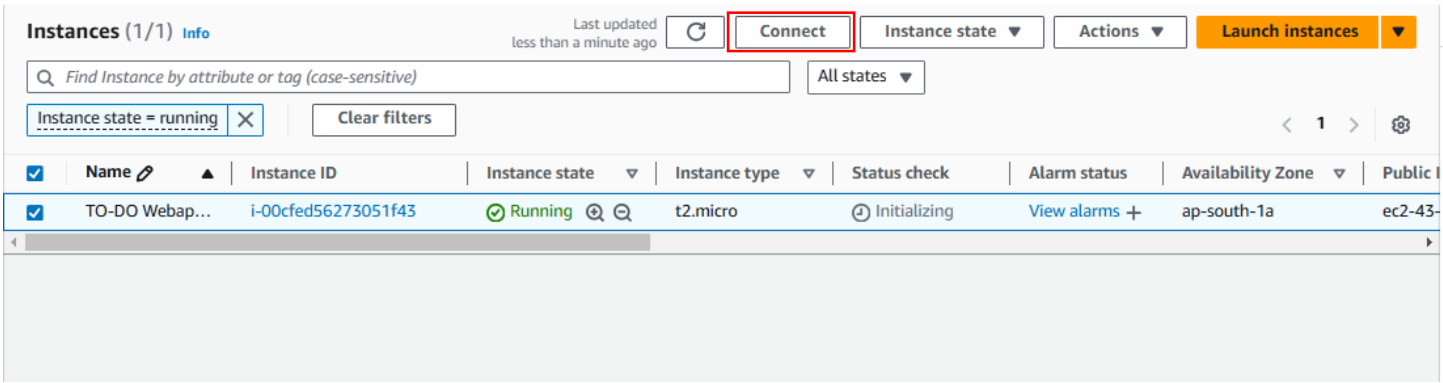
1)Create a virtual machine in AWS EC2 instance where OS is Ubuntu



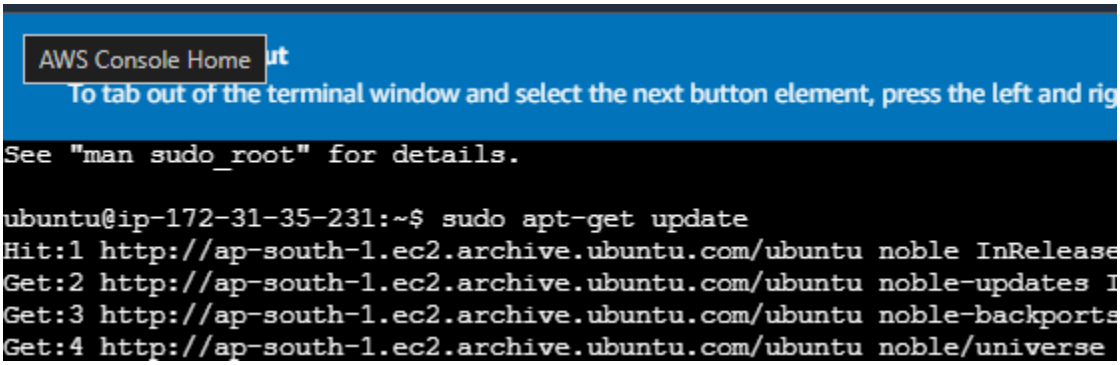
2) Create a Key Pair which will be later use to connect to the instance.



3) Instance is Created and connect the instance.



4) Update the system



5) Install Java in Ubuntu

sudo apt update

sudo apt install fontconfig openjdk-17-jre

java -version

openjdk version "17.0.8" 2023-07-18

OpenJDK Runtime Environment (build 17.0.8+7-Debian-1deb12u1)

OpenJDK 64-Bit Server VM (build 17.0.8+7-Debian-1deb12u1, mixed mode, sharing)

```
ubuntu@ip-172-31-35-231:~$ sudo apt update
sudo apt install fontconfig openjdk-17-jre
java -version
openjdk version "17.0.8" 2023-07-18
OpenJDK Runtime Environment (build 17.0.8+7-Debian-1deb12u1)
OpenJDK 64-Bit Server VM (build 17.0.8+7-Debian-1deb12u1, mixed mode, sharing)
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
41 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
```

6) INSTALLATION OF JENKINS we first generate the key.

**sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key**

```
ubuntu@ip-172-31-35-231:~$ sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
  https://pkg.jenkins.io/debian/jenkins.io-2023.key
```

7) Pushing key into devnull

**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**

```
ubuntu@ip-172-31-35-231:~$ 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
```

8) Now command for installing

Sudo apt-get update (important to do after pushing
sudo apt-get install jenkins

```
ubuntu@ip-172-31-35-231:~$ sudo apt-get update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-south-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]
```

9) start the Jenkins

sudo systemctl enable Jenkins

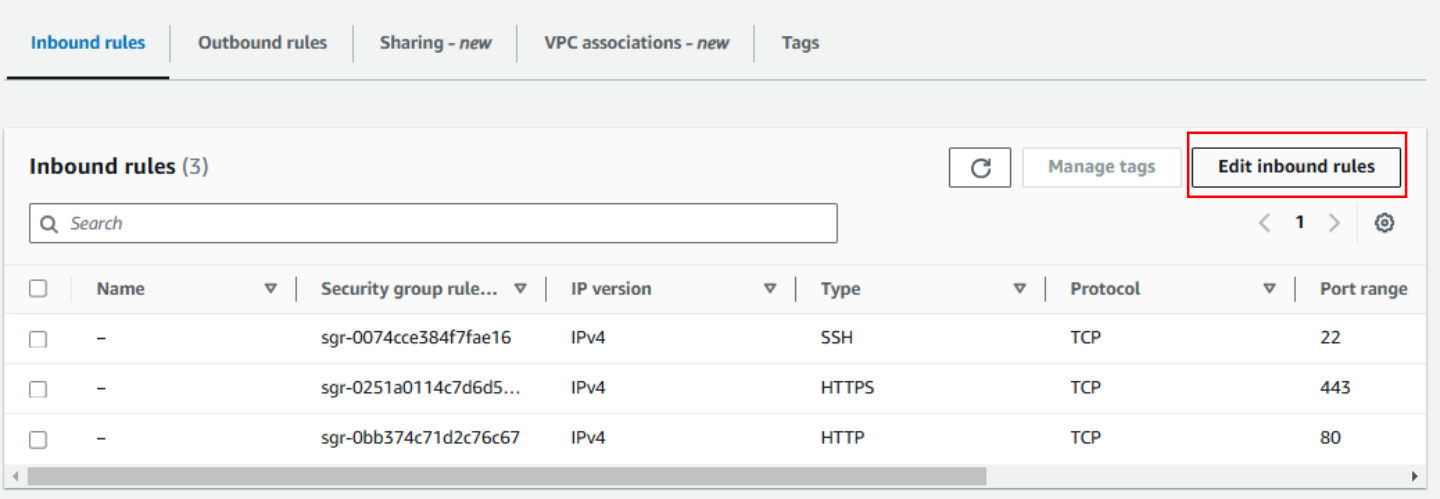
```
ubuntu@ip-172-31-35-231:~$ sudo systemctl enable jenkins
Synchronizing state of jenkins.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable jenkins
ubuntu@ip-172-31-35-231:~$
```

