Jenkins Assignment

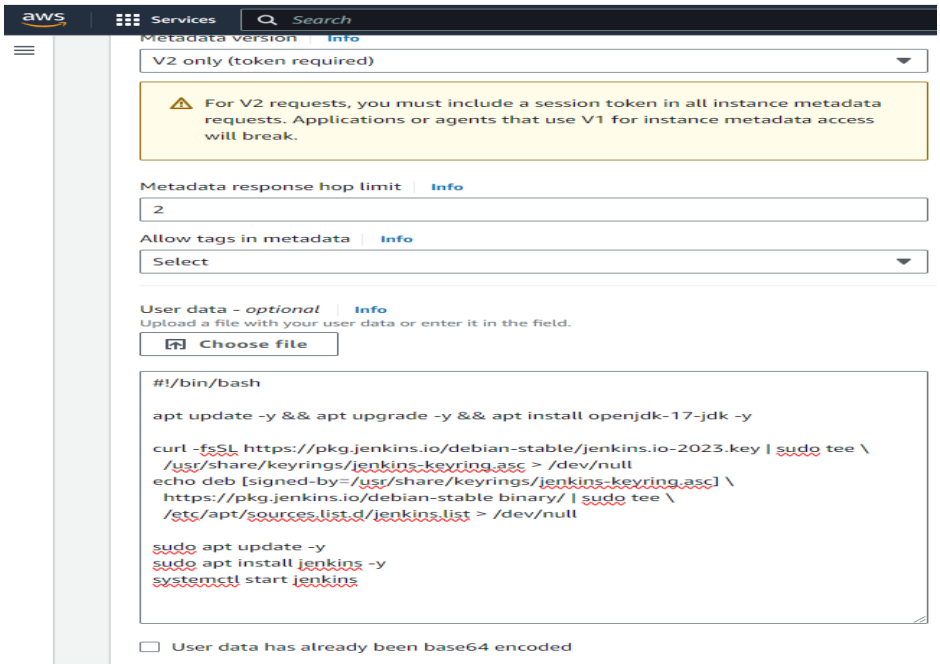
1) **How to change default port of Jenkins (8080) to some other port.**

   Default port for Jenkins [8080]

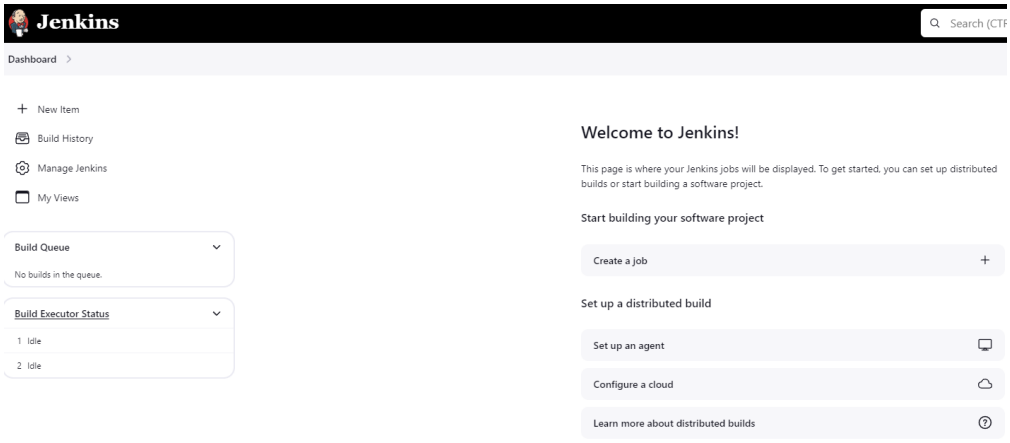
   New Port for Jenkins [8081]

- To change the default port of Jenkins, we need to have a Jenkins Server EC2 instance.

  Add the below commands in user data section of EC2 instance before launching it to install Jenkins on it



- Once the EC2 instance is up, Post deployment steps are performed for configuring Jenkins to setup the Jenkins server. Once Jenkins is configured, we see the below dashboard screen.



- Now we will go to the terminal and do the below changes:

  vi /etc/default/jenkins (Edit the jenkins configuration file)

```
root@ip-172-31-94-44:/# vi /etc/default/jenkins
```

- Scroll down until you find the following lines:

  # port for HTTP connector (default 8080; disable with -1)
  HTTP_PORT=8080

```
# port for HTTP connector (default 8080; disable with -1)
HTTP_PORT=8080
```

- Edit the second line to include the port number you want to specify)

  # port for HTTP connector (default 8080; disable with -1)
  HTTP_PORT=8081

```
# port for HTTP connector (default 8080; disable with -1)
HTTP_PORT=8081
```

- Esc, then: wq! (Save and exit the file)

- Now we will edit the jenkins service file with new port number

vi usr/lib/systemd/system/jenkins.service (Open Jenkins Service)

```
root@ip-172-31-94-44:/#
root@ip-172-31-94-44:/# vi usr/lib/systemd/system/jenkins.service
```

- Scroll down until you find the below line:

Environment="JENKINS_PORT=8080"

```
# directive below.
Environment="JENKINS_PORT=8080"
```

- Change the port with the new port number

Environment="JENKINS_PORT=8081"

```
# directive below.
Environment="JENKINS_PORT=8081"
```

Esc, then: wq! (Save and exit the file)

- Reload the Daemon

systemctl daemon-reload

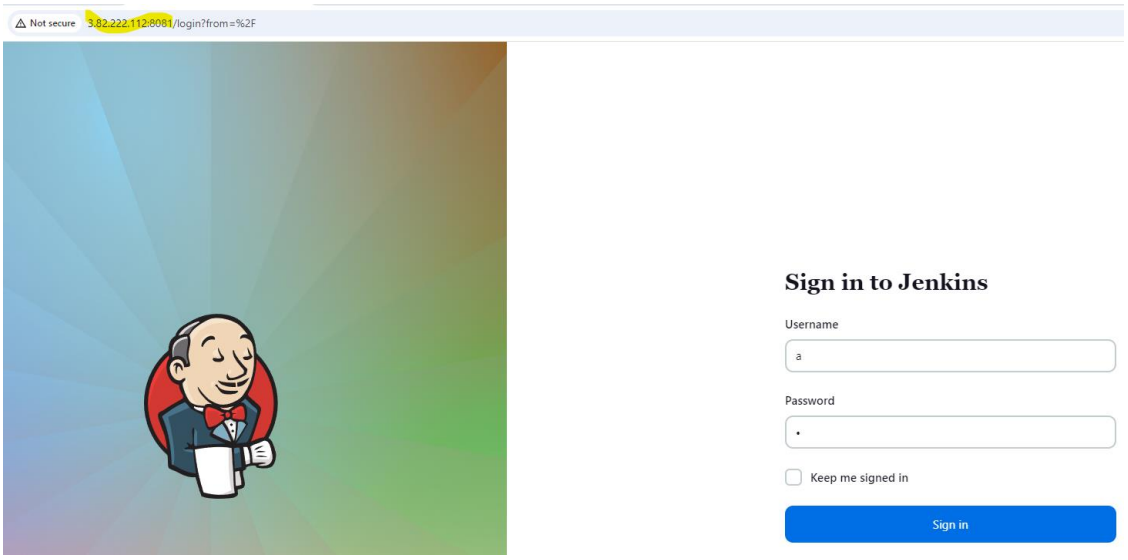
```
root@ip-172-31-94-44:/# systemctl daemon-reload
root@ip-172-31-94-44:/#
```

- Restart Jenkins Service

systemctl restart jenkins

```
root@ip-172-31-94-44:/# systemctl restart jenkins
root@ip-172-31-94-44:/#
```

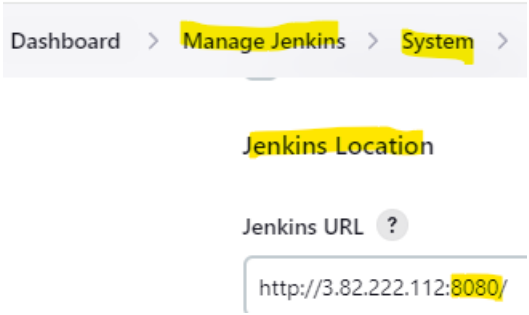
- Verify on the Webpage with Public IP: New port Number



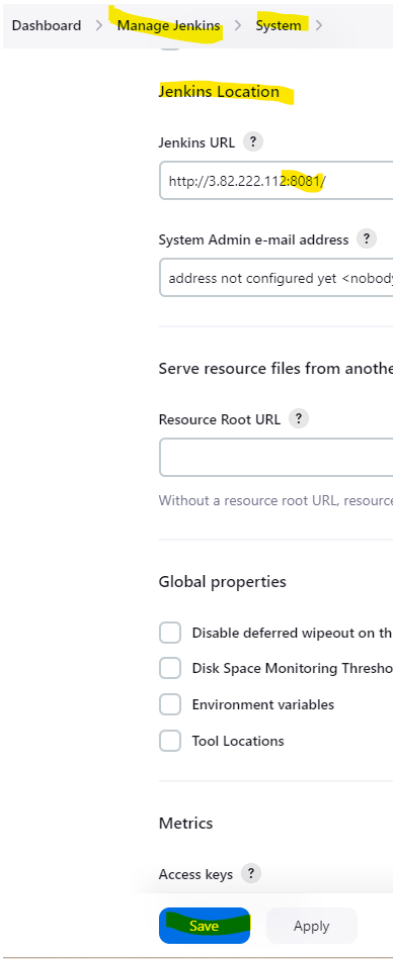
- Now to make sure all the functionalities of Jenkins are working properly. Need make change to Jenkins URL to reflect the new port Number

  Manage Jenkins > System > Jenkins Location > Jenkins URL

  Before making the change.



  change from URL with old port Number to URL with new port number and hit save



- Restart Jenkins Service

  systemctl restart jenkins

```
root@ip-172-31-94-44:/# systemctl restart jenkins
root@ip-172-31-94-44:/#
```

- The port change is completed successfully.

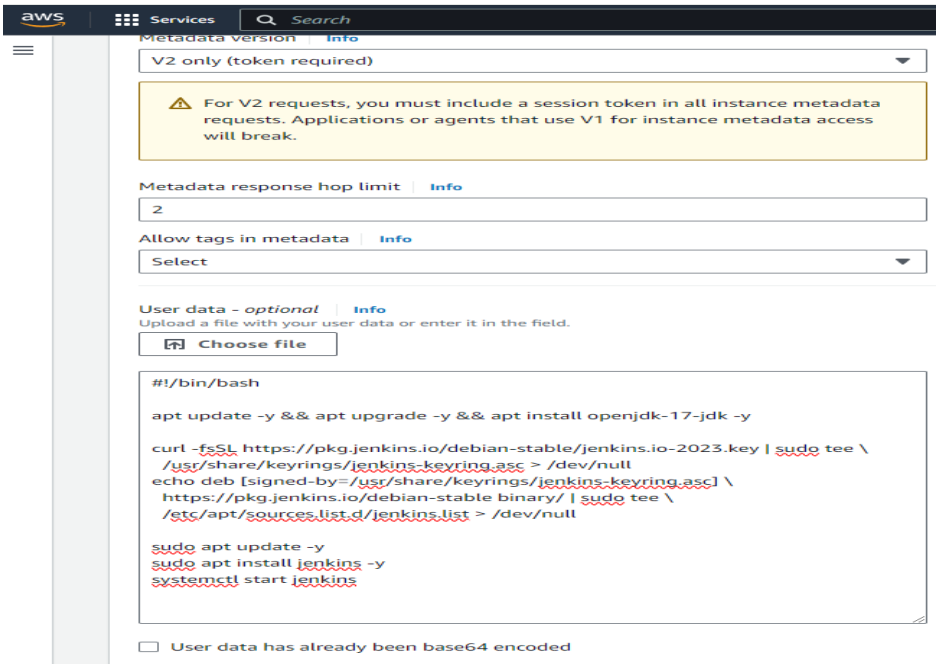
2) How to change Home Directory of Jenkins.

Default home directory path [/var/lib/jenkins]

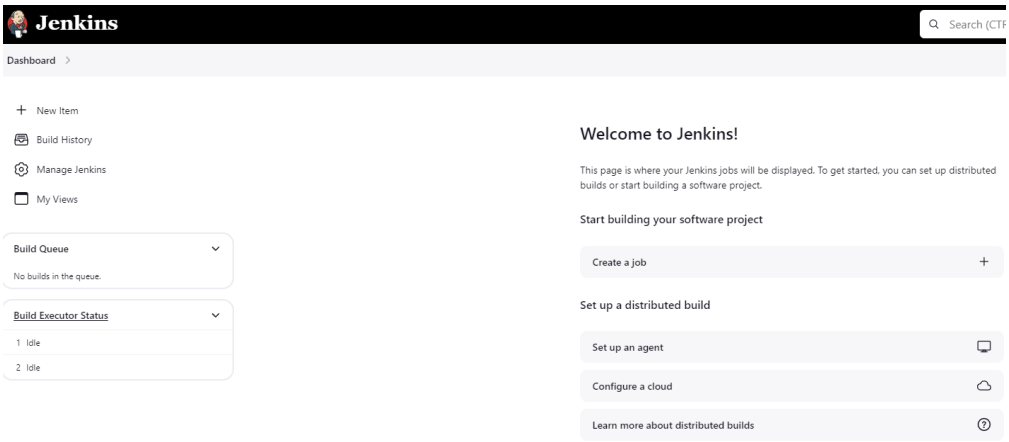
New Path we want to give is [/home/jenkins_home/jenkins]

- To change the default home directory of Jenkins, we need to have a Jenkins Server EC2 instance.

  Add the below commands in user data section of EC2 instance before launching it to install Jenkins on it



- Once the EC2 instance is up, Post deployment steps are performed for configuring Jenkins to setup the Jenkins server. Once Jenkins is configured, we see the below dashboard screen.



- Now we will go to the terminal and do the below changes:

- Stop Jenkins Service

  systemctl stop jenkins

```
root@ip-172-31-94-44:/# systemctl stop jenkins
root@ip-172-31-94-44:/#
```

- Create a new Jenkins Home directory using the mkdir command

  mkdir /home/jenkins_home

```
root@ip-172-31-94-44:/# mkdir /home/jenkins_home
root@ip-172-31-94-44:/#
```

- Change permissions for the new home directory

  chown jenkins:jenkins /home/jenkins_home

```
root@ip-172-31-94-44:/# chown jenkins:jenkins /home/jenkins_home
root@ip-172-31-94-44:/#
```

- Copy the contents from the old Jenkins Home directory to the new one

  cp -prv /var/lib/jenkins /home/jenkins_home

```
root@ip-172-31-94-44:/# cp -prv /var/lib/jenkins /home/jenkins_home
'/var/lib/jenkins' -> '/home/jenkins_home/jenkins'
'/var/lib/jenkins/.java' -> '/home/jenkins_home/jenkins/.java'
'/var/lib/jenkins/.java/fonts' -> '/home/jenkins_home/jenkins/.java/fonts'
'/var/lib/jenkins/.java/fonts/17.0.12' -> '/home/jenkins_home/jenkins/.java/fonts/17.0.12'
'/var/lib/jenkins/.java/fonts/17.0.12/fcinfo-1-ip-172-31-94-44-Ubuntu-24.04-en-.properties' -> '/home/jenkins_home/jen
kins/.java/fonts/17.0.12/fcinfo-1-ip-172-31-94-44-Ubuntu-24.04-en-.properties'
'/var/lib/jenkins/secret.key' -> '/home/jenkins_home/jenkins/secret.key'
'/var/lib/jenkins/secret.key.not-so-secret' -> '/home/jenkins_home/jenkins/secret.key.not-so-secret'
'/var/lib/jenkins/plugins' -> '/home/jenkins_home/jenkins/plugins'
'/var/lib/jenkins/plugins/ionicons-api.jpi' -> '/home/jenkins_home/jenkins/plugins/ionicons-api.jpi'
'/var/lib/jenkins/plugins/ionicons-api' -> '/home/jenkins_home/jenkins/plugins/ionicons-api'
'/var/lib/jenkins/plugins/ionicons-api/META-INF' -> '/home/jenkins_home/jenkins/plugins/ionicons-api/META-INF'
'/var/lib/jenkins/plugins/ionicons-api/META-INF/MANIFEST.MF' -> '/home/jenkins_home/jenkins/plugins/ionicons-api/META-
INF/MANIFEST.MF'
'/var/lib/jenkins/plugins/ionicons-api/META-INF/maven' -> '/home/jenkins_home/jenkins/plugins/ionicons-api/META-INF/ma
ven'
```

- Assign Jenkins as the user for the new home directory

  usermod -d /home/jenkins_home jenkins

```
root@ip-172-31-94-44:/# usermod -d /home/jenkins_home jenkins
root@ip-172-31-94-44:/#
```

- Edit the Jenkins configuration file.

  vi /etc/default/jenkins

```
root@ip-172-31-94-44:/# vi /etc/default/jenkins
```

- Scroll down until you reach the JENKINS_HOME entry

  Edit the line to include the path to the new home directory

  In our Case

  NAME=jenkins

```
# pulled in from
NAME=jenkins
```

- Old home directory path

  JENKINS_HOME=/var/lib/$NAME

```
# jenkins home location
JENKINS_HOME=/var/lib/$NAME
```

- New home directory path

  JENKINS_HOME=/home/jenkins_home/$NAME

```
# jenkins home location
JENKINS_HOME=/home/jenkins_home/$NAME
```

  Esc, then: wq! (Save and exit the file)

- Now we will edit the jenkins service file with new port number

  vi usr/lib/systemd/system/jenkins.service

```
root@ip-172-31-94-44:/#
root@ip-172-31-94-44:/# vi usr/lib/systemd/system/jenkins.service
```

- Scroll down until you reach the Environment and Working Directory

  Edit the line to include the path to the new home directory

  In our Case

- OLD Entry of Jenkins home path

  Environment="JENKINS_HOME=/var/lib/jenkins"
  WorkingDirectory=/var/lib/jenkins

```
# Directory where Jenkins stores its configuration and workspaces
Environment="JENKINS_HOME=/var/lib/jenkins"
WorkingDirectory=/var/lib/jenkins
```

- NEW Entry of Jenkins home path

  Environment="JENKINS_HOME=/home/jenkins_home/jenkins”
  WorkingDirectory=/home/jenkins_home/jenkins

  Esc, then: wq! (Save and exit the file)

- Reload the Daemon

  systemctl daemon-reload

```
root@ip-172-31-94-44:/# systemctl daemon-reload
root@ip-172-31-94-44:/#
```

- Rename the old Jenkins home directory

  mv /var/lib/jenkins /var/lib/jenkins.old

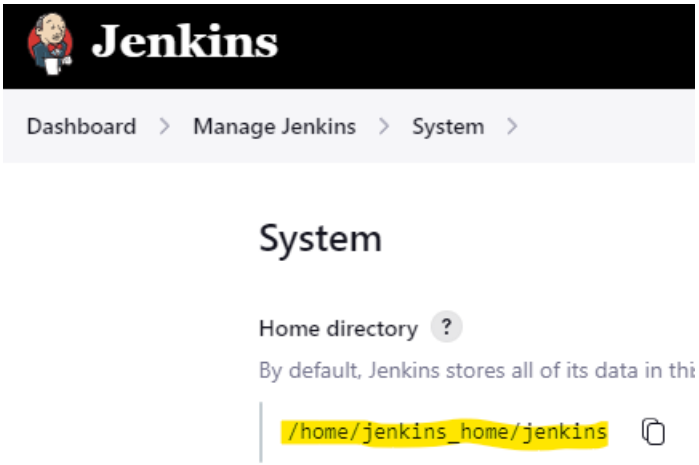
```
root@ip-172-31-94-44:/# mv /var/lib/jenkins /var/lib/jenkins.old
root@ip-172-31-94-44:/#
```

- Restart Jenkins Service

  systemctl restart jenkins

```
root@ip-172-31-94-44:/# systemctl restart jenkins
root@ip-172-31-94-44:/#
```

- Verify the New Home directory under > Manage Jenkins > System > Home directory.