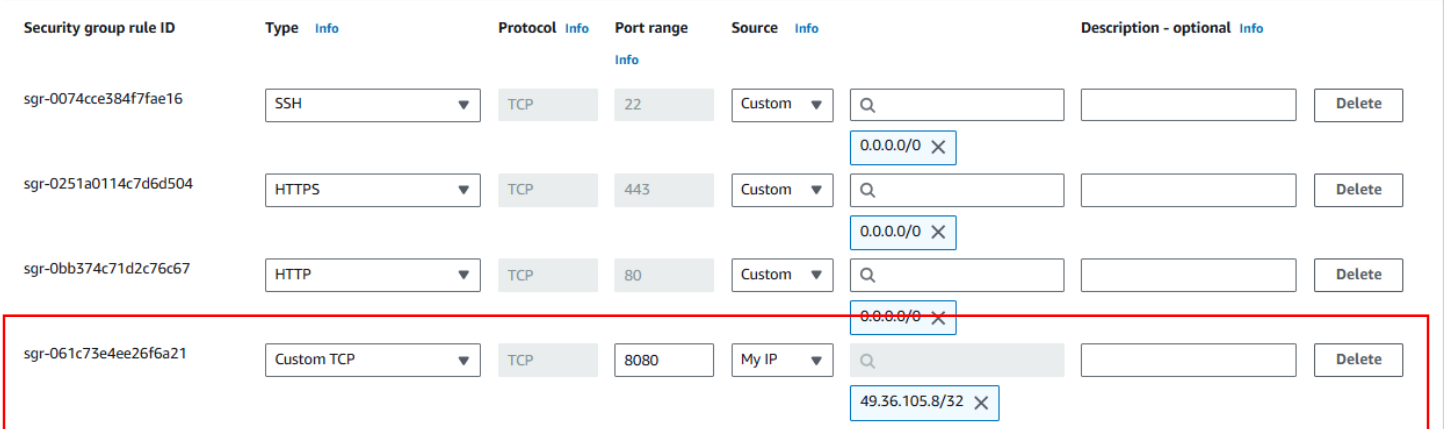
Sudo systemctl start Jenkins

Sudo systemctl stataus Jenkins

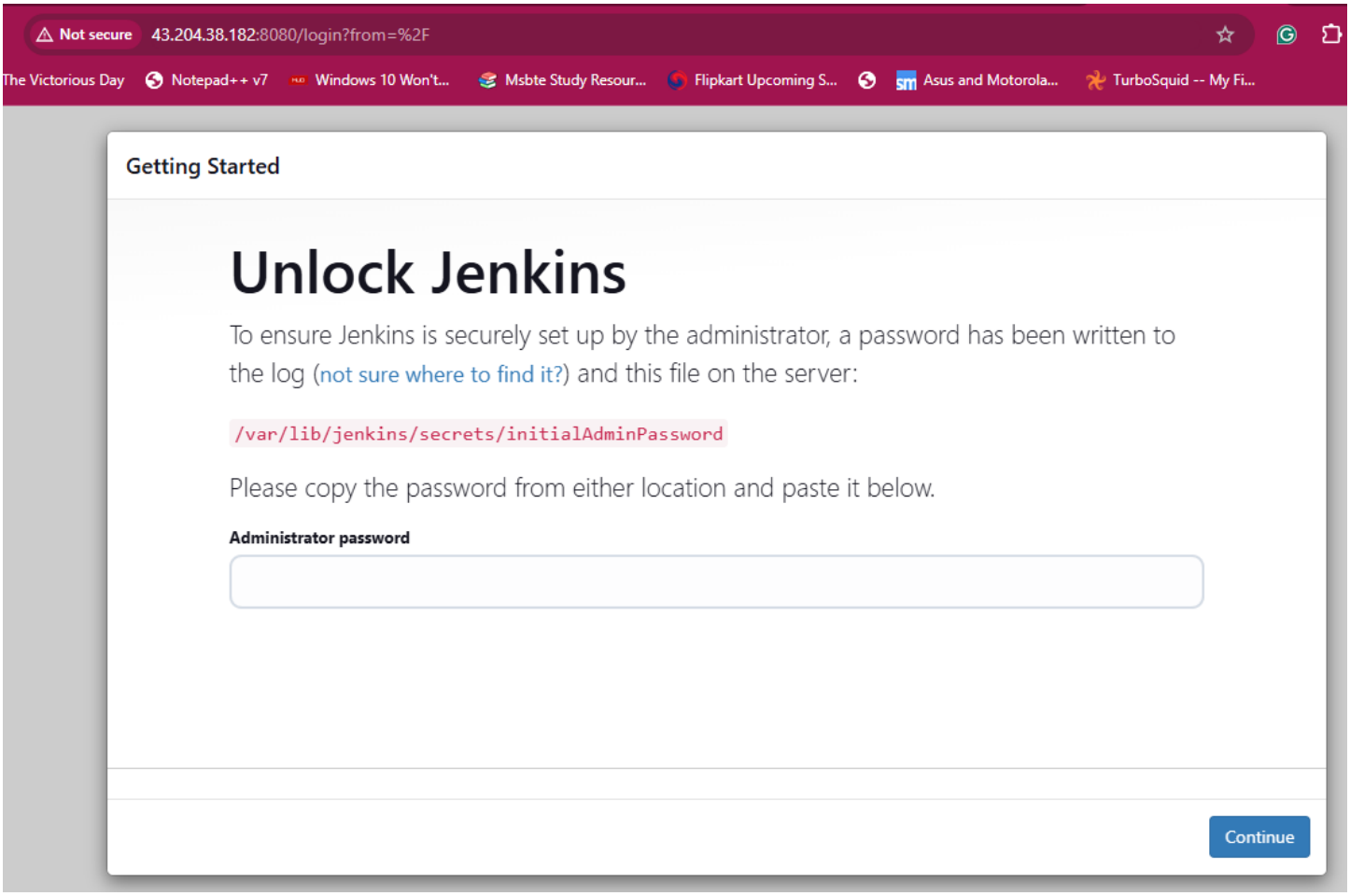
```
ubuntu@ip-172-31-35-231:~$ sudo systemctl start jenkins
ubuntu@ip-172-31-35-231:~$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
     Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
     Active: active (running) since Fri 2024-11-15 05:56:21 UTC; 4min 26s ago
   Main PID: 4572 (java)
      Tasks: 38 (limit: 1130)
     Memory: 293.3M (peak: 338.3M)
        CPU: 18.904s
     CGroup: /system.slice/jenkins.service
             └─4572 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=

Nov 15 05:55:57 ip-172-31-35-231 jenkins[4572]: ef944f392fe94cad9969f1214e1a301f
Nov 15 05:55:57 ip-172-31-35-231 jenkins[4572]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Nov 15 05:55:57 ip-172-31-35-231 jenkins[4572]: ************************************************************
Nov 15 05:55:57 ip-172-31-35-231 jenkins[4572]: ************************************************************
Nov 15 05:55:57 ip-172-31-35-231 jenkins[4572]: ************************************************************
Nov 15 05:56:21 ip-172-31-35-231 jenkins[4572]: 2024-11-15 05:56:21.354+0000 [id=31]        INFO        jenkins.InitReactorRunner$1#onAt
Nov 15 05:56:21 ip-172-31-35-231 jenkins[4572]: 2024-11-15 05:56:21.383+0000 [id=23]        INFO        hudson.lifecycle.Lifecycle#onRea
Nov 15 05:56:21 ip-172-31-35-231 systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Nov 15 05:56:27 ip-172-31-35-231 jenkins[4572]: 2024-11-15 05:56:27.807+0000 [id=46]        INFO        h.m.DownloadService$Downloadable
Nov 15 05:56:27 ip-172-31-35-231 jenkins[4572]: 2024-11-15 05:56:27.808+0000 [id=46]        INFO        hudson.util.Retrier#start: Perfo
lines 1-20/20 (END)
```

10) Jenkins runs on port 8080 so if we are working with AWS EC2 instance we have to change the security group inbound rules and allow port 8080 for only My IP so it can be access by the root user only





11) Open Jenkins in the browser with the public ip and the port 8080

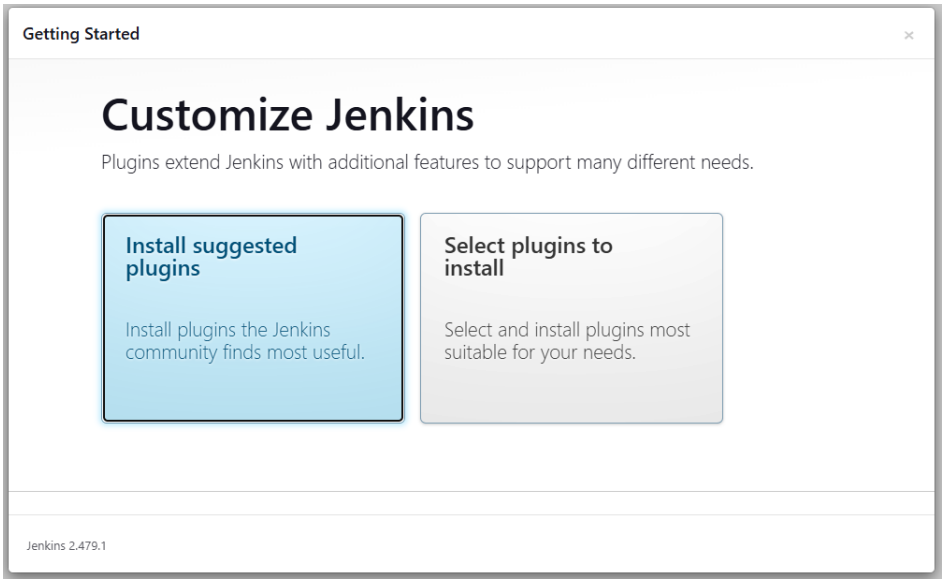


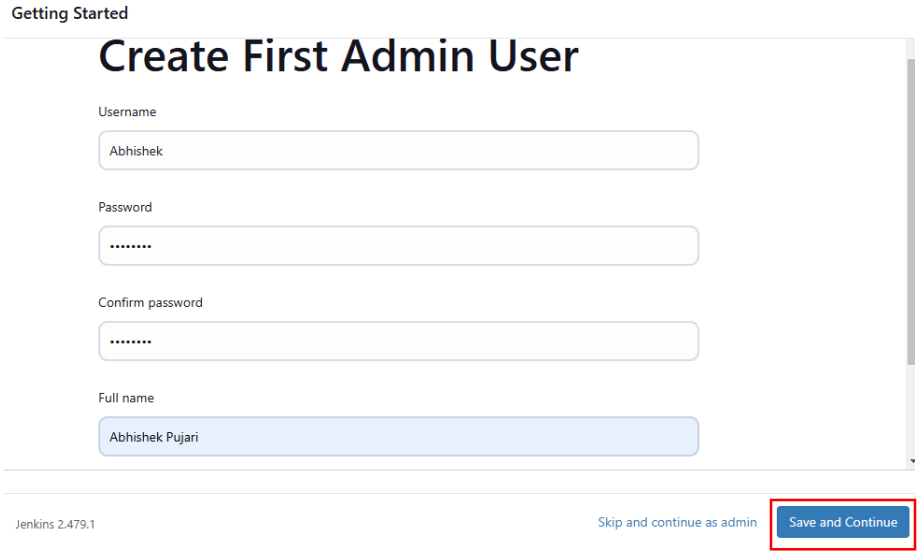
12)Getting the Admin password.

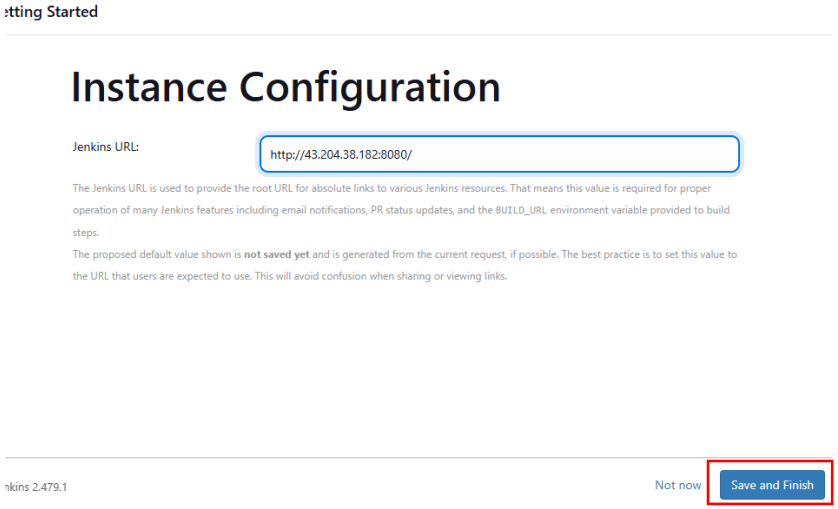
Sudo cat `/var/lib/jenkins/secrets/initialAdminPassword`

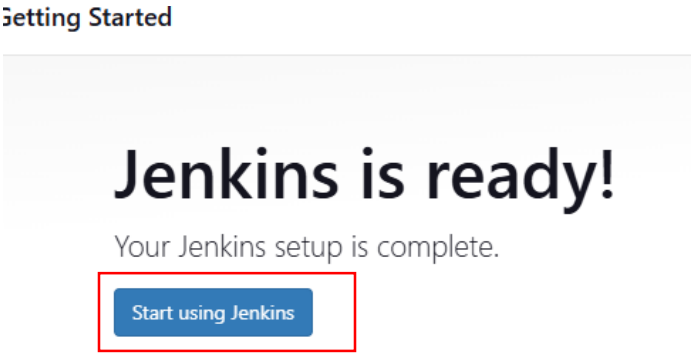
```
ubuntu@ip-172-31-35-231:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
ef944f392fe94cad9969f1214e1a301f
ubuntu@ip-172-31-35-231:~$
```

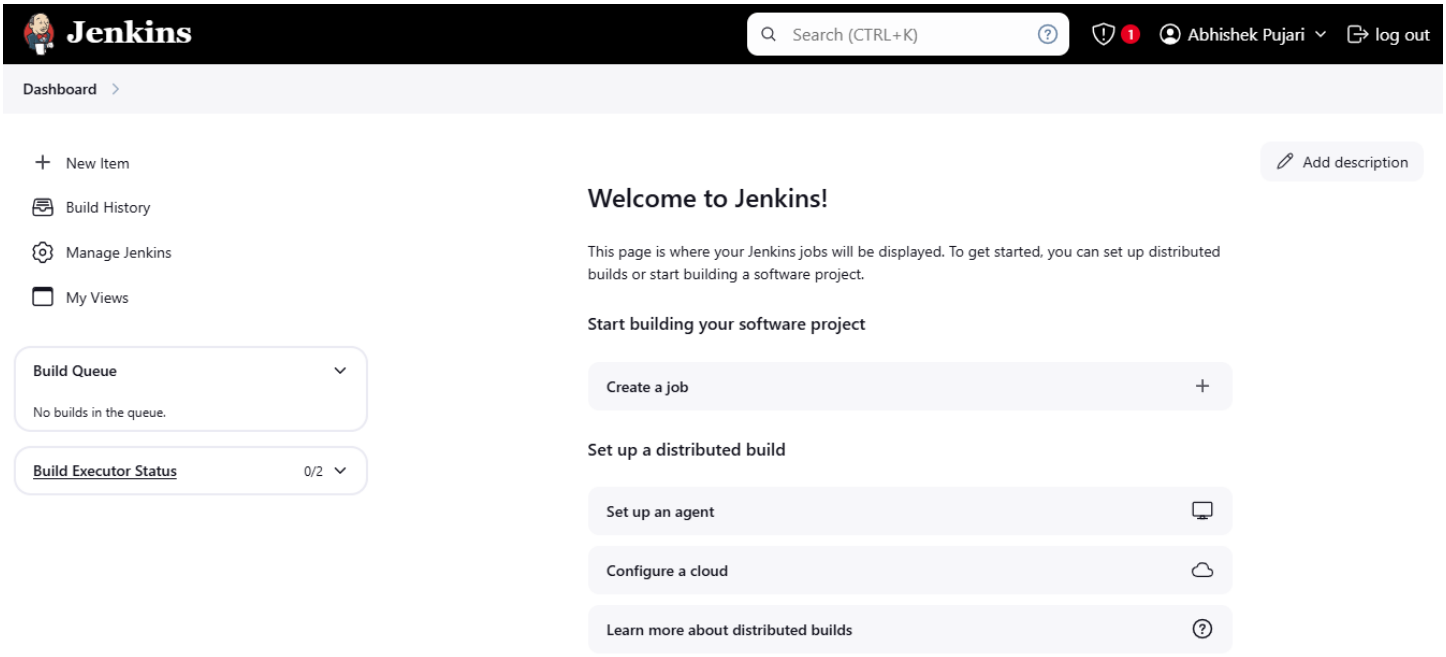
Now copy the code and paste it and continue



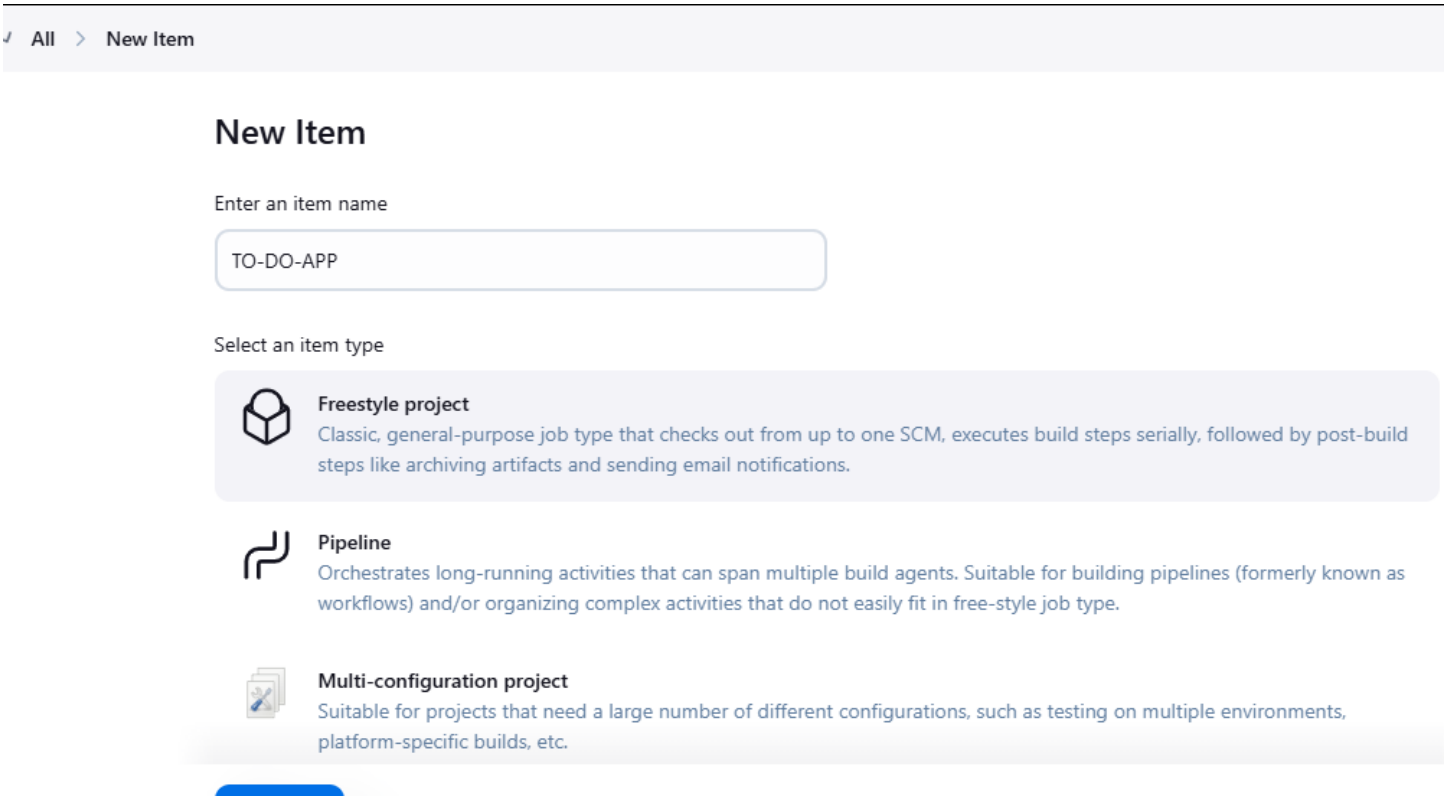

Getting Started
Create First Admin User
Username
Abhishek
Password
Confirm password
Full name
Abhishek Pujari
Jenkins 2.479.1
Skip and continue as admin
Save and Continue