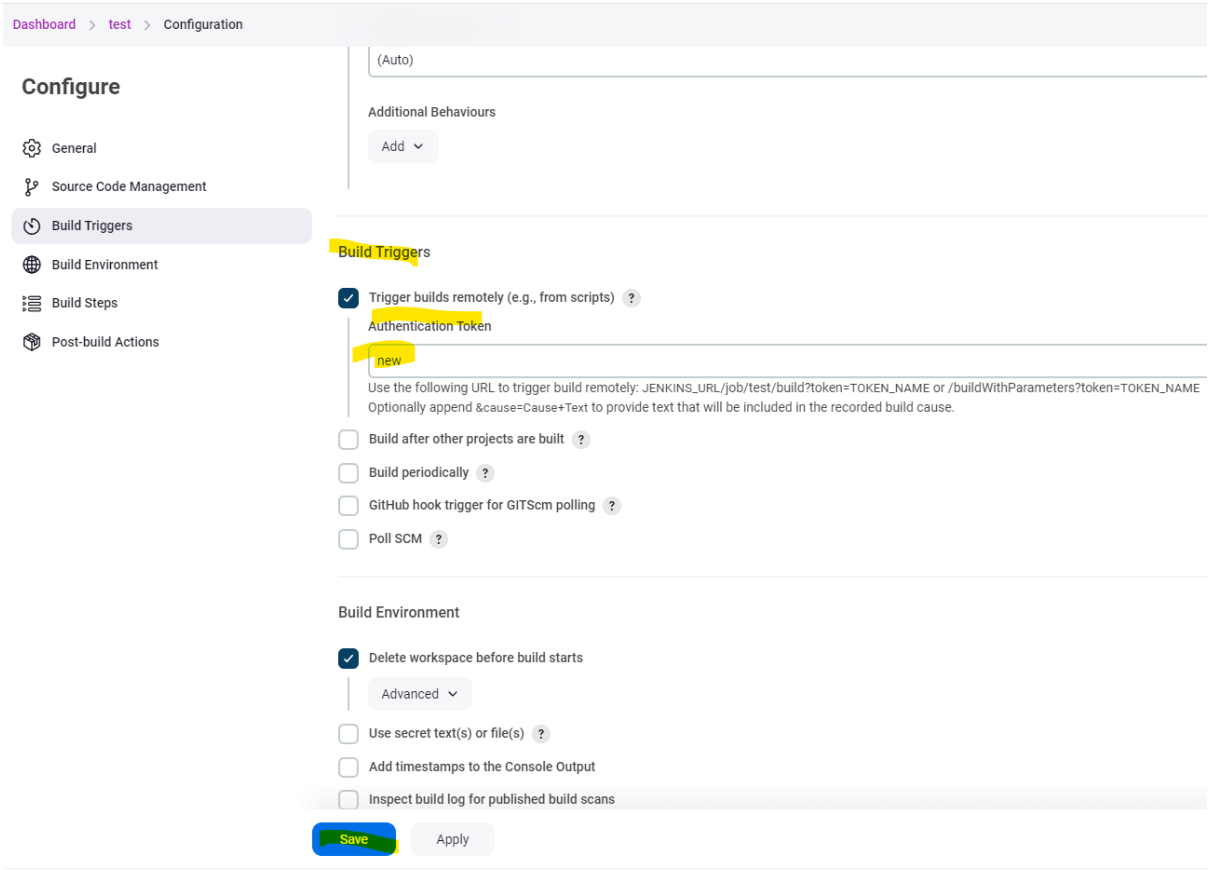


- The Home Directory path has been changed successfully.

3) How to run Remote Job in Jenkins.

We can trigger a job remotely through a Web URL in Jenkins also.

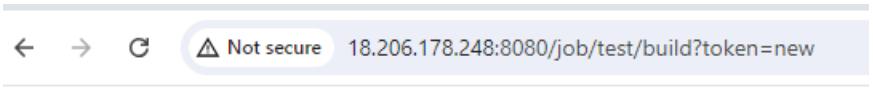
We will follow the below steps to achieve the same.
While configuring the job, we need to select Trigger builds remotely under Build Triggers, then we can give any authentication token eg new and Click Save.



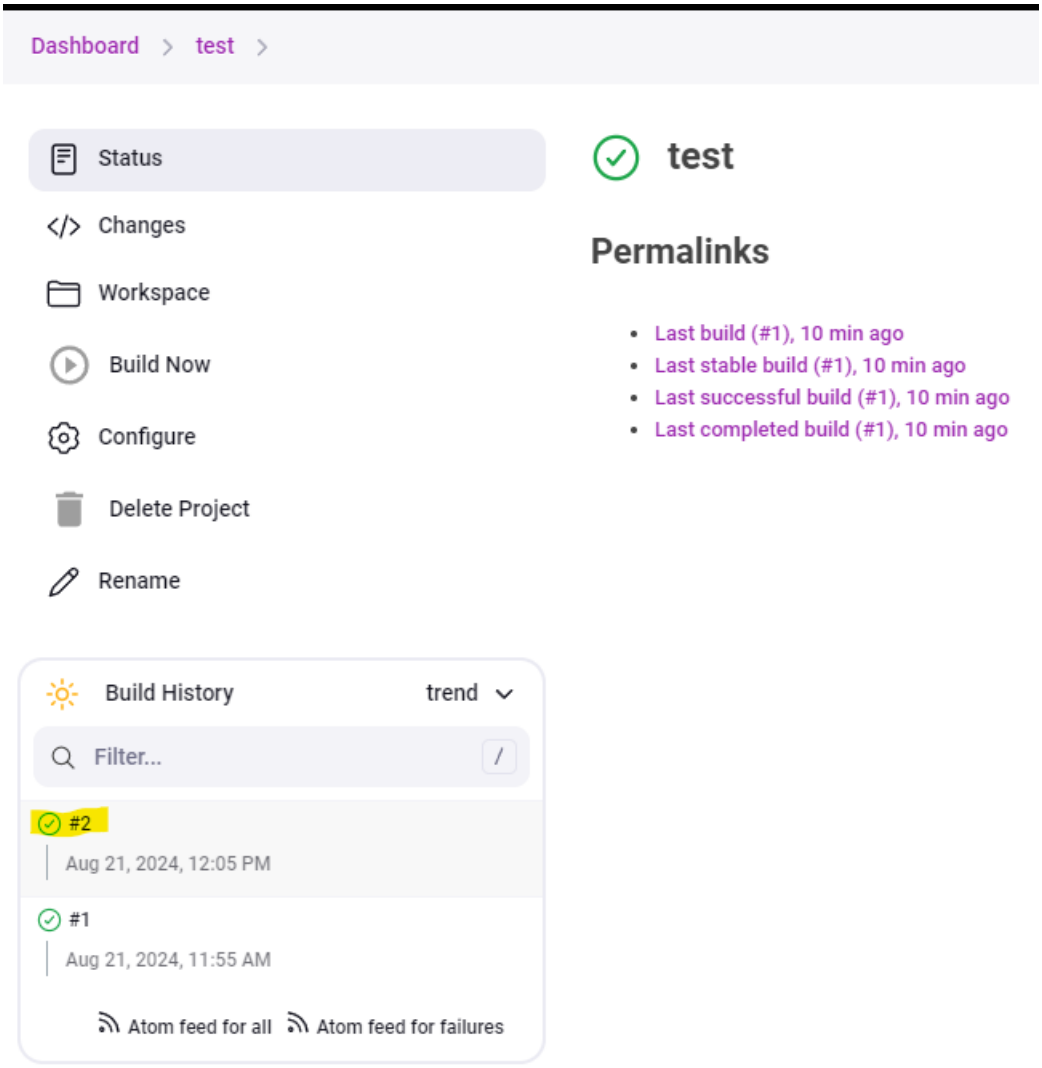
To trigger the build remotely we need use the below URL

JENKINS_URL/job/test/build?token=TOKEN_NAME or /buildWithParameters?token=TOKEN_NAME

Below screenshot is in our case

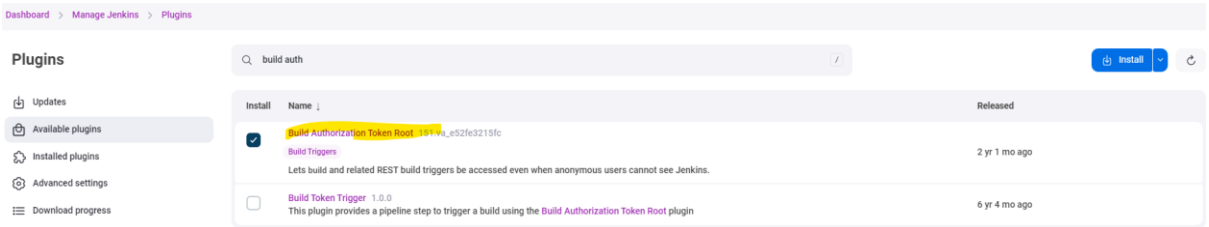


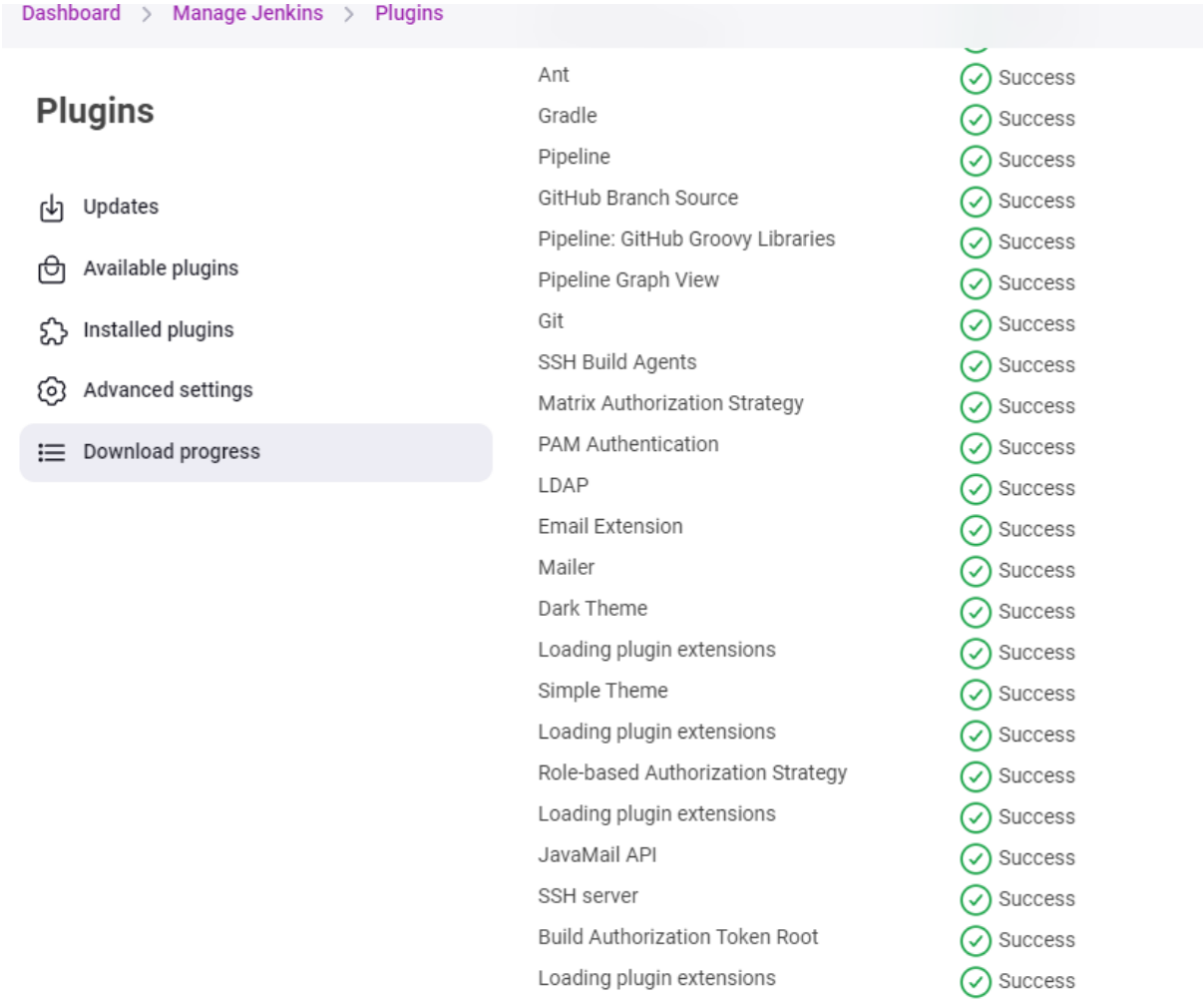
We see the 2nd Build is also executed successfully.



Alternate Way to run the job through remotely is by using a plugin

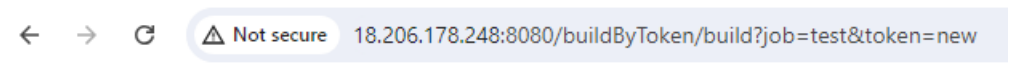
We need to install the plugin Build Authorization Token Root



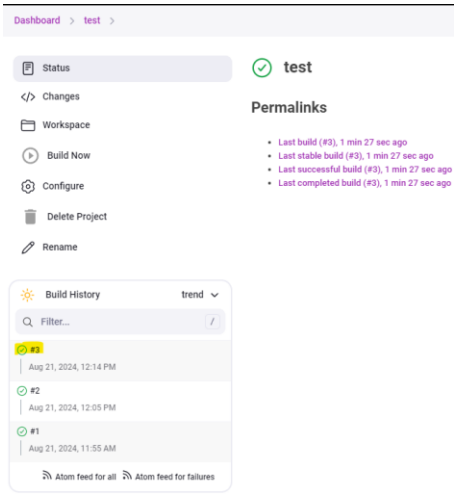


Now we will run the below URL to see if build runs.

<Jenkins_URL>buildByToken/build?job=<Job_Name>&token=<token Name>



See the Build 3 is also completed successfully.



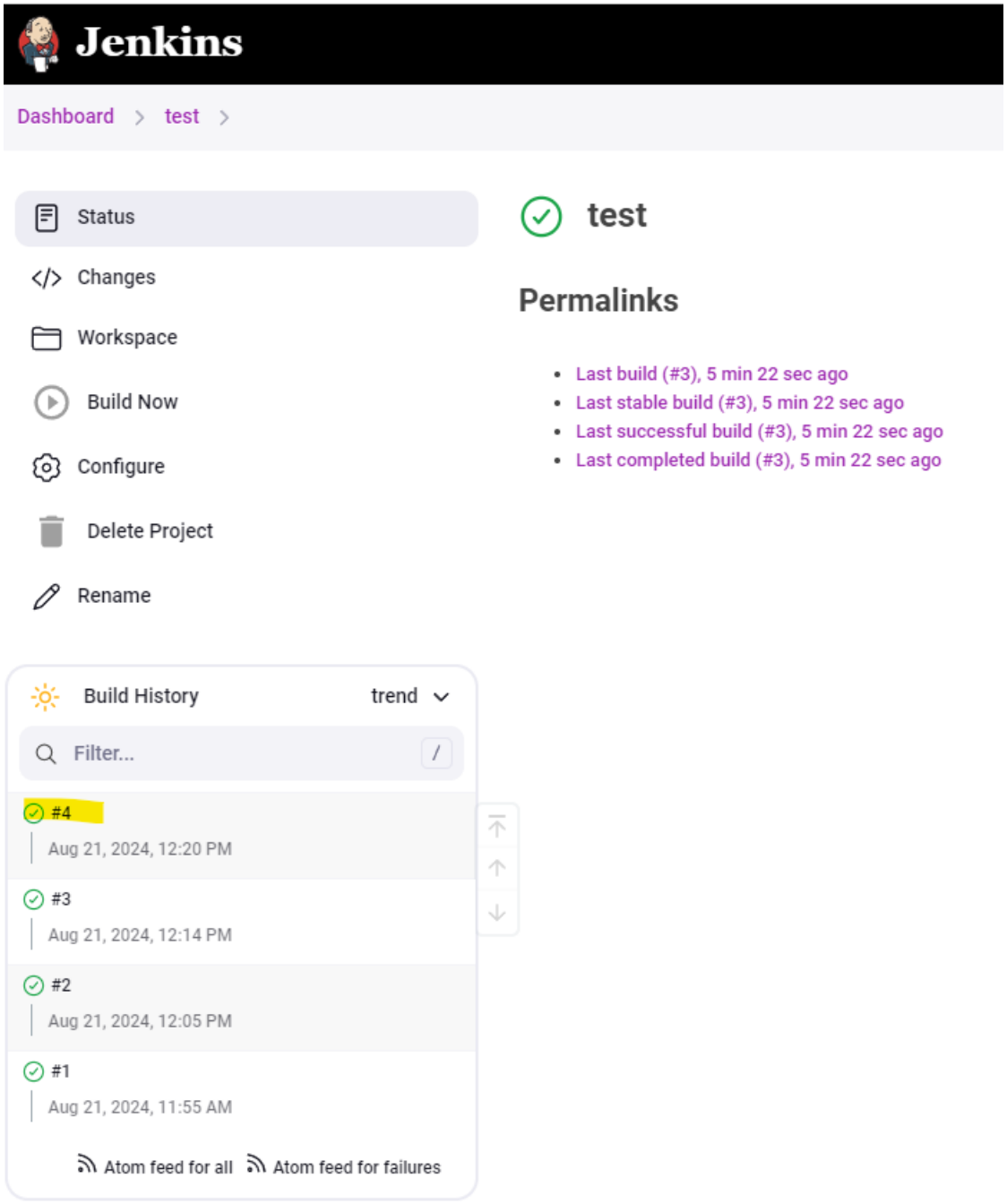
We can also run it on the Ubuntu terminal using the below command.