Instance Configuration
Jenkins URL:
http://43.204.38.182:8080/
The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD_URL environment variable provided to build steps.
The proposed default value shown is not saved yet and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.
Not now
Save and Finish

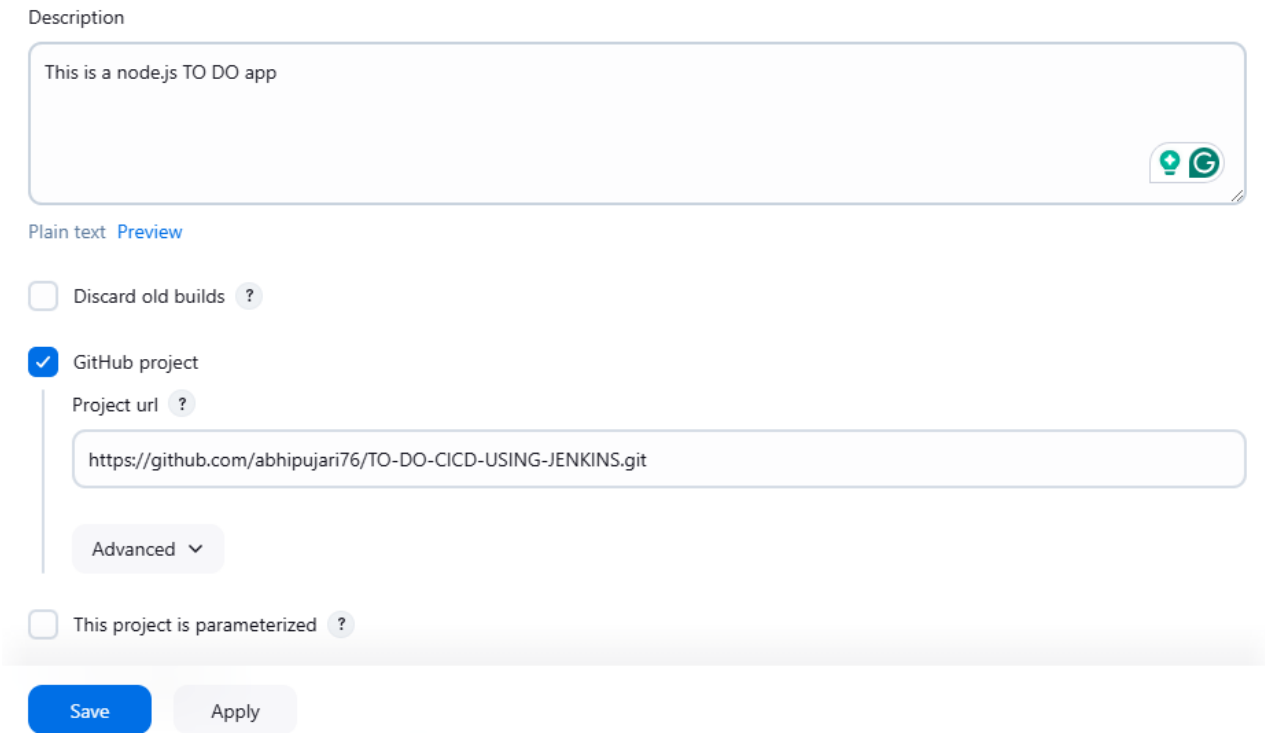

Getting Started
Jenkins is ready!
Your Jenkins setup is complete.
Start using Jenkins


Jenkins
Search (CTRL+K)
Abhishek Pujari
log out
Dashboard
New Item
Build History
Manage Jenkins
My Views
Build Queue
No builds in the queue.
Build Executor Status
0/2
Add description
Welcome to Jenkins!
This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.
Start building your software project
Create a job
Set up a distributed build
Set up an agent
Configure a cloud
Learn more about distributed builds

13) Create a job in Jenkins from create new item.



14) Adding information in General



15) Generating Privte key pair using ssh

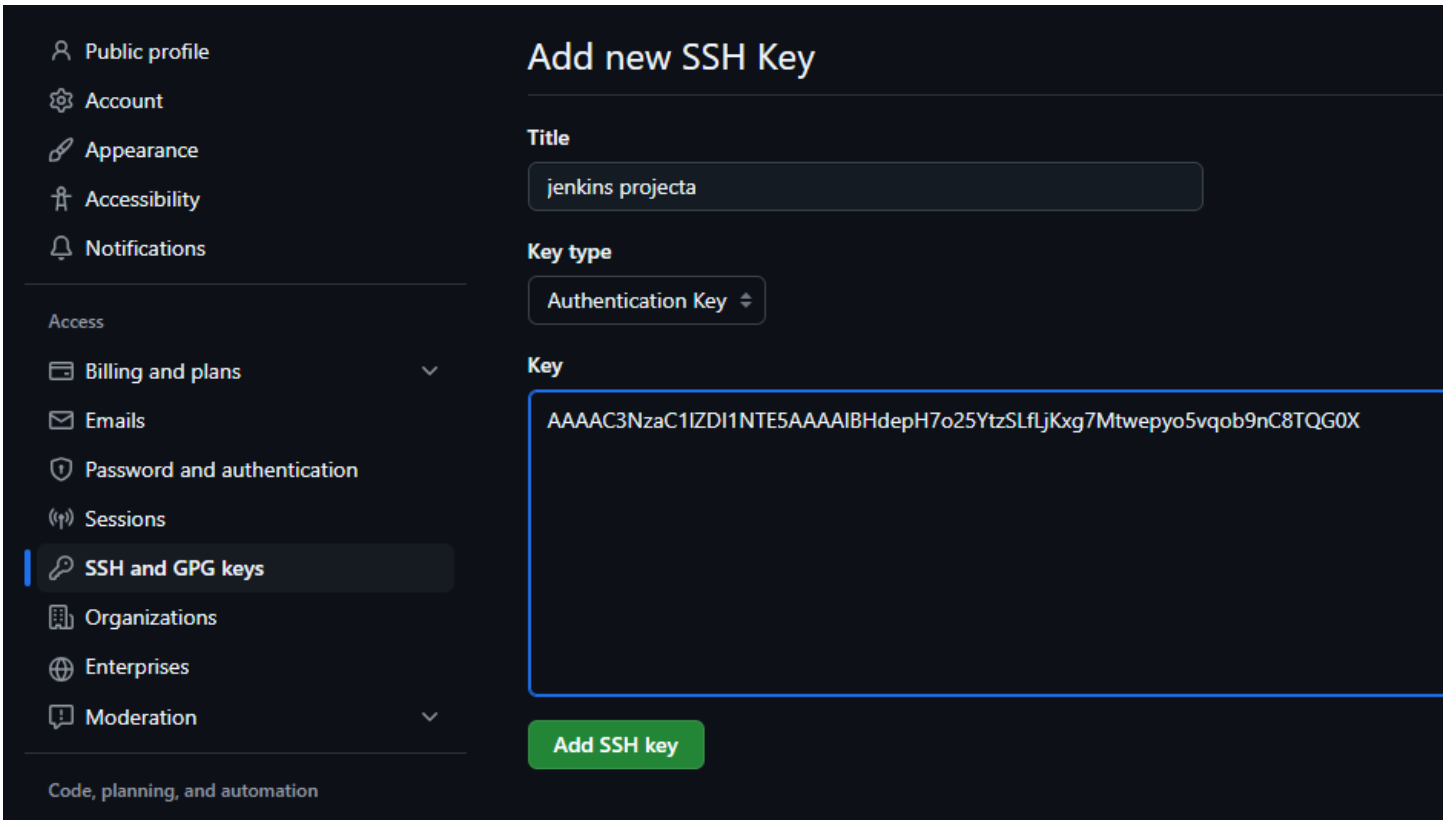
```
ubuntu@ip-172-31-35-231:~$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ubuntu/
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
```