Just need to add “\” before the & and then run.

<Jenkins_URL>buildByToken/build?job=<Job_Name>\&token=<token Name>

```
root@ip-172-31-33-31:/var/lib/jenkins/workspace/test# curl http://18.206.178.248:8080/buildByToken/build?job=test\&token=new
root@ip-172-31-33-31:/var/lib/jenkins/workspace/test#
```

Build 4 is completed successfully

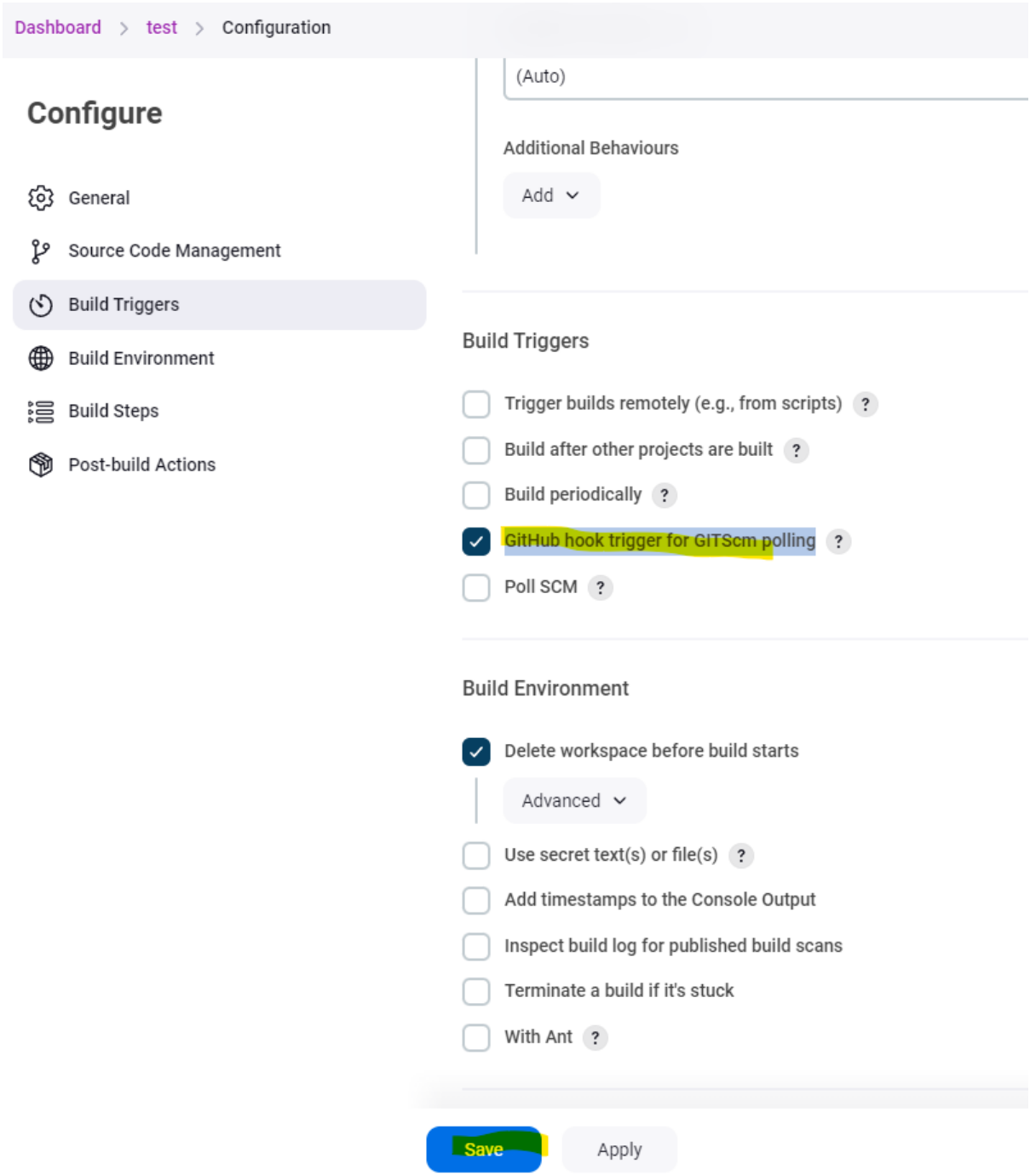


4) How to create GitHub hook trigger for GITScm polling through Jenkins.

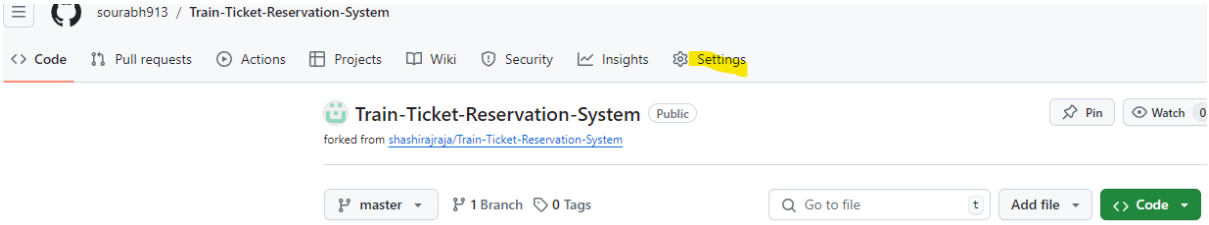
This is used when a job needs to be triggered when there is any change in the GitHub repo.

Steps to followed are below:

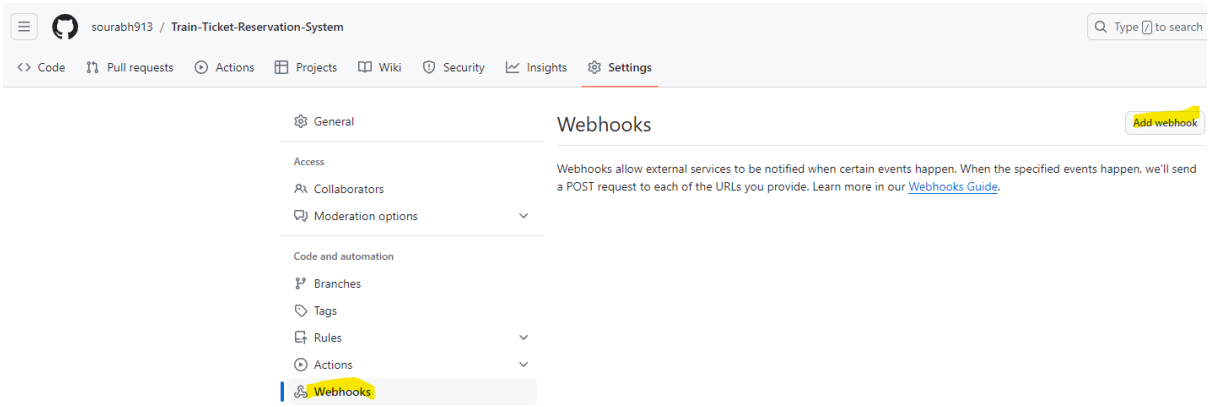
While configuring the job, Check the option of GitHub hook trigger for GITScm polling under Build Triggers and then click Save.



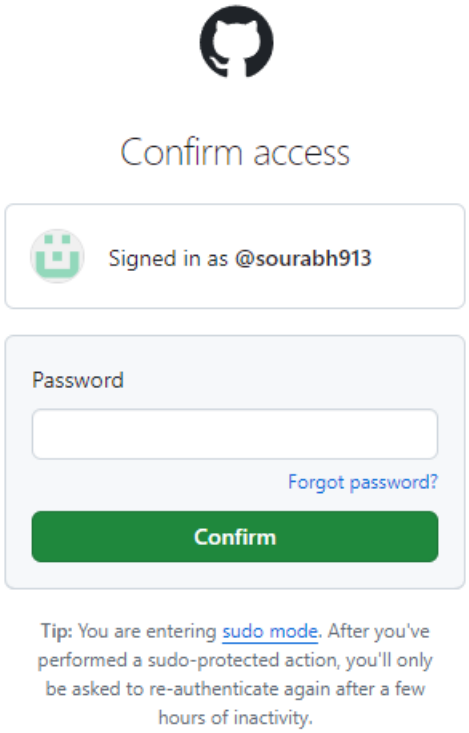
Then on the GitHub side, Go to Settings



Select Webhooks > Add webhook



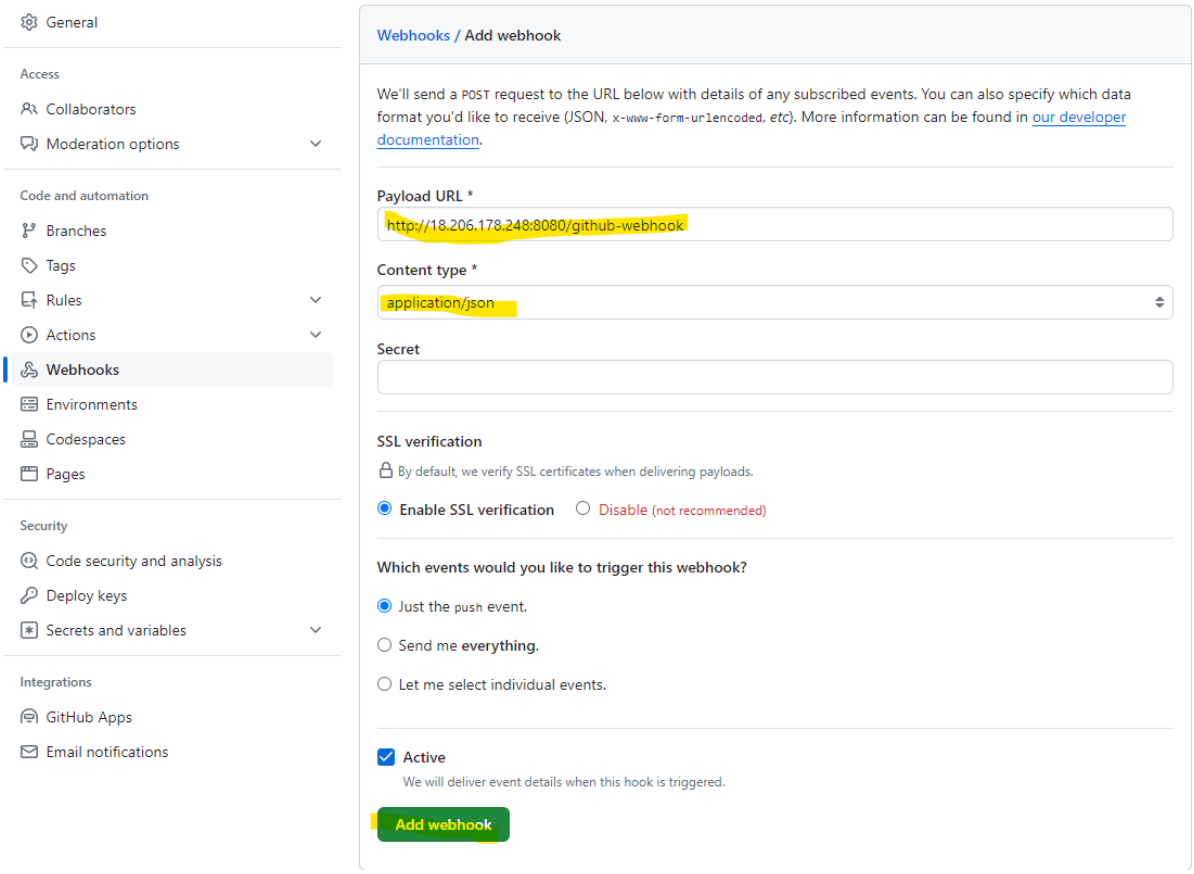
Need to put in password for your GitHub account.

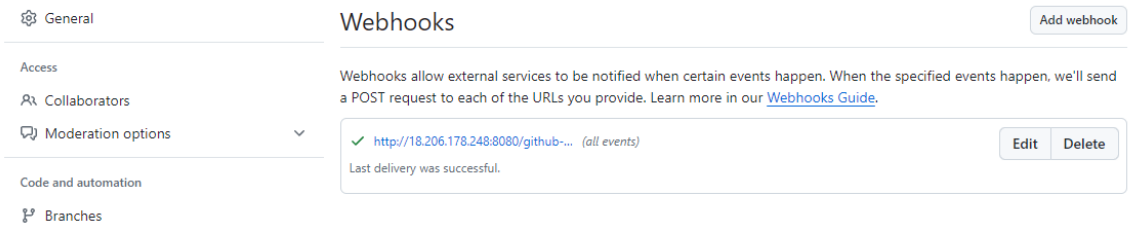


Then under the Payload URL, give the Jenkins URL /github webhook

And under Content type > Select application/json.

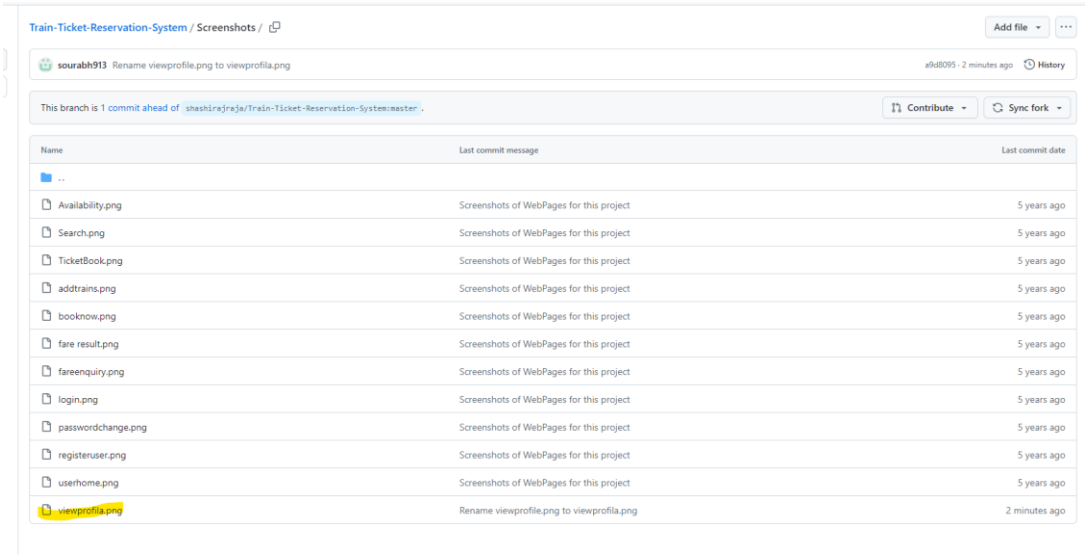
Click Add Webhook

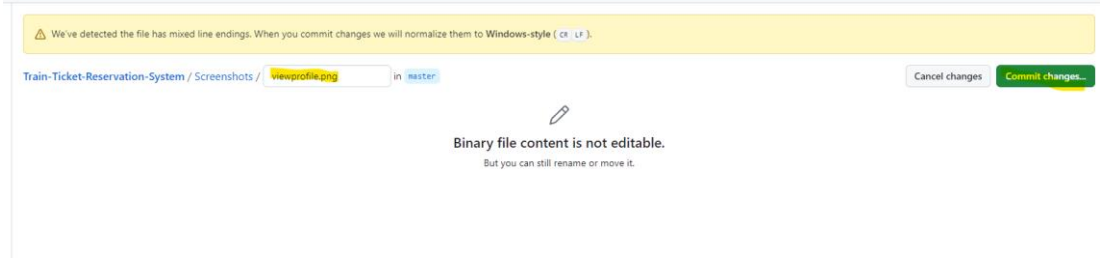


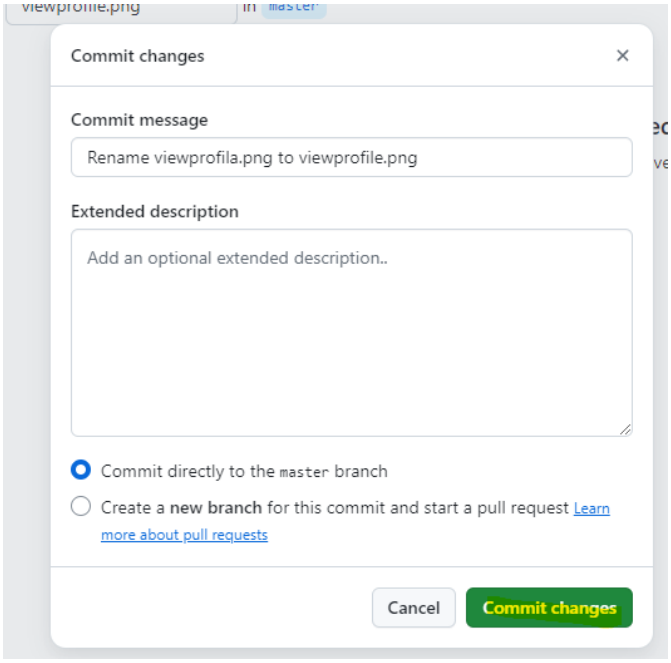


For Example, in my case I made a change under my repo > screenshots > rename viewprofila.png to viewprofile.png