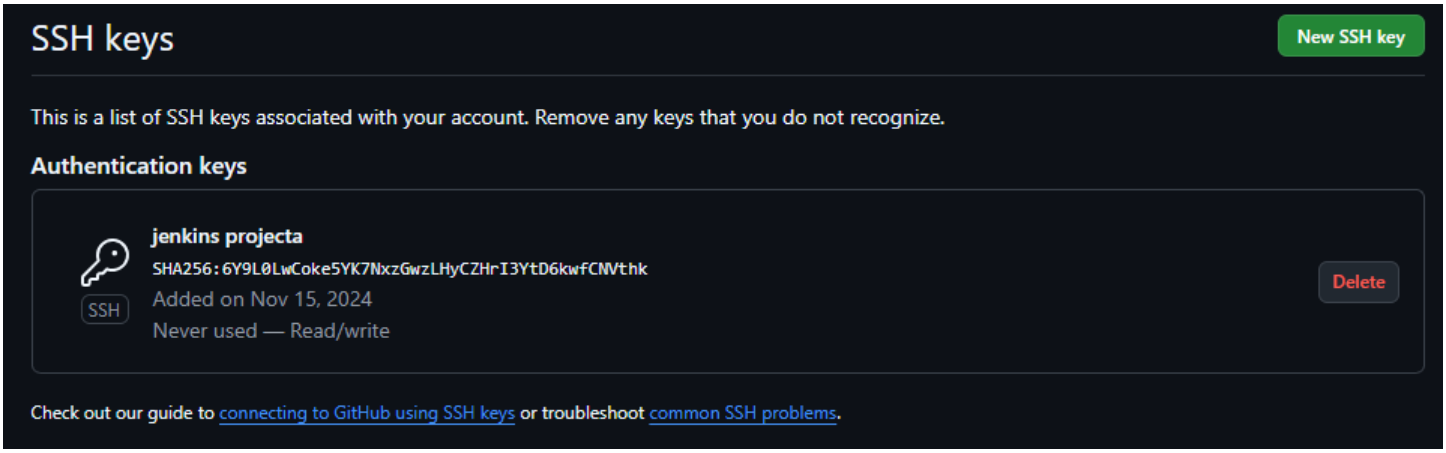
16) Getting key pair location

Cd .ssh > ls > cat id_ed2559.pub(public key) >cat id_ed2559(private key)

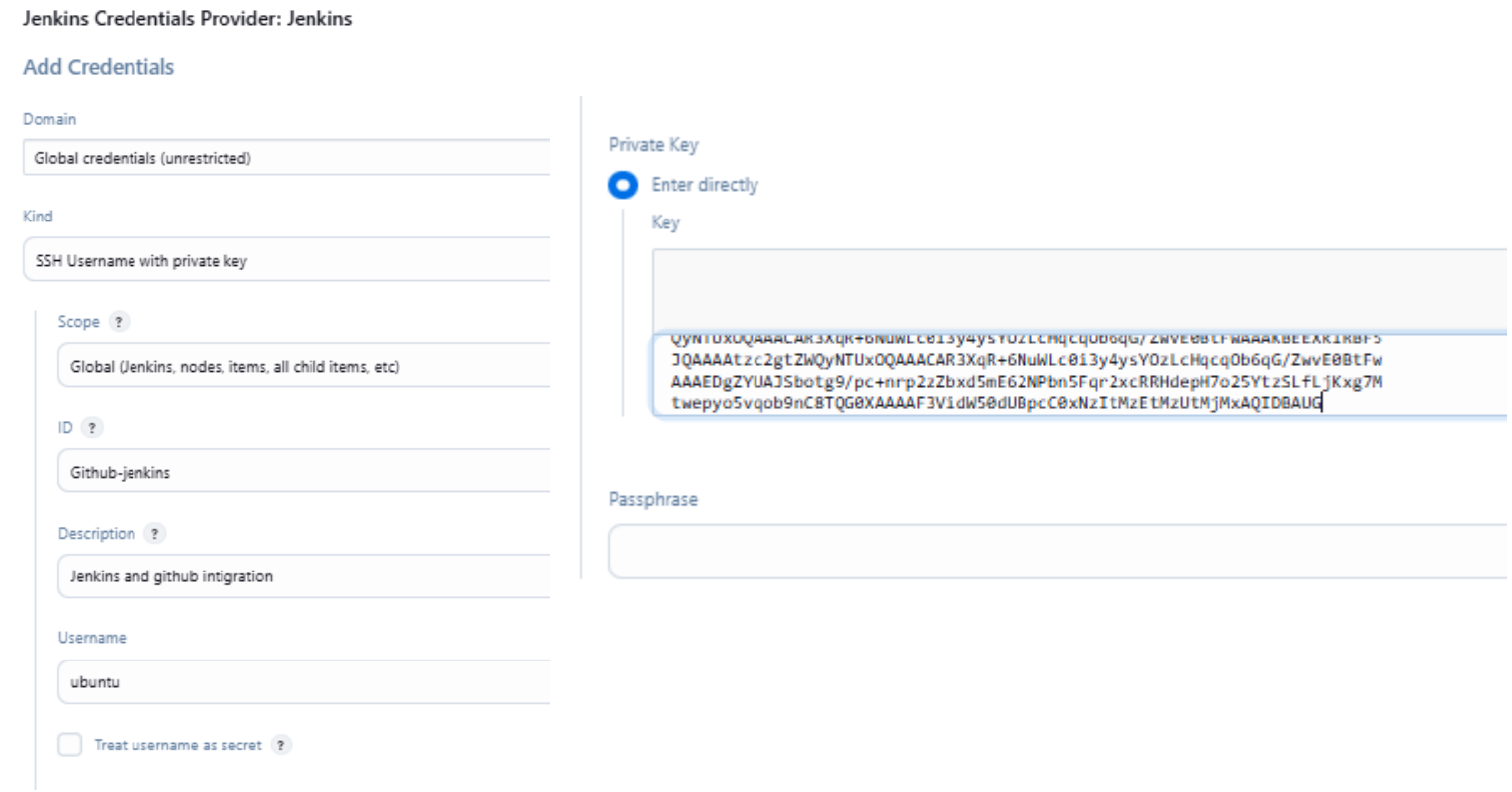
```
ubuntu@ip-172-31-35-231:~$ cd.ssh
cd.ssh: command not found
ubuntu@ip-172-31-35-231:~$ cd .ssh
ubuntu@ip-172-31-35-231:~/.ssh$ ls
authorized_keys  id_ed25519  id_ed25519.pub
ubuntu@ip-172-31-35-231:~/.ssh$ cat id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIBHdepH7o25YtzSLfLjKxg7Mtwepyo5vqob9nC8TQG0X ubuntu@ip-172-31-35-231
ubuntu@ip-172-31-35-231:~/.ssh$ cat id_ed25519
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAAMwAAAAtzc2gtZW
QyNTUxOQAAACAR3XqR+6NuWLc0i3y4ysYOzLcHqcqOb6qG/ZwvE0BtFwAAAKBEEXklRBF5
JQAAAAtzc2gtZWQyNTUxOQAAACAR3XqR+6NuWLc0i3y4ysYOzLcHqcqOb6qG/ZwvE0BtFw
AAAEDgZYUAJSbotg9/pc+nrp2z2bxd5mE62NPbn5Fqr2xcRRHdepH7o25YtzSLfLjKxg7M
twepyo5vqob9nC8TQG0XAAAAF3VidW50dUBpcC0xNzItMzEtMzUtMjMxAQIDBAUG
-----END OPENSSH PRIVATE KEY-----
```

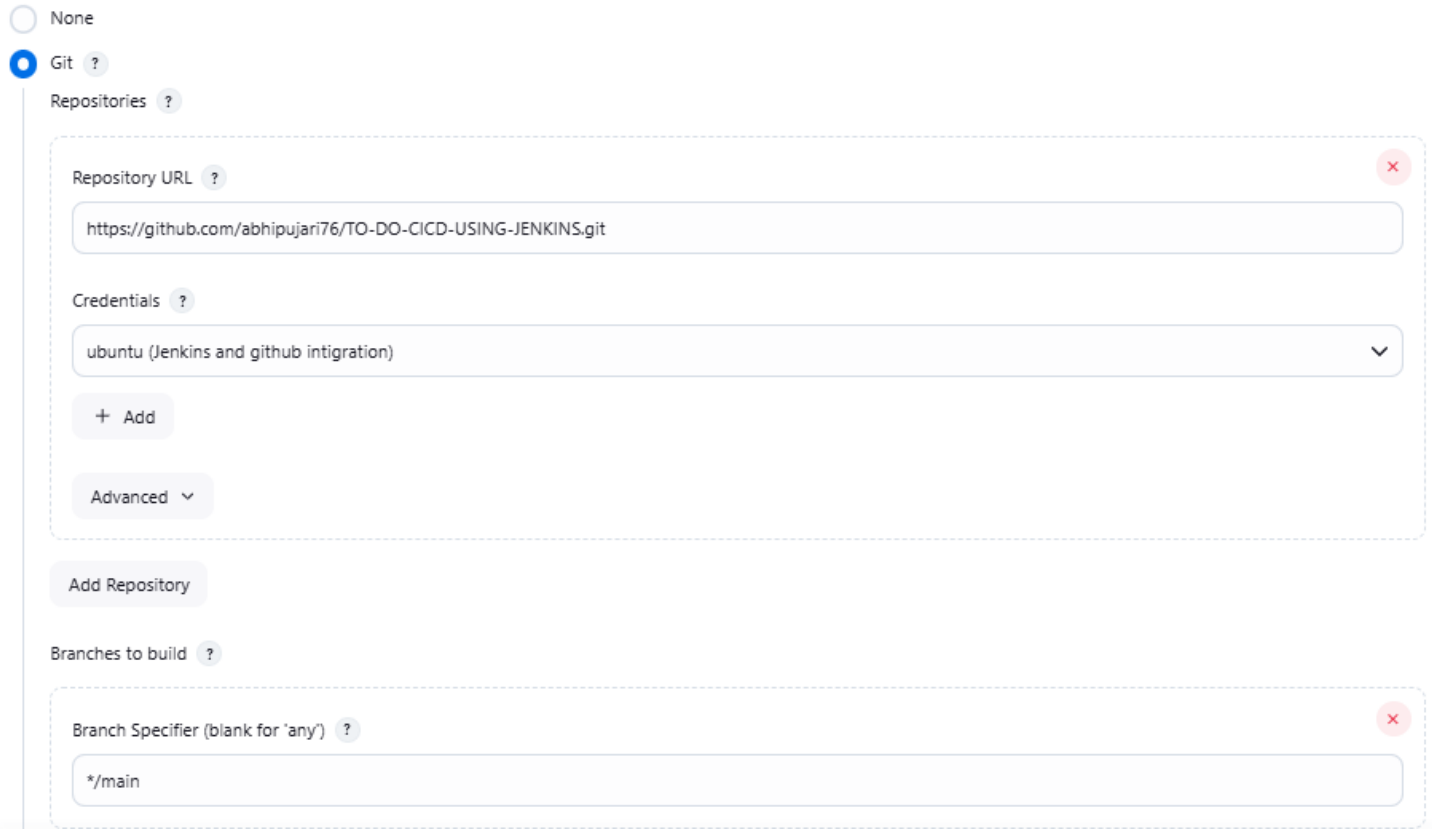
17) To connect Git hub to Jenkins we need to add Public key to the git hub account in the settings .