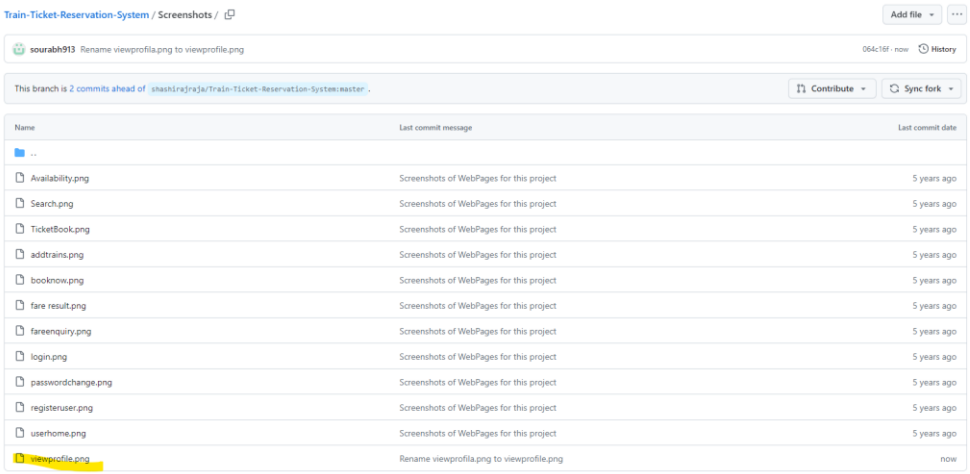
Before Change



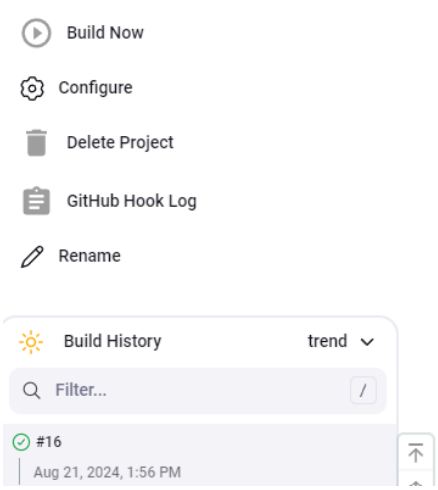




After Change

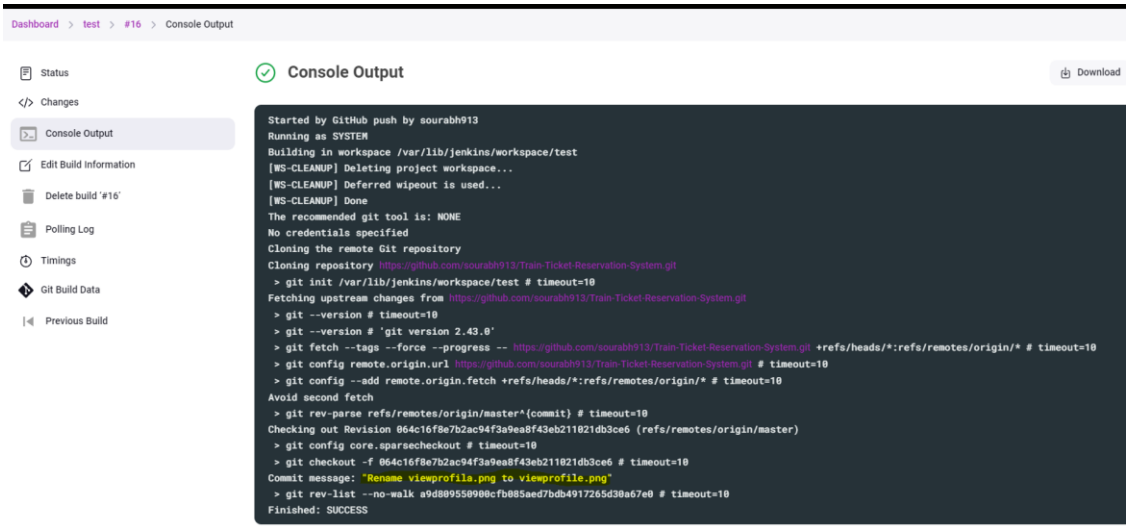


Immediately after the change, we see that build ran in Jenkins.



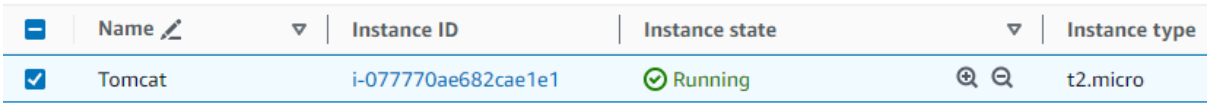
- Last build (#16), 45 sec ago
- Last stable build (#16), 45 sec ago
- Last successful build (#16), 45 sec ago
- Last completed build (#16), 45 sec ago

We can see the changes in the console output reflected successfully



5) How to install Jenkins in Tomcat.

- To install jenkins in Tomcat, we need to have a EC2 instance with open-jdk installed on it.



Add the below commands in user data section of EC2 instance before launching it to install open-jdk on it.

```
#!/bin/bash

apt update -y && apt upgrade -y && apt install openjdk-17-jdk -y
```

- Once EC2 is up, we will verify that OpenJDK is installed properly by checking its version.

java --version (Verify open jdk is installed)

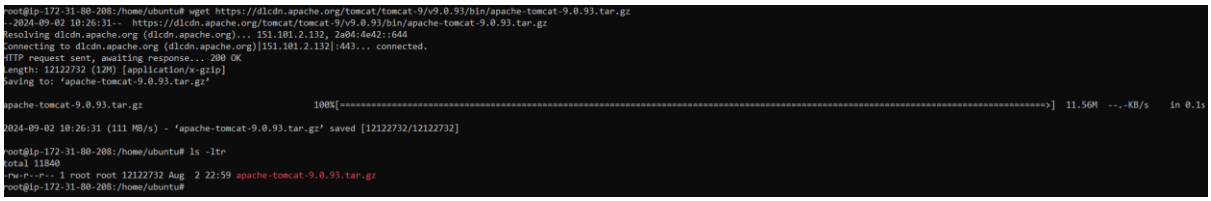
```
root@ip-172-31-80-208:/home/ubuntu# java --version
openjdk 17.0.12 2024-07-16
OpenJDK Runtime Environment (build 17.0.12+7-Ubuntu-1ubuntu224.04)
OpenJDK 64-Bit Server VM (build 17.0.12+7-Ubuntu-1ubuntu224.04, mixed mode, sharing)
root@ip-172-31-80-208:/home/ubuntu#
```

- Now we install Apache tomcat version 9 so that we can install jenkins on it.

- Since Tomcat version 9 is not available as default package in ubuntu so we have to install Tomcat 9 manually. Below are the steps to download and install tomcat 9.

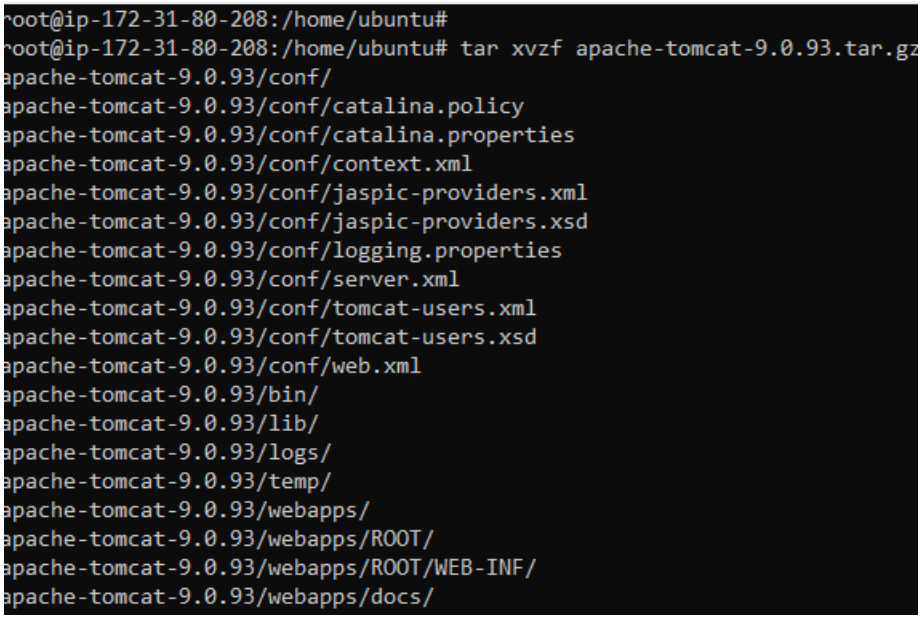
wget https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.93/bin/apache-tomcat-9.0.93.tar.gz

ls -ltr to check if the file is downloaded

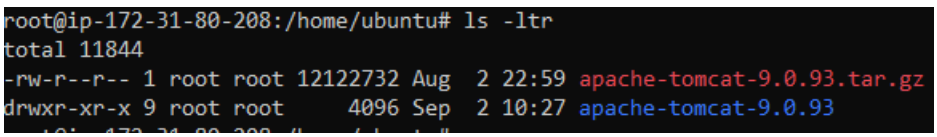


- Untar the tar file using below command and check tomcat folder is created or not.

tar xzvf apache-tomcat-9.0.93.tar.gz



ls -ltr



- Now rename the extracted folder tomcat so that it's easy to remember and check.

mv apache-tomcat-9.0.93 tomcat

ls -ltr

```
root@ip-172-31-80-208:/home/ubuntu# mv apache-tomcat-9.0.93 tomcat
root@ip-172-31-80-208:/home/ubuntu#
root@ip-172-31-80-208:/home/ubuntu# ls -ltr
total 11844
-rw-r--r-- 1 root root 12122732 Aug  2 22:59 apache-tomcat-9.0.93.tar.gz
drwxr-xr-x 9 root root     4096 Sep  2 10:27 tomcat
root@ip-172-31-80-208:/home/ubuntu#
```

- Now we have to create the tomcat service manually in the system location mentioned below and add the contents in the service file.

vi /etc/systemd/system/tomcat.service

```
root@ip-172-31-80-208:/home/ubuntu# vi /etc/systemd/system/tomcat.service
```

Contents of tomcat.service file

[Unit]
Description=Tomcat 9 servlet container
After=network.target

[Service]
Type=forking

Environment="JAVA_HOME=/usr/lib/jvm/java-17-openjdk-amd64"
Environment="JAVA_OPTS=-Djava.security.egd=file:///dev/urandom -Djava.awt.headless=true"

Environment="CATALINA_BASE=/home/ubuntu/tomcat/"
Environment="CATALINA_HOME=/home/ubuntu/tomcat/"
Environment="CATALINA_PID=/home/ubuntu/tomcat/temp/tomcat.pid"
Environment="CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"

ExecStart=/home/ubuntu/tomcat/bin/startup.sh
ExecStop=/home/ubuntu/tomcat/bin/shutdown.sh

[Install]
WantedBy=multi-user.target

```
root@ip-172-31-80-208: /home/ubuntu
[Unit]
Description=Tomcat 9 servlet container
After=network.target

[Service]
Type=forking

Environment="JAVA_HOME=/usr/lib/jvm/java-17-openjdk-amd64"
Environment="JAVA_OPTS=-Djava.security.egd=file:///dev/urandom -Djava.awt.headless=true"

Environment="CATALINA_BASE=/home/ubuntu/tomcat/"
Environment="CATALINA_HOME=/home/ubuntu/tomcat/"
Environment="CATALINA_PID=/home/ubuntu/tomcat/temp/tomcat.pid"
Environment="CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"

ExecStart=/home/ubuntu/tomcat/bin/startup.sh
ExecStop=/home/ubuntu/tomcat/bin/shutdown.sh

[Install]
WantedBy=multi-user.target
```

- Now we will reload the daemon and start tomcat service and also verify that the service is running.

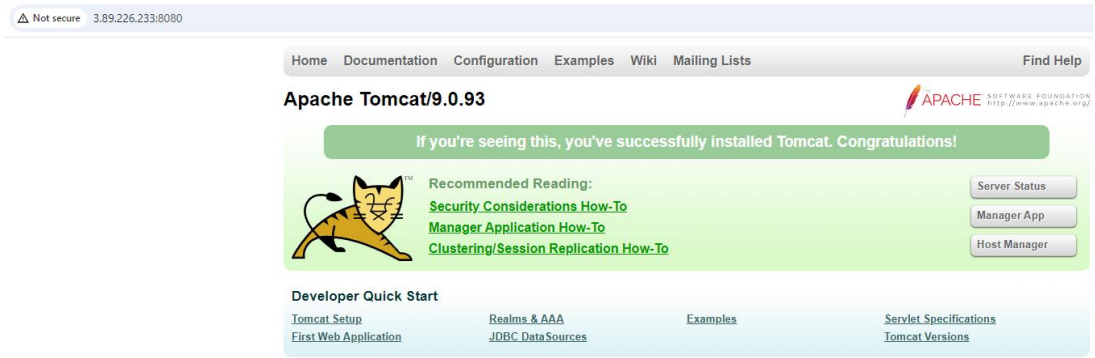
  systemctl daemon-reload

  systemctl start tomcat

  systemctl status tomcat

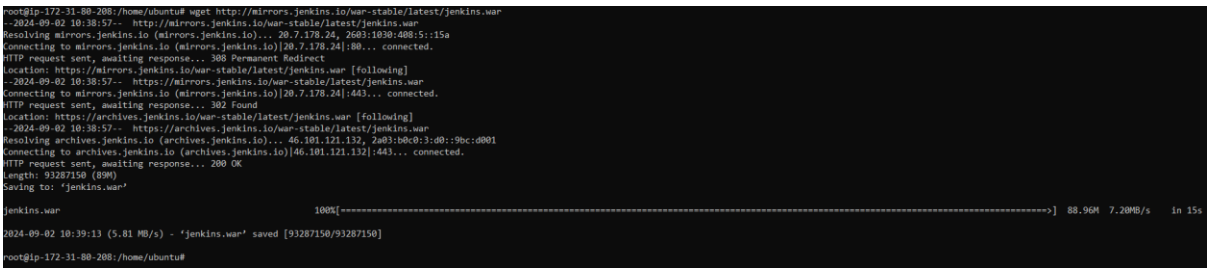
- We will verify that tomcat is running or not on the WEG UI by accessing tomcat URL – PublicIP of EC2 instance:8080 (Tomcat works on port 8080)

http://3.89.226.233:8080

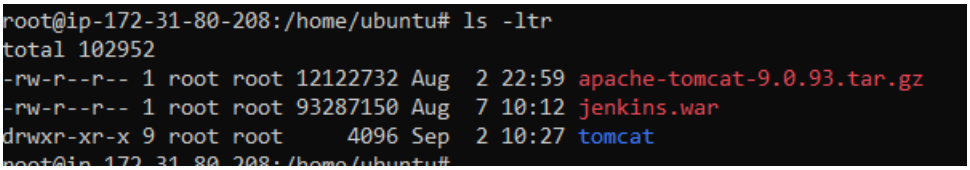


- Once we have apache tomcat 9 up and running, we will download jenkins.war file from the official webpage from the terminal by running the below command.

wget http://mirrors.jenkins.io/war-stable/latest/jenkins.war



ls -ltr



- Now we need to copy this file to webapps directory located in Tomcat home. So, if you have tomcat installed in home directory, then run the following command.

cp -rv jenkins.war tomcat/webapps

```
root@ip-172-31-80-208:/home/ubuntu# cp -rv jenkins.war tomcat/webapps
'jenkins.war' -> 'tomcat/webapps/jenkins.war'
root@ip-172-31-80-208:/home/ubuntu#
```

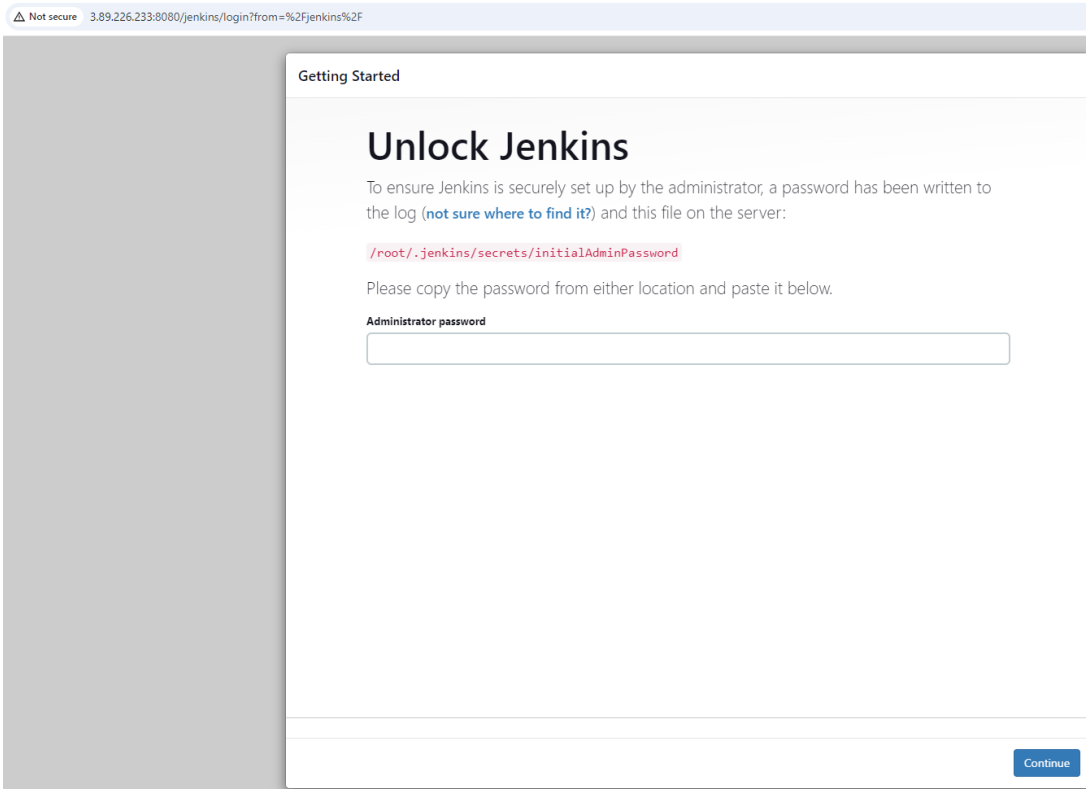
- We will also verify that jenkins.war has been copied successfully and also jenkins folder is also created.

  ls -ltr tomcat/webapps

```
root@ip-172-31-80-208:/home/ubuntu# ls -ltr tomcat/webapps
total 91128
drwxr-x---  3 root root     4096 Sep  2 10:27 ROOT
drwxr-x--- 16 root root     4096 Sep  2 10:27 docs
drwxr-x---  7 root root     4096 Sep  2 10:27 examples
drwxr-x---  6 root root     4096 Sep  2 10:27 host-manager
drwxr-x---  6 root root     4096 Sep  2 10:27 manager
-rw-r--r--  1 root root 93287150 Sep  2 10:40 jenkins.war
drwxr-x--- 10 root root     4096 Sep  2 10:40 jenkins
root@ip-172-31-80-208:/home/ubuntu#
```

- Since our tomcat server is up. To start using the jenkins, we will visit the following URL.

  http://3.89.226.233:8080/jenkins



- We will have to configure the jenkins server now. As mentioned on the webpage, we can find the admin password from location '/root/.jenkins/secrets/initialAdminPassword' . Get the password with the command,

  cat /root/.jenkins/secrets/initialAdminPassword

```
root@ip-172-31-80-208:/home/ubuntu# cat /root/.jenkins/secrets/initialAdminPassword
5d866465f9bb44ac9bb7ad7e9b738451
root@ip-172-31-80-208:/home/ubuntu#
```

- Now complete the setup by filling out the required details. After the setup is complete, we can access jenkins & can also create new jobs on the jenkins server.

6) Install old version of Jenkins and then upgrade it to a latest version

- To install old version of jenkins, we need to have a EC2 instance with open-jdk installed on it.

- Add the below commands in user data section of EC2 instance before launching it to install open-jdk on it.

```
#!/bin/bash

apt update -y && apt upgrade -y && apt install openjdk-17-jdk -y
```

- Once EC2 is up, we will verify that OpenJDK is installed properly by checking its version.

  java --version (Verify open jdk is installed)

```
root@ip-172-31-81-148:/home/ubuntu# java --version
openjdk 17.0.12 2024-07-16
OpenJDK Runtime Environment (build 17.0.12+7-Ubuntu-1ubuntu224.04)
OpenJDK 64-Bit Server VM (build 17.0.12+7-Ubuntu-1ubuntu224.04, mixed mode, sharing)
root@ip-172-31-81-148:/home/ubuntu#
```

- Now we will prepare the system for jenkins in ubuntu by executing the following command.

- Below is the Debian package repository of Jenkins to automate installation and upgrade. To use this repository, first add the key to your system

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

```
root@ip-172-31-81-13:/home/ubuntu# sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
>   https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
--2024-09-02 13:04:39--  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
Resolving pkg.jenkins.io (pkg.jenkins.io)... 146.75.38.133, 2a04:4e42:79::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|146.75.38.133|: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-k 100%[=========================================>]   3.10K  --.-KB/s    in 0s

2024-09-02 13:04:39 (45.8 MB/s) - '/usr/share/keyrings/jenkins-keyring.asc' saved [3175/3175]

root@ip-172-31-81-13:/home/ubuntu#
```

- Add a Jenkins apt repository entry:

```
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
```

```
root@ip-172-31-81-13:/home/ubuntu# 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
root@ip-172-31-81-13:/home/ubuntu#
```

```
apt-get update
```

```
root@ip-172-31-81-13:/home/ubuntu# apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Hit:3 http://us-east-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 [27.6 kB]
Fetched 157 kB in 1s (205 kB/s)
Reading package lists... Done
root@ip-172-31-81-13:/home/ubuntu#
```

- Now we will check which all versions of Jenkins are available, so that we can install and old version and then perform an upgrade to the latest version. Below is the command for the same. (It will give all the versions starting from the oldest to the latest.

  apt-cache madison jenkins

```
root@ip-172-31-81-13:/home/ubuntu# sudo apt-cache madison jenkins
   jenkins |     2.462.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.452.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.452.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.452.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.452.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.440.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.440.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.440.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.426.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.426.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.426.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.414.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.414.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.414.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.401.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.401.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.401.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.387.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.387.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.387.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.375.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.375.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.375.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.375.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.375.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.361.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.361.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.361.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.361.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.346.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.346.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.346.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.332.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.332.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.332.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.332.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.319.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.319.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.319.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.303.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.303.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.303.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.289.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.289.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.289.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.277.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.277.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.277.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.277.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.263.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.263.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.263.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.263.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.249.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.249.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.249.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.235.5 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.235.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.235.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.235.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.235.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.222.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.222.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.222.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.204.6 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.204.5 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.204.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.204.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.204.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.204.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.190.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.190.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.190.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.176.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.176.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.176.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
   jenkins |     2.176.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
```

```
jenkins |     2.164.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.164.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.164.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.150.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.150.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.150.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.138.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.138.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.138.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.138.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.121.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.121.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.121.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.107.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.107.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     2.107.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.89.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.89.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.89.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.89.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.73.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.73.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.73.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.60.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.60.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.60.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |    2.60.1-1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.60.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.46.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.46.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.32.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.32.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.32.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.19.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.19.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |      2.19.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |       2.7.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |       2.7.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |       2.7.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.651.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.651.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.642.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.642.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.642.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.642.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.625.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.625.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.609.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.609.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.609.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.596.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.596.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.596.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.580.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.580.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.580.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.565.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.554.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.532.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.532.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.532.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.509.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.509.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.509.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.480.3 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.480.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.480.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.466.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.466.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.447.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.447.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.424.6 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.424.5 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.424.4 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.424.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.424.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.409.2 | https://pkg.jenkins.io/debian-stable binary/ Packages
jenkins |     1.409.1 | https://pkg.jenkins.io/debian-stable binary/ Packages
```

- Here we have randomly selected a version to install i.e. 2.387.3. To install the specific jenkins version, we need to run the below command.

  apt-get install jenkins=2.387.3 -y

```
root@ip-172-31-81-13:/home/ubuntu# apt-get install jenkins=2.387.3 -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  net-tools
The following NEW packages will be installed:
  jenkins net-tools
0 upgraded, 2 newly installed, 0 to remove and 2 not upgraded.
Need to get 98.1 MB of archives.
After this operation, 99.2 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 net-tools amd64 2.10-0.1ubuntu4 [204 kB]
Get:2 https://pkg.jenkins.io/debian-stable binary/ jenkins 2.387.3 [97.9 MB]
Fetched 98.1 MB in 18s (5348 kB/s)
Selecting previously unselected package net-tools.
(Reading database ... 113852 files and directories currently installed.)
Preparing to unpack .../net-tools_2.10-0.1ubuntu4_amd64.deb ...
Unpacking net-tools (2.10-0.1ubuntu4) ...
```

- We can see in the below screenshot that Jenkins 2.387.3 version is installed.

```
Selecting previously unselected package jenkins.
Preparing to unpack .../jenkins_2.387.3_all.deb ...
Unpacking jenkins (2.387.3) ...
Setting up net-tools (2.10-0.1ubuntu4) ...
Setting up jenkins (2.387.3) ...
Created symlink /etc/systemd/system/multi-user.target.wants/jenkins.service → /usr/lib/systemd/system/jenkins.service.
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
  6.8.0-1012-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.8.0-1014-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...
 systemctl restart acpid.service chrony.service cron.service multipathd.service polkit.service udisks2.service

Service restarts being deferred:
 systemctl restart ModemManager.service
 /etc/needrestart/restart.d/dbus.service
 systemctl restart getty@tty1.service
 systemctl restart networkd-dispatcher.service
```

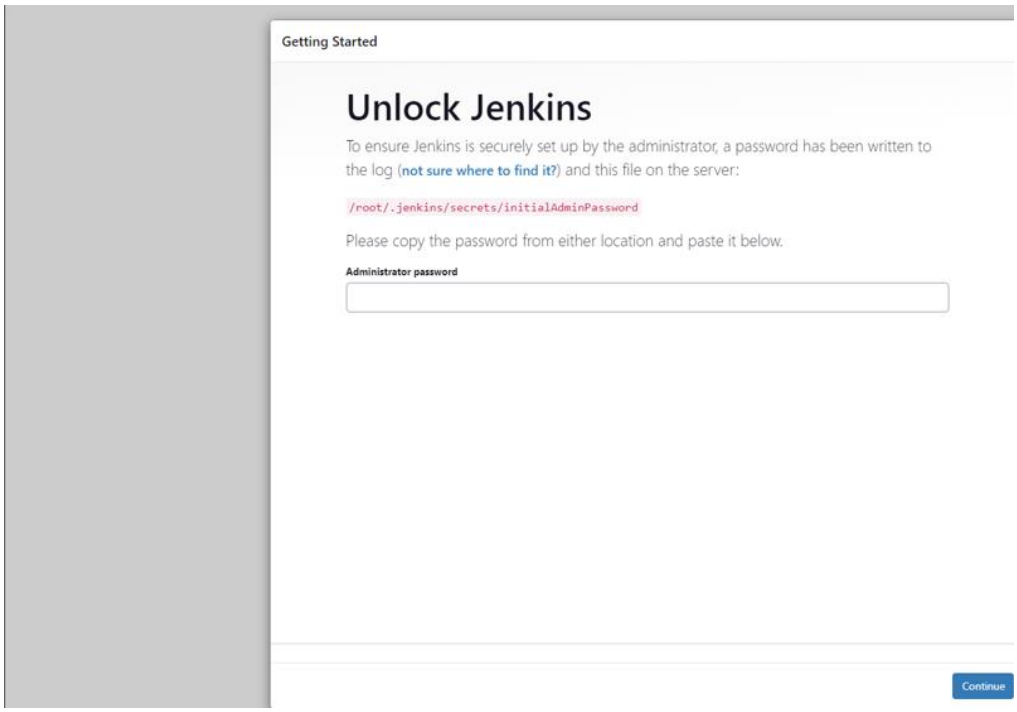
- Check status of Jenkins service.

systemctl status jenkins

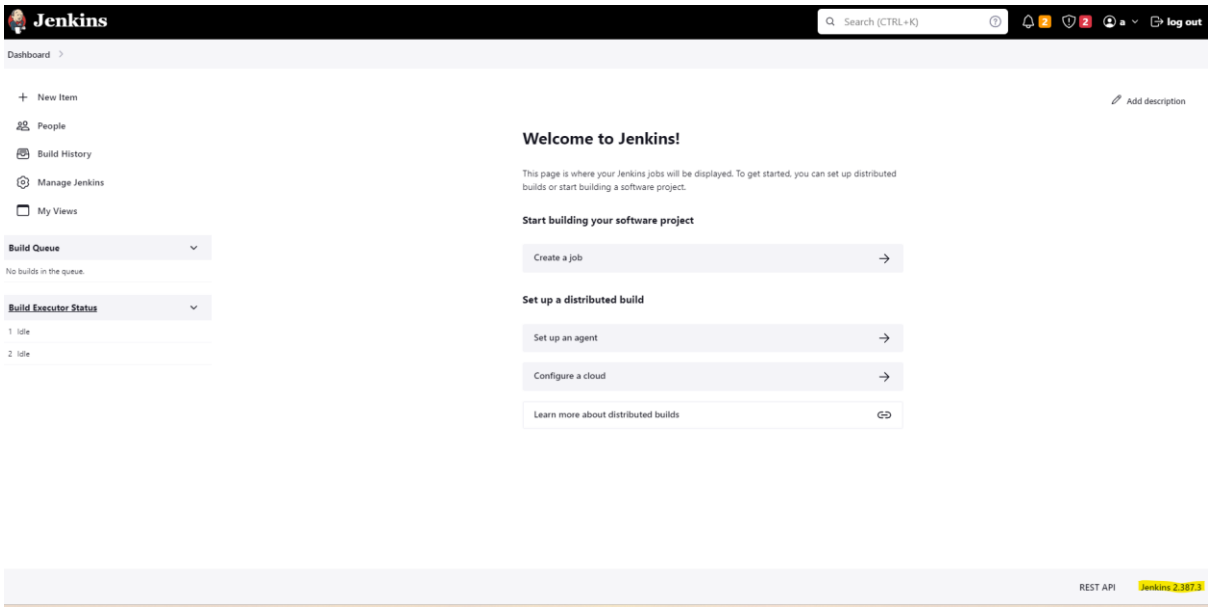
```
root@ip-172-31-25-17:/home/ubuntu# systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
     Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
     Active: active (running) since Mon 2024-09-02 14:41:12 UTC; 32s ago
   Main PID: 16218 (java)
      Tasks: 42 (limit: 1130)
     Memory: 265.3M (peak: 352.8M)
        CPU: 47.809s
     CGroup: /system.slice/jenkins.service
             └─16218 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jen>

Sep 02 14:40:35 ip-172-31-25-17 jenkins[16218]: ee4fdf3fd91744b1a562091c3172130c
Sep 02 14:40:35 ip-172-31-25-17 jenkins[16218]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassw>
Sep 02 14:40:35 ip-172-31-25-17 jenkins[16218]: *************************************************************
Sep 02 14:40:35 ip-172-31-25-17 jenkins[16218]: *************************************************************
Sep 02 14:40:35 ip-172-31-25-17 jenkins[16218]: *************************************************************
Sep 02 14:41:12 ip-172-31-25-17 jenkins[16218]: 2024-09-02 14:41:12.809+0000 [id=31]        INFO        jenkins.InitR>
Sep 02 14:41:12 ip-172-31-25-17 jenkins[16218]: 2024-09-02 14:41:12.870+0000 [id=24]        INFO        hudson.lifecy>
Sep 02 14:41:12 ip-172-31-25-17 systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Sep 02 14:41:13 ip-172-31-25-17 jenkins[16218]: 2024-09-02 14:41:13.010+0000 [id=46]        INFO        h.m.DownloadS>
Sep 02 14:41:13 ip-172-31-25-17 jenkins[16218]: 2024-09-02 14:41:13.012+0000 [id=46]        INFO        hudson.util.R>
lines 1-20/20 (END)
```

- We will verify that jenkins server is up and running and perform post deployment steps



- Once the EC2 instance is up, Post deployment steps are performed for configuring Jenkins to setup the Jenkins server. Once Jenkins is configured, we see the below dashboard screen and below screenshot will give you the confirmation of version 2.387.3 jenkins setup.



- Before upgrading to the latest version, we need to stop jenkins service by using the below command.

  systemctl stop jenkins

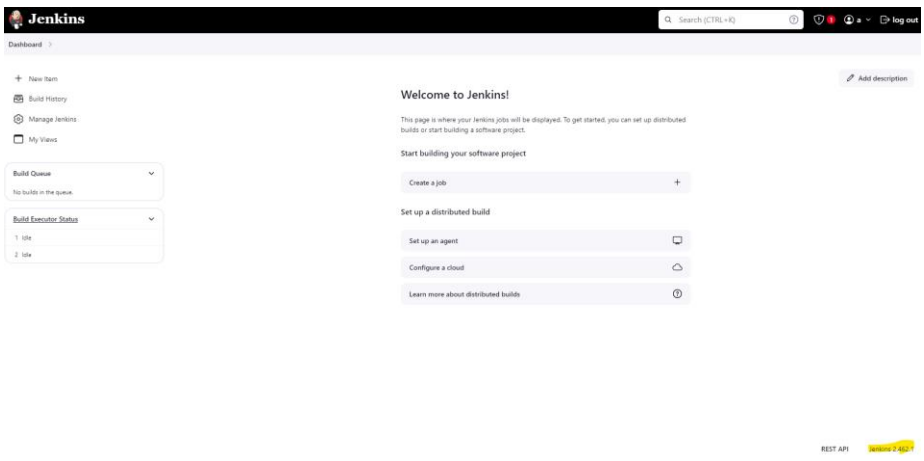
```
root@ip-172-31-25-17:/home/ubuntu# systemctl stop jenkins
root@ip-172-31-25-17:/home/ubuntu#
```

- Once the jenkins service is stopped, we will upgrade this version 2.387.3 to the latest version i.e. 2.462.1, by running the below command on the terminal. Below screenshot shows latest version is setting up.

  apt-get upgrade -y

```
root@ip-172-31-25-17:/home/ubuntu# apt-get upgrade -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following upgrades have been deferred due to phasing:
  ubuntu-pro-client ubuntu-pro-client-l10n
The following packages will be upgraded:
  jenkins
1 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
Need to get 91.2 MB of archives.
After this operation, 5068 kB disk space will be freed.
Get:1 https://pkg.jenkins.io/debian-stable binary/ jenkins 2.462.1 [91.2 MB]
Fetched 91.2 MB in 2s (38.4 MB/s)
(Reading database ... 113914 files and directories currently installed.)
Preparing to unpack .../jenkins_2.462.1_all.deb ...
Unpacking jenkins (2.462.1) over (2.387.3) ...
Setting up jenkins (2.462.1) ...
Installing new version of config file /etc/init.d/jenkins ...
Scanning processes...
```

- Now we will verify the version is updated successfully by logging into the jenkins console again. Below screenshot confirms the same.



7) Jenkins Installation and Configuration Script

- Jenkins installation and configuration script is broken down into three components.