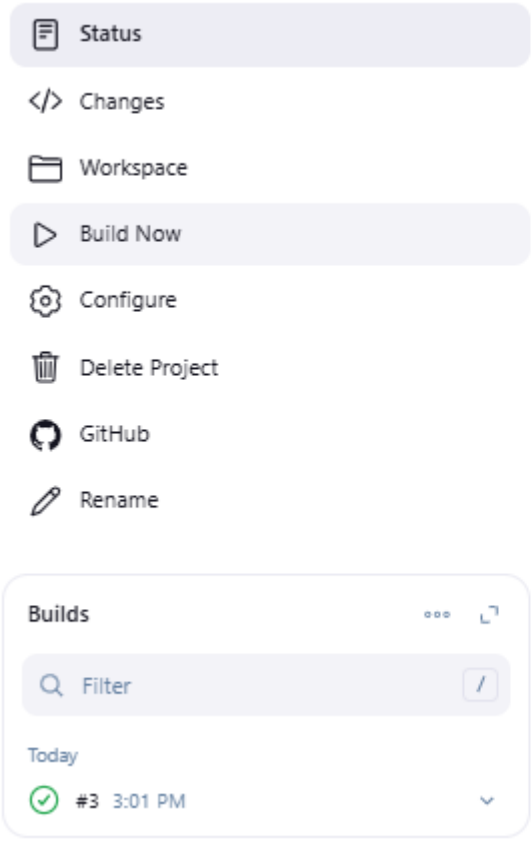


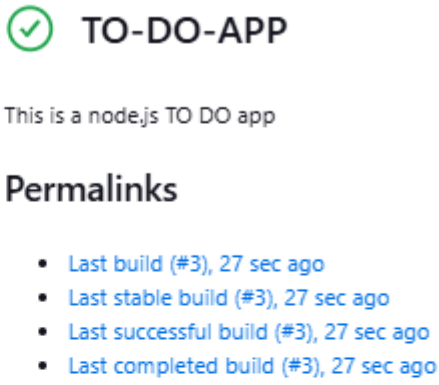
18) Adding Cradentials to the Jenkins



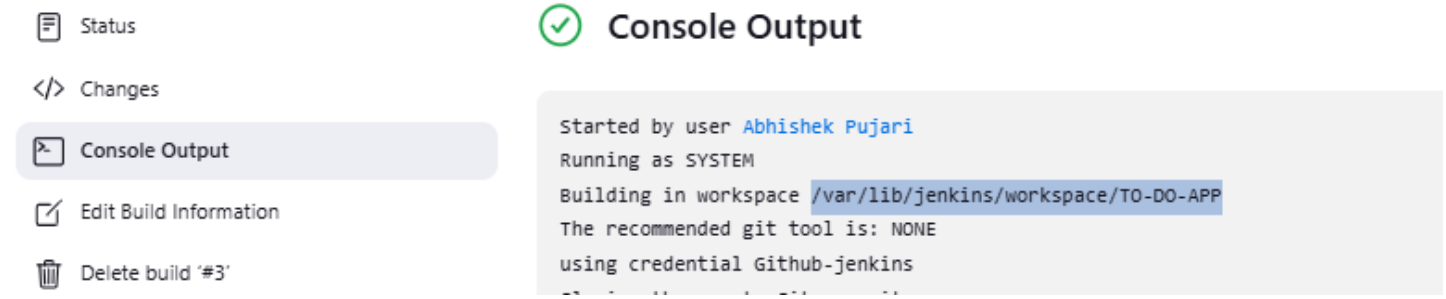


19) Now we have to build it so it can get all the files from the git hub





20) Check if the code has pull from the git hub




```
ubuntu@ip-172-31-35-231:~$ cd
ubuntu@ip-172-31-35-231:~$ cd /var/lib/jenkins/workspace/TO-DO-APP
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$ ls
Dockerfile  Jenkinsfile  app.js  package-lock.json  package.json  test.js  views
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$
```

21) We need to install Nodejs and npm on the server

Sudo apt install nodejs

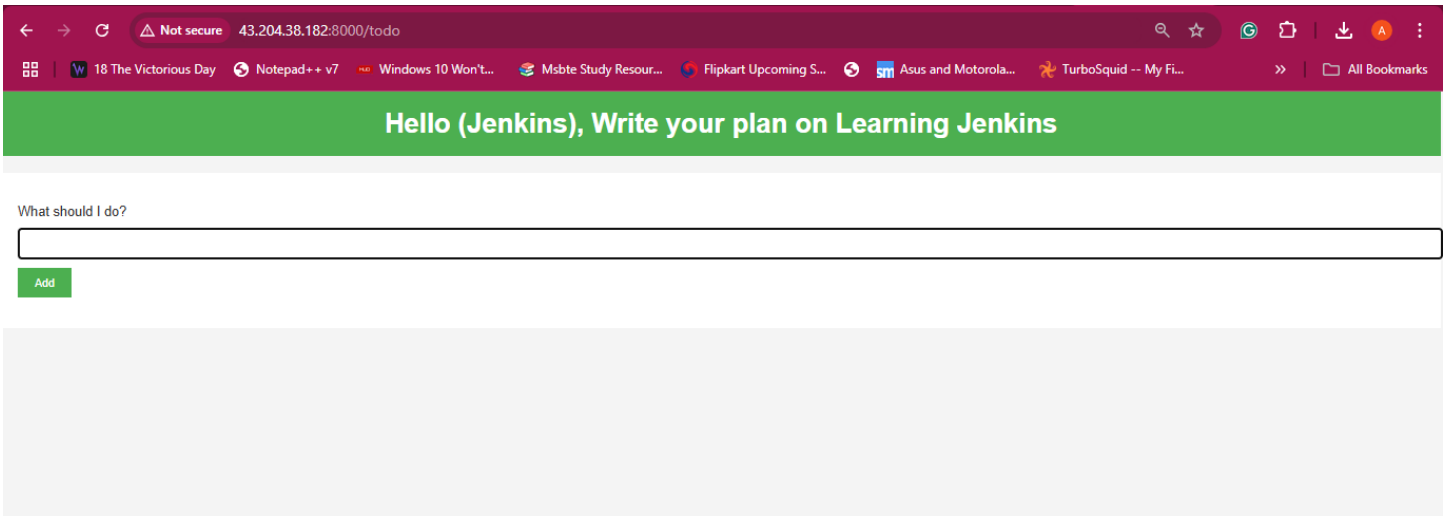
Sudo npm install

```
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$ nodejs -v
v18.19.1
```

22) Now Run the web-app using command

node app.js

```
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$ node app.js
Todolist running on http://0.0.0.0:8000
```



23) Now we have to create a docker for virtualization

Why we do dockerize so that it can run on any environment.

Sudo apt install docker.io

sudo usermod -a -G docker $USER(Give permission to the docker to use docker ps)

```
ubuntu@ip-172-31-35-231:~$ sudo apt install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base pigz
```

24) Creating a Docker file

Create a file using command sudo vim Dockerfile

Node Base Image

FROM node:12.2.0-alpine

#Working Directry

WORKDIR /node

#Copy the Code

COPY . .

#Install the dependecies

RUN npm install

RUN npm run test

EXPOSE 8000

#Run the code

CMD ["node","app.js"]

```
# Node Base Image
FROM node:12.2.0-alpine

#Working Directry
WORKDIR /node

#Copy the Code
COPY . .

#Install the dependecies
RUN npm install
RUN npm run test
EXPOSE 8000

#Run the code
CMD ["node","app.js"]
~
~
~
~
~
~
~
~
~
~
```

25) Build the docker file to make the container

Docker build . –t todo-node-app

```
ubuntu@ip-172-31-35-231:~$ cd  /var/lib/jenkins/workspace/TO-DO-APP
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$ docker build . -t todo-node-app
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/
```

26) Make a container through docker to run the code

sudo docker run -d --name node-todo-app -p 8000:8000 todo-node-app

```
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$ sudo docker run -d --name node-todo-app -p 8000:8000 todo-node-app
67a650ee931f3d2f00881c8b292fc28d2fa75412e661a78df15f9551d788badf
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$ docker ps
CONTAINER ID   IMAGE          COMMAND          CREATED          STATUS          PORTS                                       NAMES
67a650ee931f   todo-node-app  "node app.js"    7 seconds ago    Up 6 seconds    0.0.0.0:8000->8000/tcp, :::8000->8000/tcp   node-todo-app
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$
```

Now the code will be running even if it is stopped in the terminal using docker.