    - Jenkins Installation
    - Create Admin User
    - Installation of recommended plugins
    - Confirming Jenkins URL

- Jenkins installation script is as below:

```
#!/bin/bash

#JENKINS INSTALLATION

# Update and Upgrade Packages of Ubuntu OS
sudo apt update -y && sudo apt upgrade -y

#Install Java SDK 17
sudo apt install openjdk-17-jdk -y

# Check Java version
java -version

# Add Jenkins repository key
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key

# Add Jenkins repository to sources list
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

# Update package lists
sudo apt-get update
```

```
# Install Jenkins
sudo apt-get install jenkins -y

#Start Jenkins
sudo systemctl start jenkins

# Enable Jenkins to run on Boot
sudo systemctl enable jenkins
```

- Jenkins create admin user script is as below: Highlighted items in below script will be changed according to your requirements

```
#! /bin/bash


#JENKINS ADMIN USER CREATION


# Install python3-pip (Prerequisite)

sudo apt install python3-pip -y


# Install Dev Packages of Python (Prerequisite)

sudo apt install -y build-essential libssl-dev libffi-dev python3-dev


url=<Jenkins URL>

password=$(sudo cat /var/lib/jenkins/secrets/initialAdminPassword)


# NEW ADMIN CREDENTIALS URL ENCODED USING PYTHON

username=$(python3 -c "import urllib.parse;print
(urllib.parse.quote(input(), safe=''))" <<< "user")

new_password=$(python3 -c "import urllib.parse;print
(urllib.parse.quote(input(), safe=''))" <<< "password")
```

```
fullname=$(python3 -c "import urllib.parse;print
(urllib.parse.quote(input(), safe=''))" <<< "full name")

email=$(python3 -c "import urllib.parse;print (urllib.parse.quote(input(),
safe=''))" <<< "hello@world.com")


# GET THE CRUMB AND COOKIE

cookie_jar="$(mktemp)"

full_crumb=$(curl -u "admin:$password" --cookie-jar "$cookie_jar"
$url/crumbIssuer/api/xml?xpath=concat\(//crumbRequestField,%22:%22
,//crumb\))

arr_crumb=(${full_crumb//:/ })

only_crumb=$(echo ${arr_crumb[1]})


# MAKE THE REQUEST TO CREATE AN ADMIN USER

curl -X POST -u "admin:$password" $url/setupWizard/createAdminUser \

      -H "Connection: keep-alive" \

      -H "Accept: application/json, text/javascript" \

      -H "X-Requested-With: XMLHttpRequest" \

      -H "$full_crumb" \

      -H "Content-Type: application/x-www-form-urlencoded" \

      --cookie $cookie_jar \

      --data-raw
"username=$username&password1=$new_password&password2=$new
_password&fullname=$fullname&email=$email&Jenkins-
Crumb=$only_crumb&json=%7B%22username%22%3A%20%22$userna
me%22%2C%20%22password1%22%3A%20%22$new_password%22%2
C%20%22%24redact%22%3A%20%5B%22password1%22%2C%20%22pa
ssword2%22%5D%2C%20%22password2%22%3A%20%22$new_passwor
d%22%2C%20%22fullname%22%3A%20%22$fullname%22%2C%20%22e
mail%22%3A%20%22$email%22%2C%20%22Jenkins-
```

```
Crumb%22%3A%20%22$only_crumb%22%7D&core%3Aapply=&Submit
=Save&json=%7B%22username%22%3A%20%22$username%22%2C%20
%22password1%22%3A%20%22$new_password%22%2C%20%22%24re
dact%22%3A%20%5B%22password1%22%2C%20%22password2%22%5
D%2C%20%22password2%22%3A%20%22$new_password%22%2C%20
%22fullname%22%3A%20%22$fullname%22%2C%20%22email%22%3A
%20%22$email%22%2C%20%22Jenkins-
Crumb%22%3A%20%22$only_crumb%22%7D"
```

- Jenkins install recommended plugin script is as below:

```
#!bin/bash

#INSTALLATION OF RECOMMENDED PLUGINS IN JENKINS

url=<jenkins URL>

#ENTER THE CREDENTIALS HIGHLIGHTED IN THE ABOVE CODE
user=user
password=password

cookie_jar="$(mktemp)"
full_crumb=$(curl -u "$user:$password" --cookie-jar "$cookie_jar"
$url/crumbIssuer/api/xml?xpath=concat\(//crumbRequestField,%22:%22
,//crumb\))
arr_crumb=(${full_crumb//:/ })
only_crumb=$(echo ${arr_crumb[1]})

# MAKE THE REQUEST TO DOWNLOAD AND INSTALL REQUIRED
MODULES
 curl -X POST -u "$user:$password" $url/pluginManager/installPlugins \
  -H 'Connection: keep-alive' \
   -H 'Accept: application/json, text/javascript, */*; q=0.01' \
    -H 'X-Requested-With: XMLHttpRequest' \
     -H "$full_crumb" \
      -H 'Content-Type: application/json' \
       -H 'Accept-Language: en,en-US;q=0.9,it;q=0.8' \
```

```
        --cookie $cookie_jar \
         --data-raw "{'dynamicLoad':true,'plugins':['cloudbees-
folder','antisamy-markup-formatter','build-timeout','credentials-
binding','timestamper','ws-cleanup','ant','gradle','workflow-
aggregator','github-branch-source','pipeline-github-lib','pipeline-stage-
view','git','ssh-slaves','matrix-auth','pam-auth','ldap','email-
ext','mailer'],'Jenkins-Crumb':'$only_crumb'}"
```

- Confirming Jenkins URL script is as below:

```
#!/bin/bash

#CONFIRMING JENKINS URL

url=<your_jenkins_url>

user=user
password=password

url_urlEncoded=$(python3 -c "import urllib.parse;print
(urllib.parse.quote(input(), safe=''))" <<< "$url")

cookie_jar="$(mktemp)"
full_crumb=$(curl -u "$user:$password" --cookie-jar "$cookie_jar"
$url/crumbIssuer/api/xml?xpath=concat\(//crumbRequestField,%22:%22
,//crumb\))
arr_crumb=(${full_crumb//:/ })
only_crumb=$(echo ${arr_crumb[1]})

curl -X POST -u "$user:$password" $url/setupWizard/configureInstance \
  -H 'Connection: keep-alive' \
  -H 'Accept: application/json, text/javascript, */*; q=0.01' \
  -H 'X-Requested-With: XMLHttpRequest' \
  -H "$full_crumb" \
  -H 'Content-Type: application/x-www-form-urlencoded' \
  -H 'Accept-Language: en,en-US;q=0.9,it;q=0.8' \
  --cookie $cookie_jar \
```

--data-raw "rootUrl=$url_urlEncoded%2F&Jenkins-Crumb=$only_crumb&json=%7B%22rootUrl%22%3A%20%22$url_urlEncoded%2F%22%2C%20%22Jenkins-Crumb%22%3A%20%22$only_crumb%22%7D&core%3Aapply=&Submit=Save&json=%7B%22rootUrl%22%3A%20%22$url_urlEncoded%2F%22%2C%20%22Jenkins-Crumb%22%3A%20%22$only_crumb%22%7D"

NOTE: https://kevin-denotariis.medium.com/download-install-and-setup-jenkins-completely-from-bash-unlock-create-admin-user-and-more-debd3320414a **(Script Reference, You can refer the above link for detail explanation)**

8) **How to take backup in Jenkins and Restore.**

- To perform the backup of Jenkins and Restore the backup on other machine, we need to install/perform following prerequisites:

❖ Need to have two AWS EC2 instance of Jenkins.

- ➢ First Jenkins will be the primary one where we will create jobs and there will build history and all and we will take the backup of this server which will include jobs, configs and all.
- ➢ Second Jenkins will be the backup one, which will be simple jenkins installed instance with default settings and all and this instance will be used to restore the backup of the first Jenkins server.

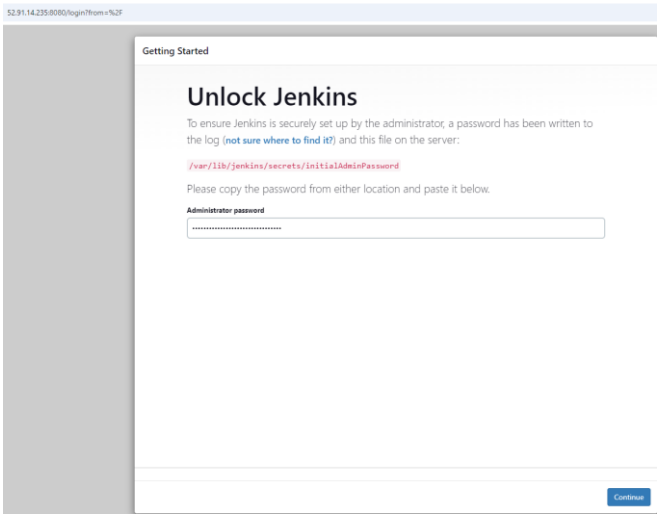
❖ Need to have S3 bucket created which will be used to store the jenkins backup and also will be used for restore.

❖ IAM role for S3 full access, this role is required so that we can upload the backup on S3 bucket from first Jenkins instance and also download backup to restore on second Jenkins instance.

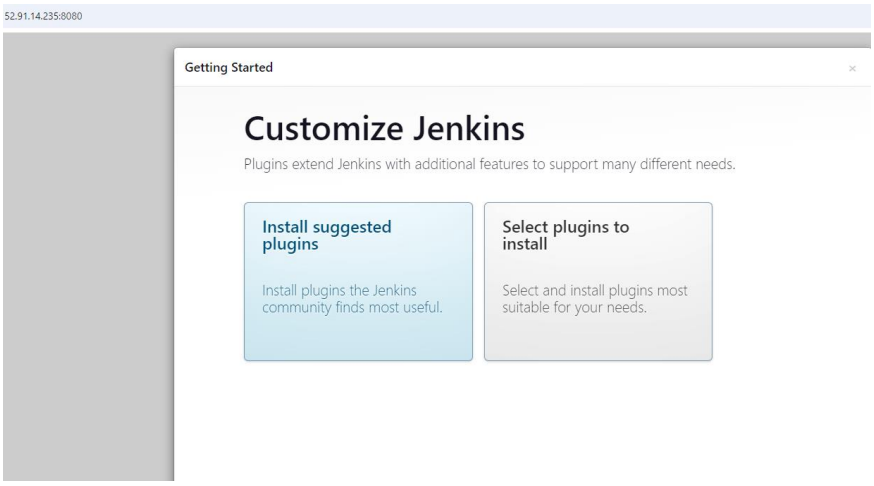
❖ We need have aws cli installed using python on both the jenkins instances so that we can run the commands to upload to S3 bucket and download from the S3 bucket.

- **Now will setup the first jenkins instance. In our case below is the first server.**

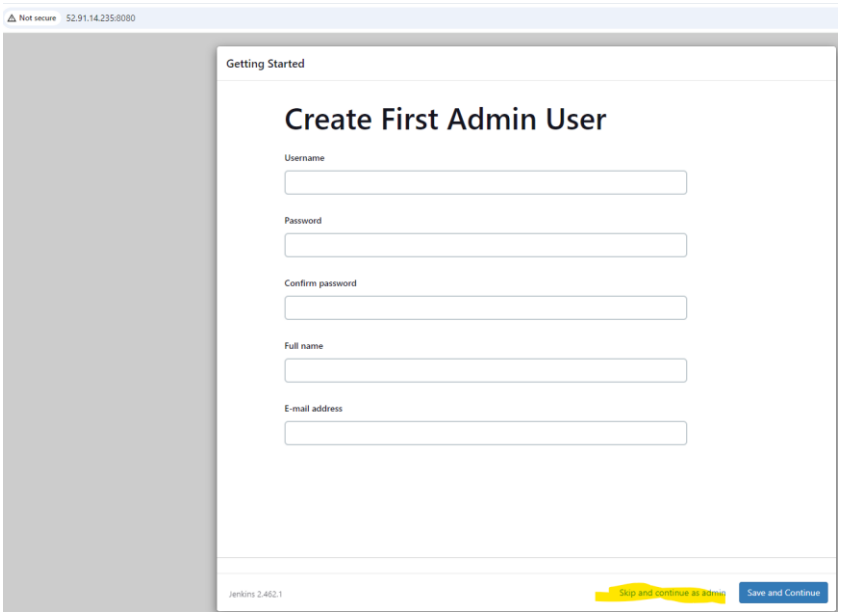
✓ We will get the default admin password and put on the below screen.
NOTE: Make a note of default password



✓ We will install suggested plugins

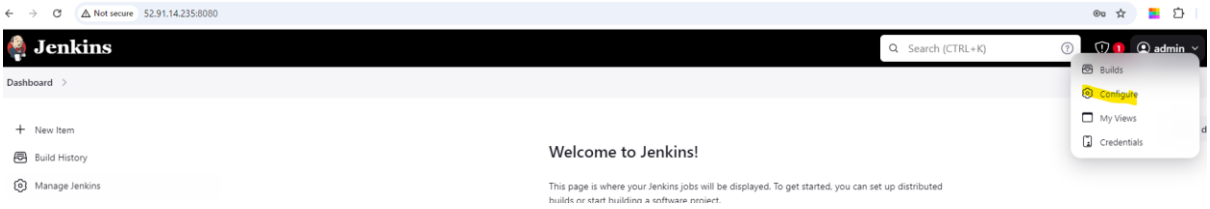


✓ We will skip the next screen and continue as admin

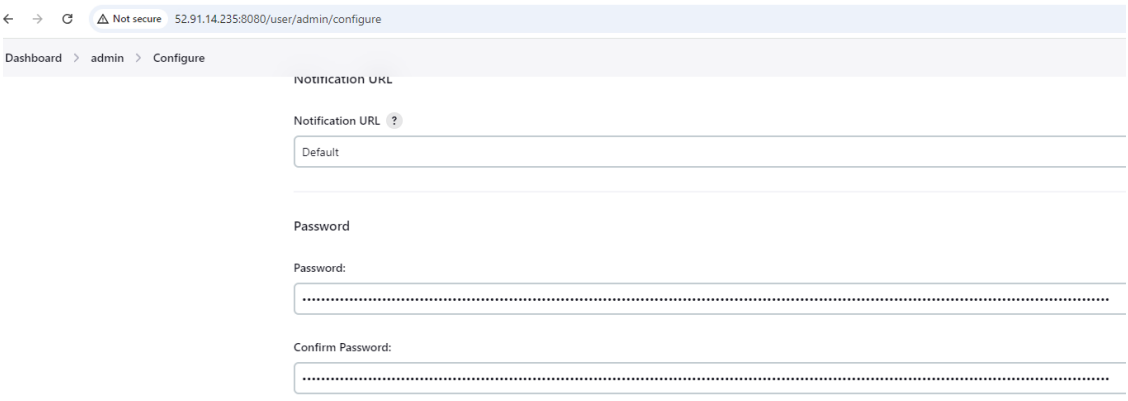


✓ Verify Jenkins URL and start using Jenkins.

✓ Now we will change the admin password as per our convenience by dropping down on the admin > Configure

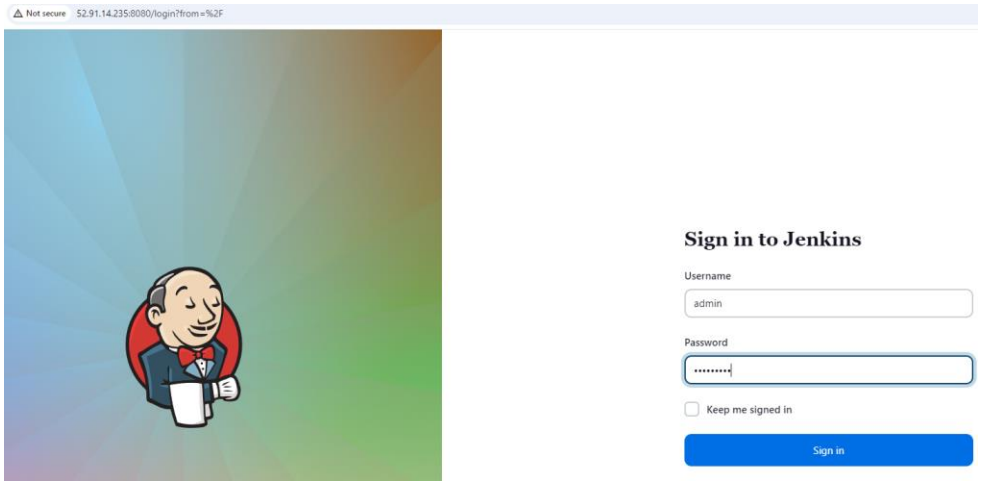


✓ Change the password at the location mentioned in the below screenshot and click Apply and Save



✓ Below Screen will appear

✓ Now login to admin user with the new credentials and verify the password change

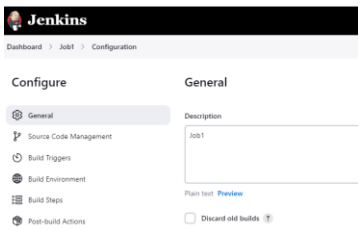


✓ You should be able to get in with new credentials.

✓ Now we will create two jobs so that we have some data is there which we can backup.

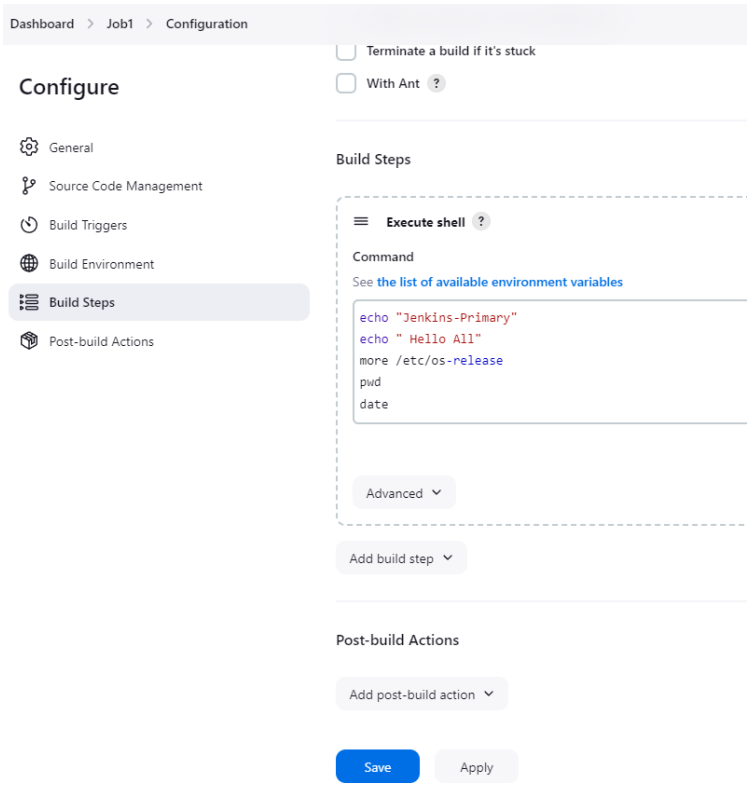
✓ Job 1 is freestyle project and below are the configurations.

General > Description > Job1

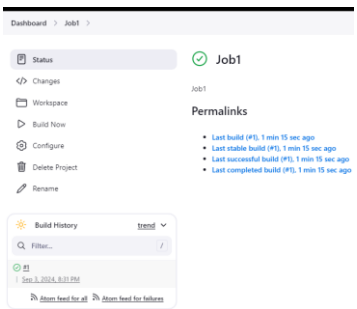


✓ Build Steps > Execute Shell >Command

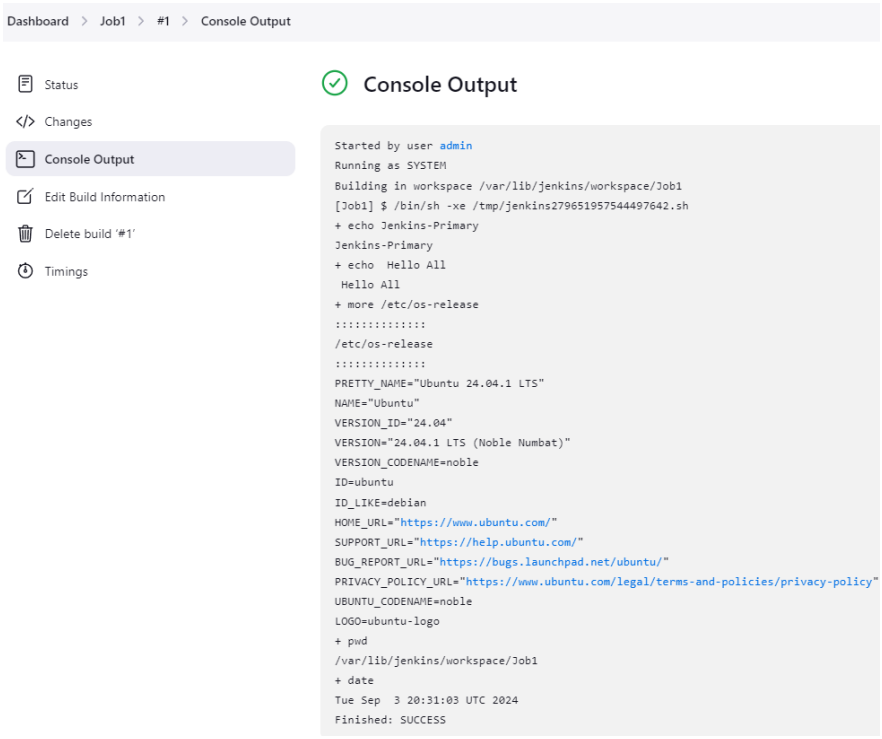
```
echo "Jenkins-Primary"
echo " Hello All"
more /etc/os-release
pwd
date
```



✓ Build Now and build is successful.

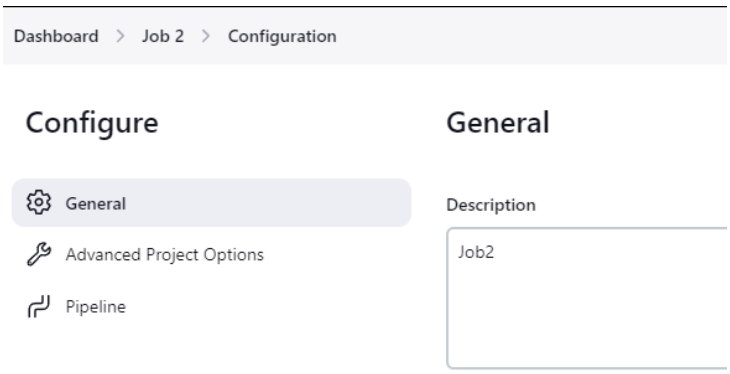


✓ console output shows the below output.



✓ Job 2 is pipeline project and below are the configurations.

General > Description > Job2

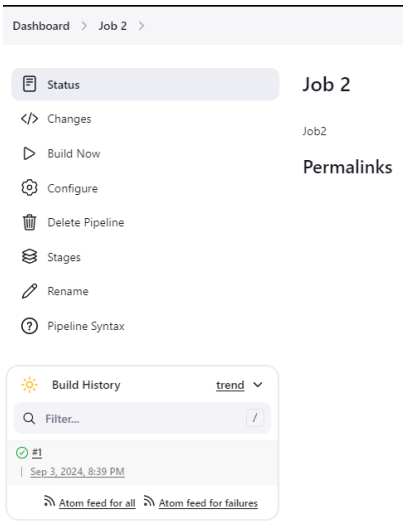


Under Pipline > Pipline script > Script

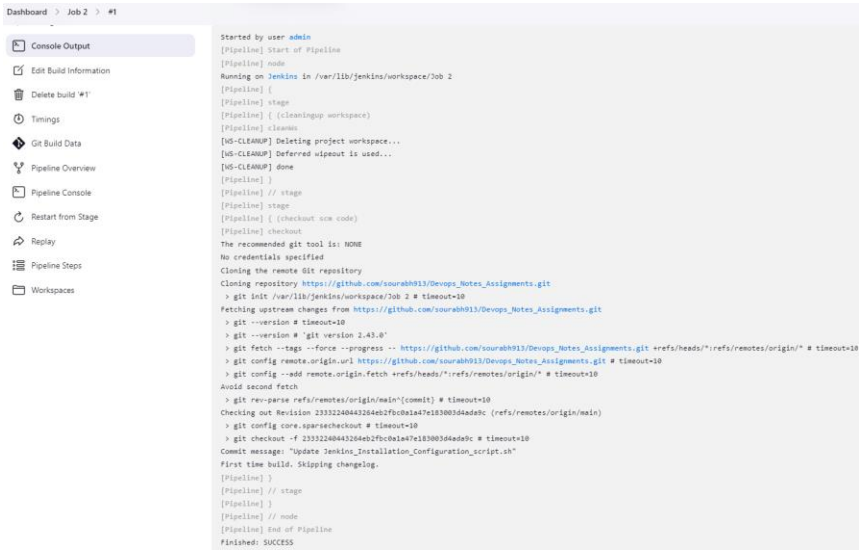
```
pipeline {
   agent any
   stages {
      stage ("cleaningup workspace") {
         steps {
```

```
            cleanWs()
        }
    }
    stage ("checkout scm code") {
        steps {
            checkout scmGit(branches: [[name: '*/main']], extensions: [],
userRemoteConfigs: [[url:
'https://github.com/sourabh913/Devops_Notes_Assignments.git']])
        }
    }
 }
}
```

✓ Build Now and build is successful.

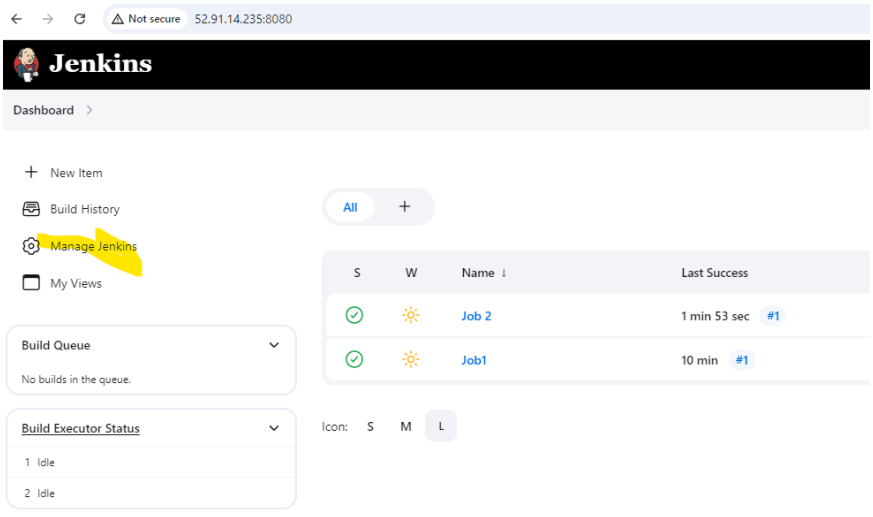


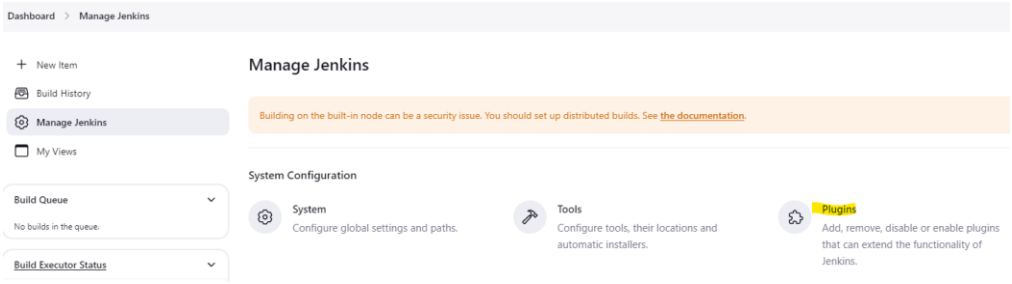
✓ console output shows the below output.



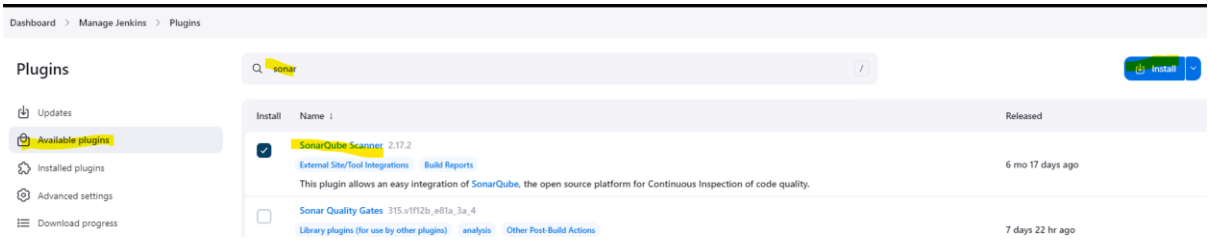
✓ For Backup perspective, we will install any available plugin which is not installed by default with recommended plugins. In our case we will install SonarQube Scanner plugin.

We will go to Manage Jenkins > Plugins

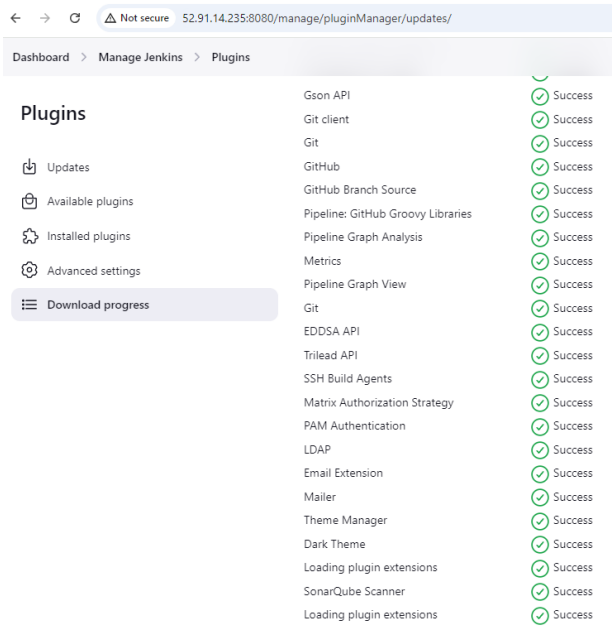




Under Available plugins > Select SonarQubeScanner > Install

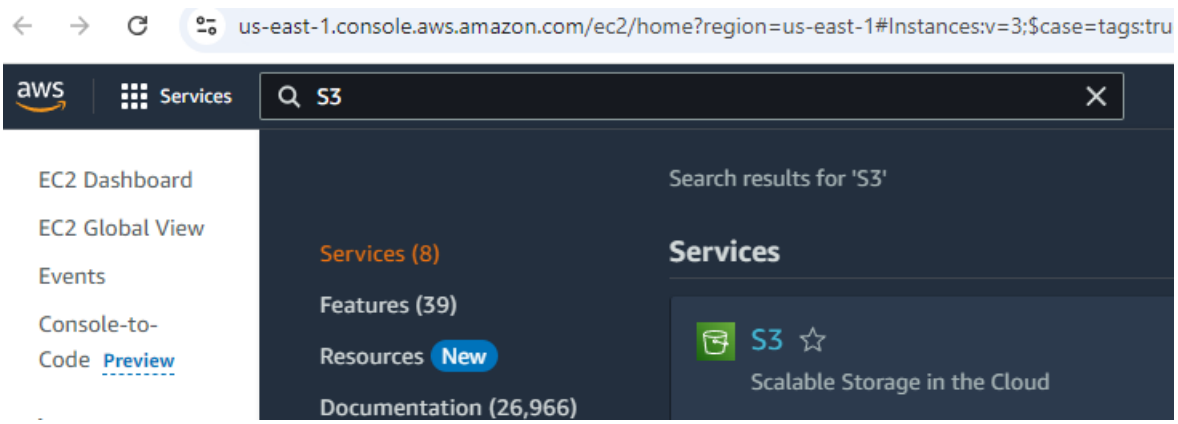


✓ Plugin installed successfully.

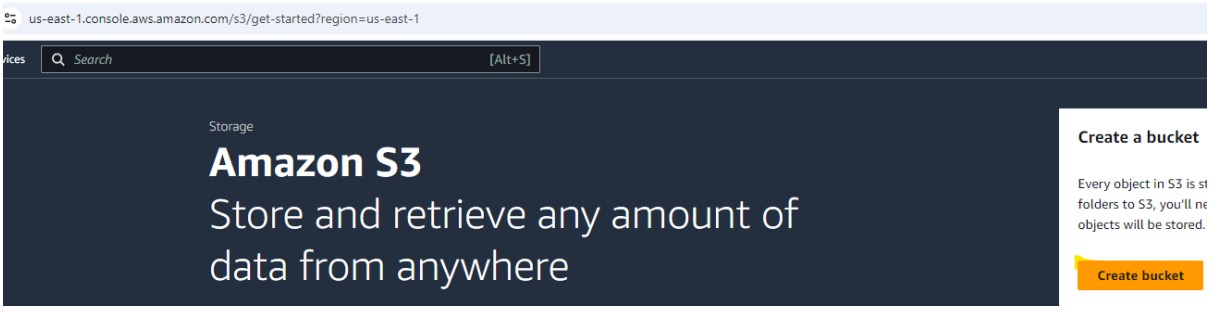


- **Now we will create S3 Bucket where we need to upload our backup**

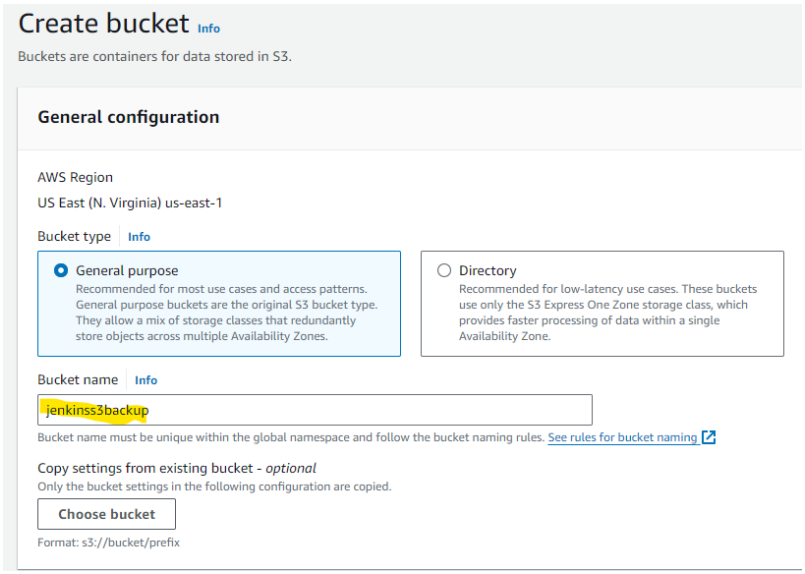
✓ On the AWS Console, go to console and search S3 and click on it.