Running code through Jenkins

1) Kill the docker

Docker kill Container_id

```
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$ docker kill 67a650ee93
67a650ee93
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$
```

2) Go to Jenkins and select the job go to configuration and go to build steps

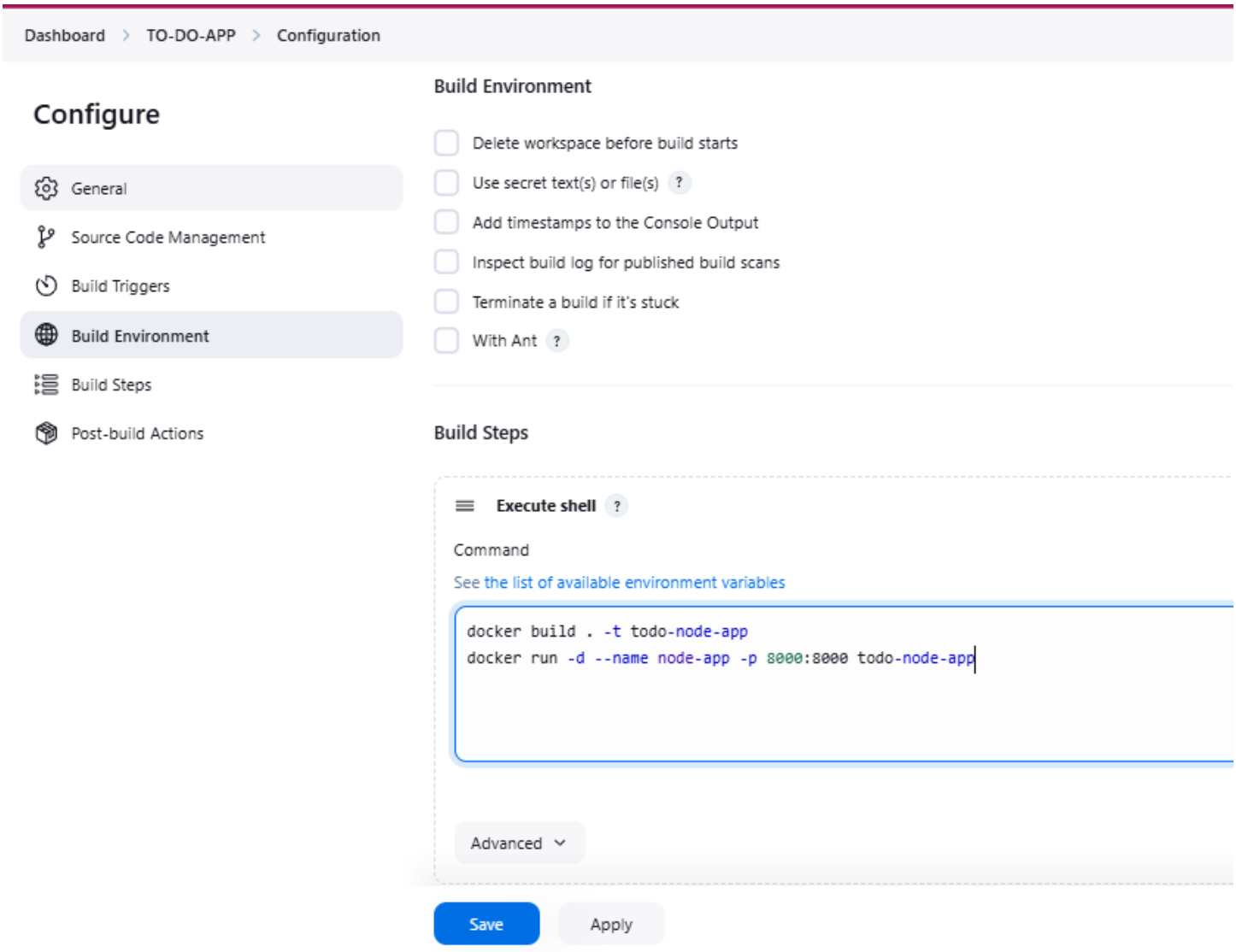
Go in the Build Environment

Click on build steps using shell

Do the steps which use to create the container and run the container.

docker build . -t todo-node-app

docker run -d --name node-app -p 8000:8000 todo-node-app



Important Note : Before running the build in the Jenkins give permission to the Jenkins

sudo chmod 777 /var/lib/jenkins/workspace/TO-DO-APP

sudo usermod -a -G docker jenkins

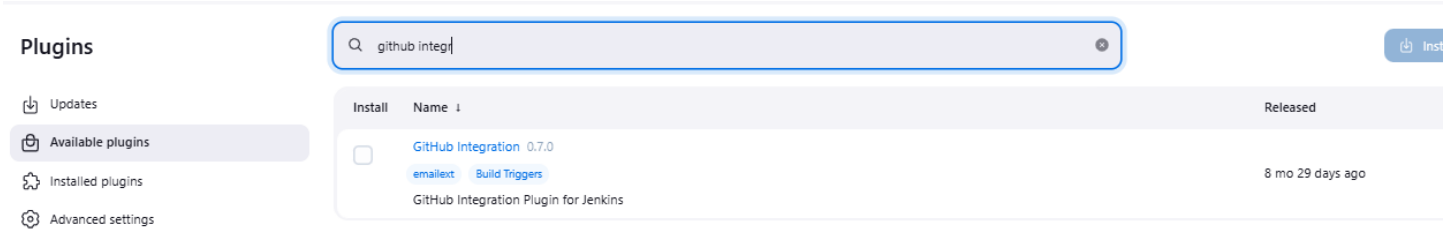
```
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$ sudo chmod 777 /var/lib/jenkins/workspace/TO-DO-APP
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$
```

Now click on the Build button

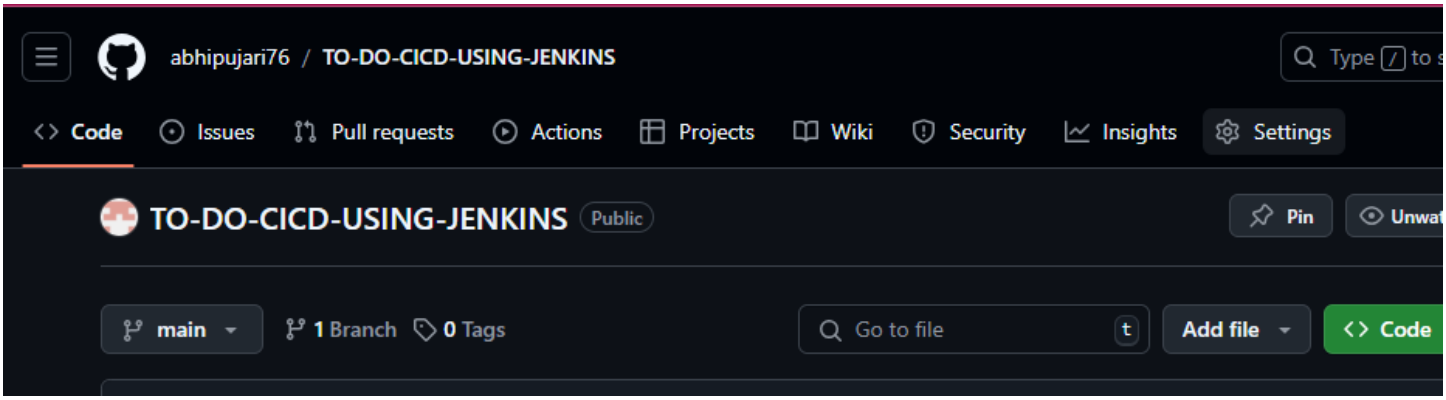
```
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$ sudo usermod -a -G docker jenkins
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$ sudo systemctl restart jenkins
^C
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$ sudo systemctl restart jenkins
ubuntu@ip-172-31-35-231:/var/lib/jenkins/workspace/TO-DO-APP$ 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 Fri 2024-11-15 19:02:22 UTC; 7s ago
   Main PID: 2701 (java)
      Tasks: 46 (limit: 1130)
```

To auto-mate all the process of CICD we will be using WEBHOOKS which will be in the middle of git hub and Jenkins

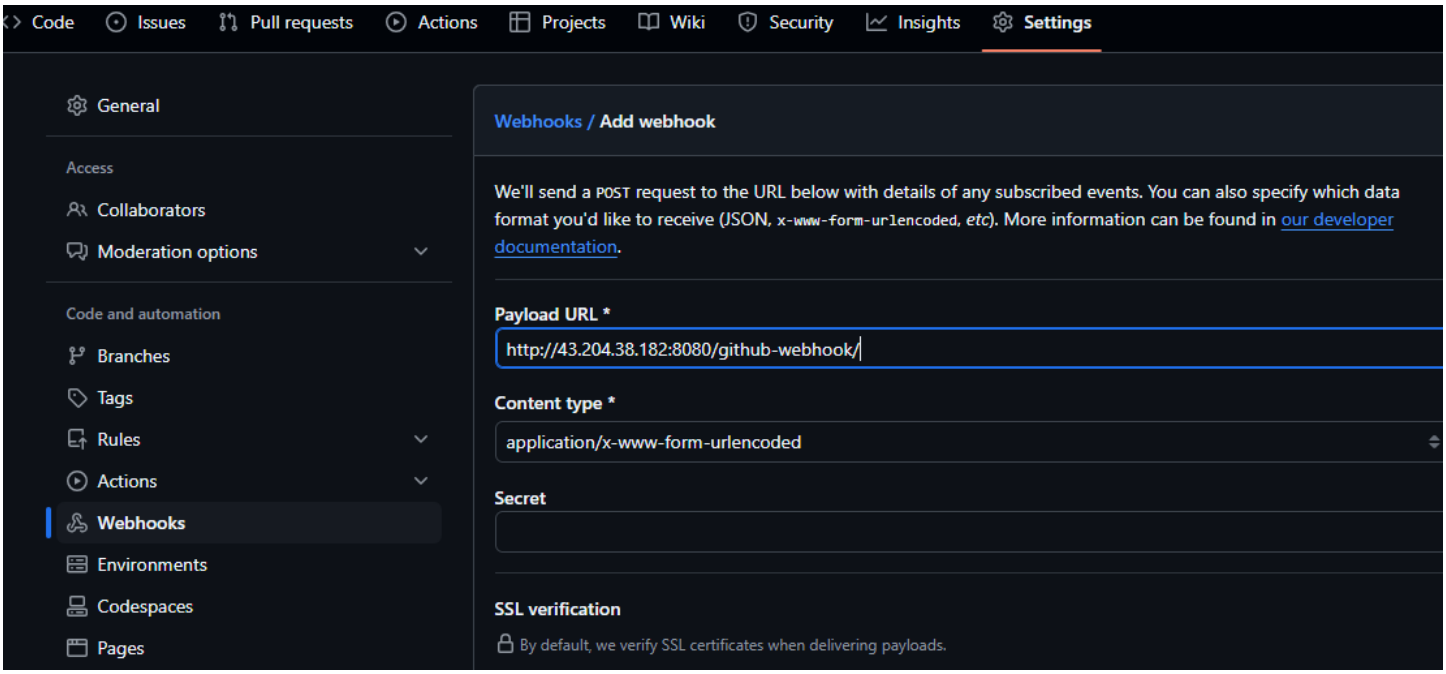
1) Install a plugin call github Integration



2) Now we have to create a webhook in the GitHub through the repo setting

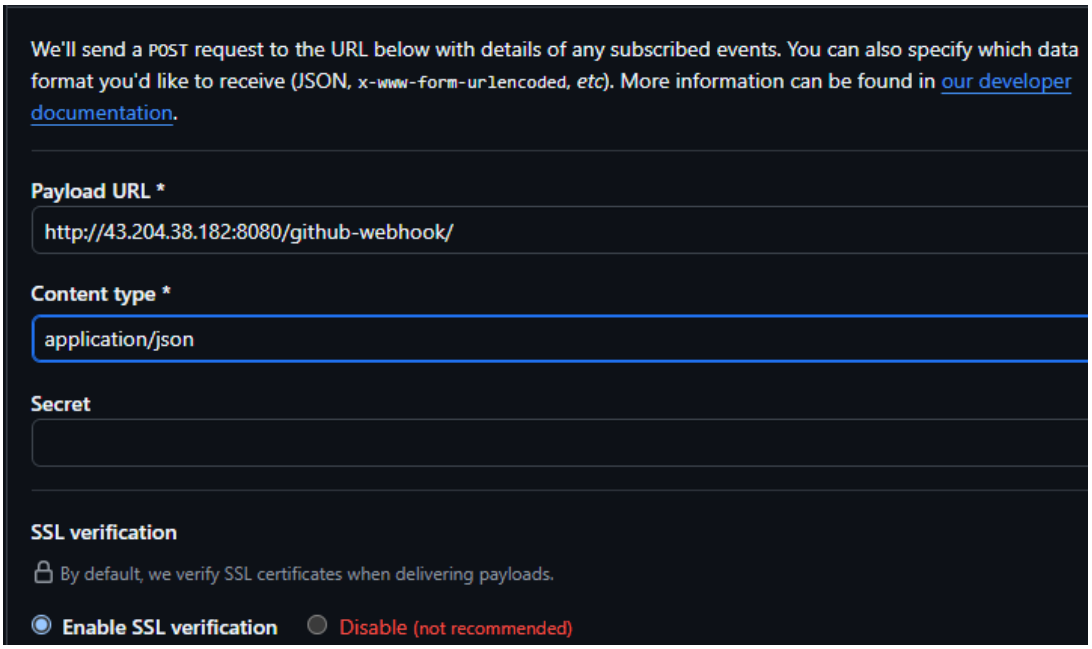


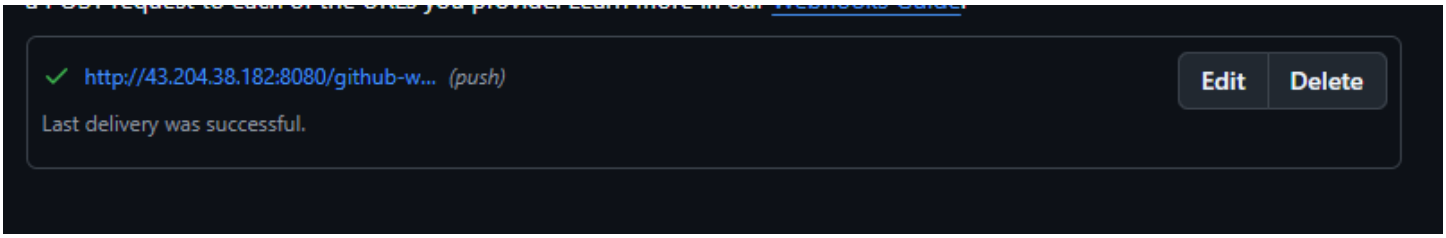
Add your Jenkins payload URL in the github Webhook

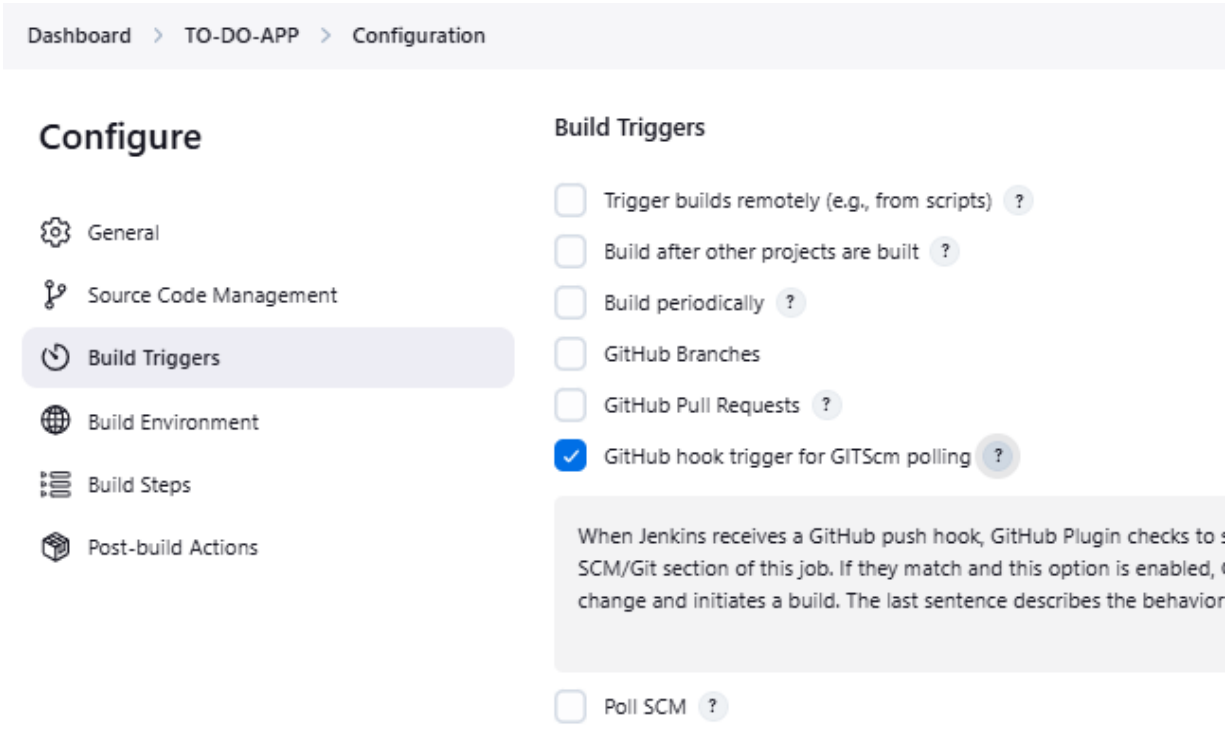


Important : Make Changes for instance security group where Jenkins is for MyIP to anywhere so that github can access it from any where.

Make content type to application/json and then add webhook







3) When you make some Changes in the Github it will automatic build trigger and make the build