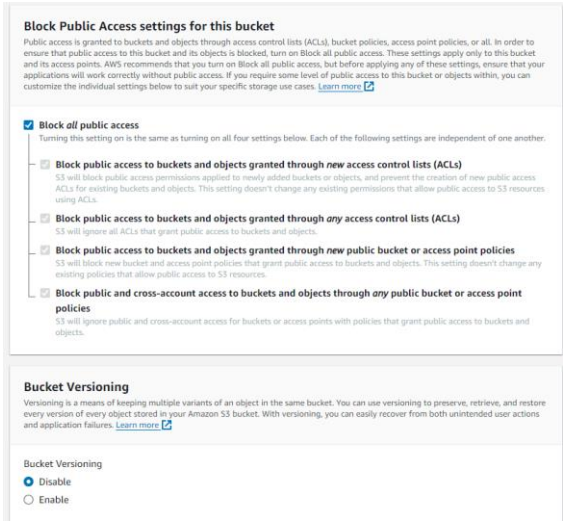


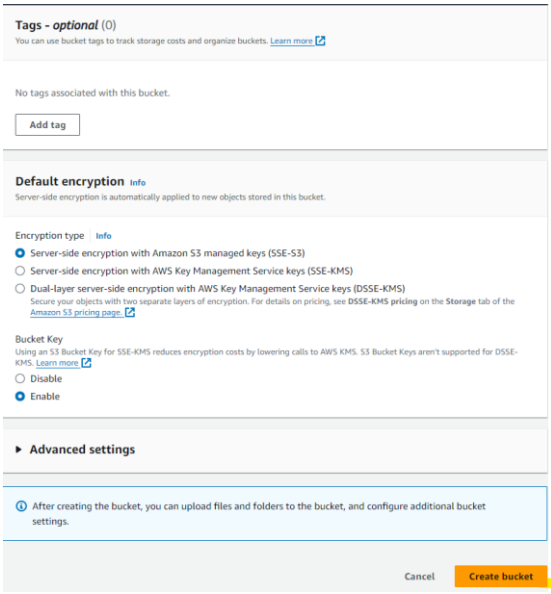
✓ Now Click Create bucket



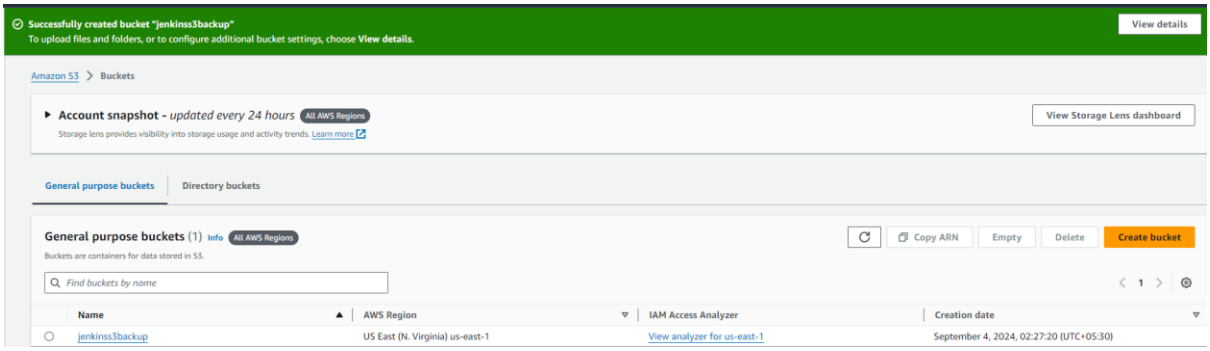
✓ Now under general configuration, keep everything as default and just give a unique name to S3 bucket and click create bucket. In our case, bucket name is jenkinss3backup



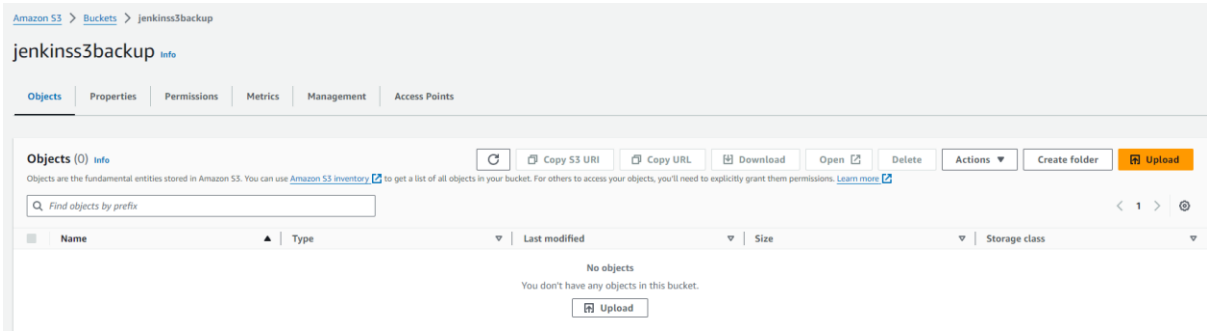




✓ Bucket is created successfully.

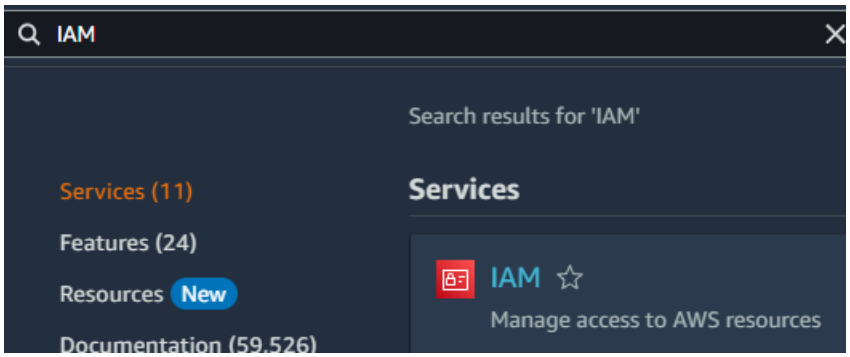


✓ Bucket is blank right now.

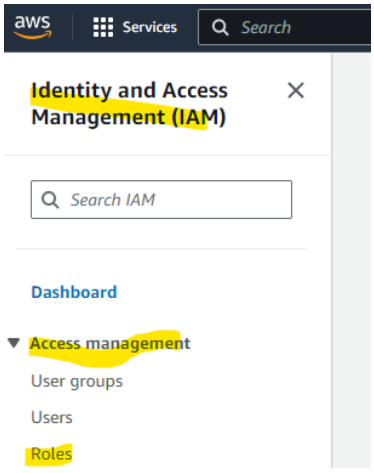


- **Now we will create the IAM Role for S3 full access**

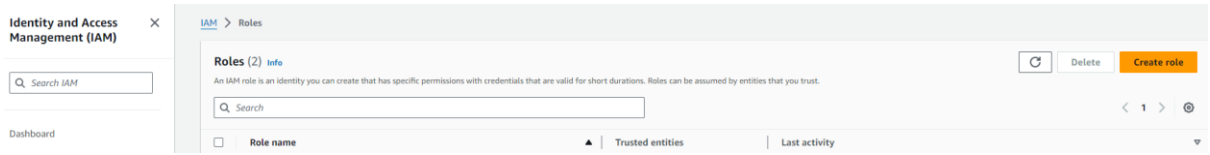
✓ On the AWS Console, go to console and search IAM and click on it.



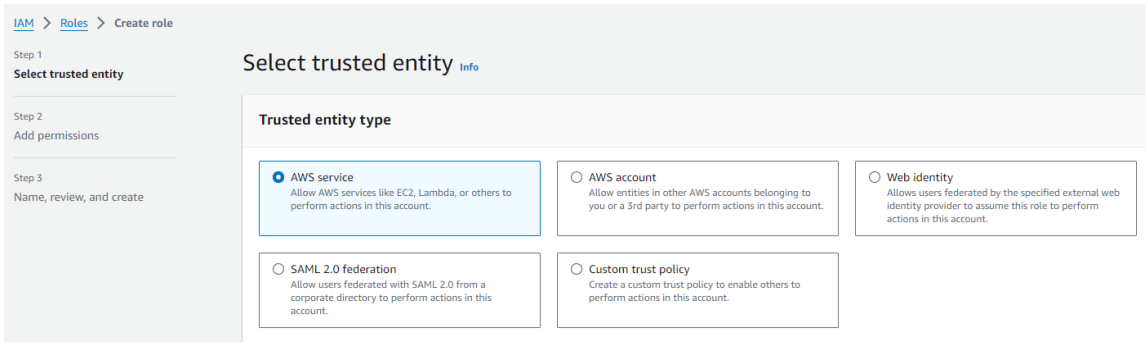
✓ Click on Roles under Access Management.



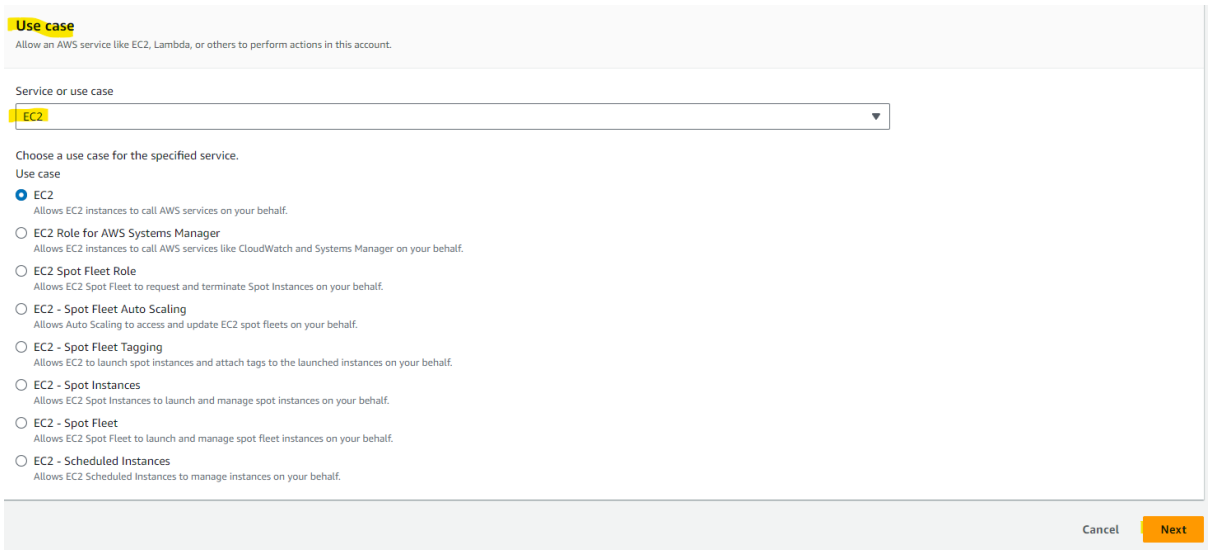
✓ Now Click Create Role.



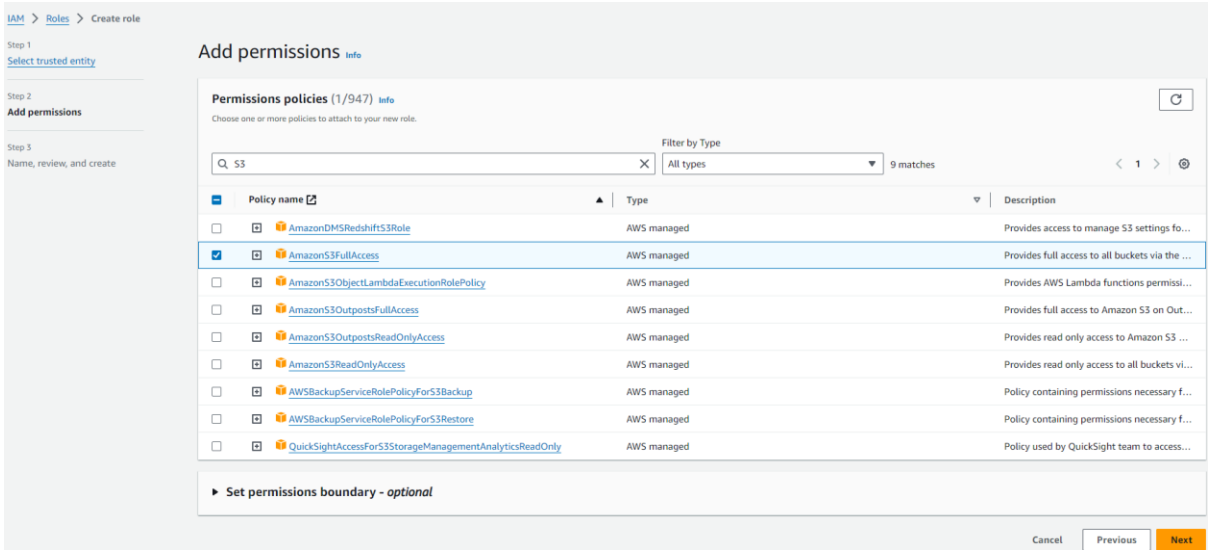
✓ Under Select Trusted Entity > Aws Service



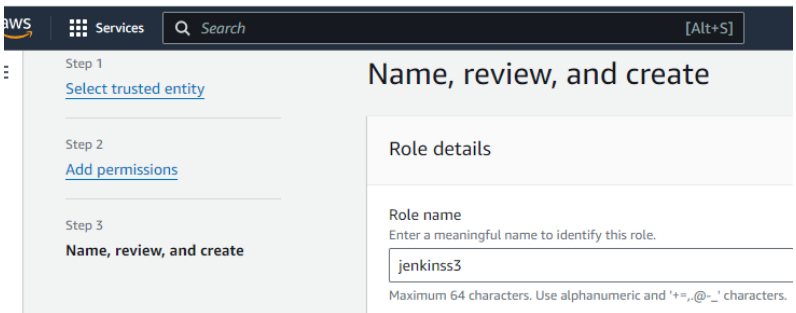
✓ Under Use Case > Drop down to EC2 and click Next

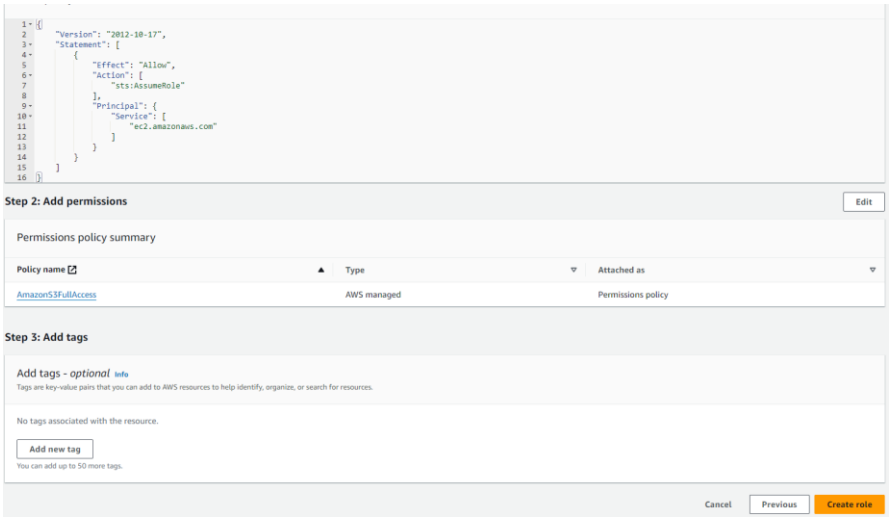


✓ Add Permissions > Search for S3 > AmazonS3FullAccess and click Next

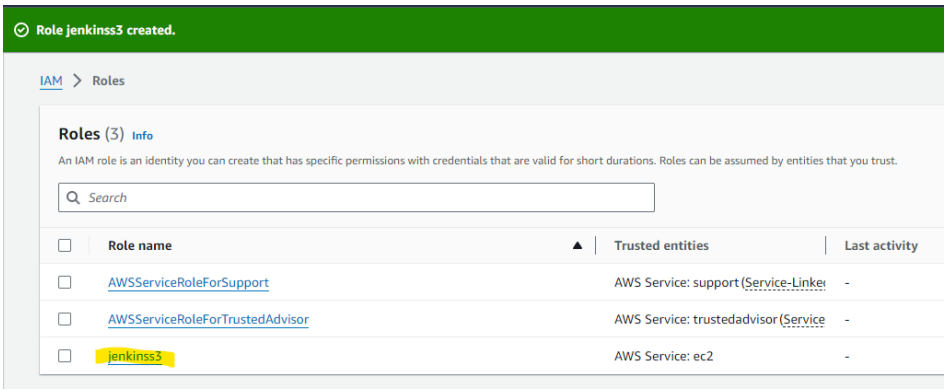


✓ Give the Role Name. In our case, It will be jenkinss3 and click create role.





✓ Role is created.



- **Now we will take the backup of first jenkins instance.**

✓ First of all, we need to stop the jenkins service by using the below command

systemctl stop jenkins

```
root@ip-172-31-88-11:/home/ubuntu# systemctl stop jenkins
root@ip-172-31-88-11:/home/ubuntu#
```

✓ Now, the most important folder which has all the configs, secrets, jobs and builds is the home path of jenkins /var/lib/jenkins

cd /var/lib/jenkins

ls -ltr

```
root@ip-172-31-88-11:/home/ubuntu# cd /var/lib/jenkins/
root@ip-172-31-88-11:/var/lib/jenkins#
root@ip-172-31-88-11:/var/lib/jenkins#
root@ip-172-31-88-11:/var/lib/jenkins#
root@ip-172-31-88-11:/var/lib/jenkins#
root@ip-172-31-88-11:/var/lib/jenkins#
root@ip-172-31-88-11:/var/lib/jenkins#
root@ip-172-31-88-11:/var/lib/jenkins# ls -ltr
total 96
-rw-r--r--  1 jenkins jenkins    64 Sep  3 20:08 secret.key
-rw-r--r--  1 jenkins jenkins     0 Sep  3 20:08 secret.key.not-so-secret
-rw-r--r--  1 jenkins jenkins   156 Sep  3 20:08 hudson.model.UpdateCenter.xml
-rw-r--r--  1 jenkins jenkins   171 Sep  3 20:08 jenkins.telemetry.Correlator.xml
drwxr-xr-x  2 jenkins jenkins  4096 Sep  3 20:08 userContent
-rw-r--r--  1 jenkins jenkins  1037 Sep  3 20:08 nodeMonitors.xml
drwxr-xr-x  3 jenkins jenkins  4096 Sep  3 20:08 users
-rw-r--r--  1 jenkins jenkins  1660 Sep  3 20:08 config.xml
-rw-------  1 jenkins jenkins  1680 Sep  3 20:12 identity.key.enc
-rw-r--r--  1 jenkins jenkins   370 Sep  3 20:12 hudson.plugins.git.GitTool.xml
-rw-r--r--  1 jenkins jenkins   182 Sep  3 20:13 jenkins.model.JenkinsLocationConfiguration.xml
-rw-r--r--  1 jenkins jenkins     7 Sep  3 20:13 jenkins.install.UpgradeWizard.state
-rw-r--r--  1 jenkins jenkins     7 Sep  3 20:13 jenkins.install.InstallUtil.lastExecVersion
drwxr-xr-x  4 jenkins jenkins  4096 Sep  3 20:34 jobs
-rw-r--r--  1 jenkins jenkins   504 Sep  3 20:39 org.jenkinsci.plugins.resourcedisposer.AsyncResourceDisposer.xml
drwxr-xr-x  4 jenkins jenkins  4096 Sep  3 20:39 workspace
-rw-r--r--  1 jenkins jenkins    46 Sep  3 20:39 org.jenkinsci.plugins.workflow.flow.FlowExecutionList.xml
drwx------  2 jenkins jenkins  4096 Sep  3 20:40 secrets
drwxr-xr-x  3 jenkins jenkins  4096 Sep  3 20:42 logs
drwxr-xr-x 91 jenkins jenkins 12288 Sep  3 20:46 plugins
drwxr-xr-x  2 jenkins jenkins  4096 Sep  3 20:46 updates
-rw-r--r--  1 jenkins jenkins   313 Sep  3 20:46 hudson.plugins.sonar.SonarGlobalConfiguration.xml
-rw-r--r--  1 jenkins jenkins   258 Sep  3 21:13 queue.xml
root@ip-172-31-88-11:/var/lib/jenkins#
```

- ✓ Now we compress all the files & directories under /var/lib/jenkins by creating a tar file.

tar -zcvf <file-name> /var/lib/jenkins/

In our case, it's as below

tar -zcvf jenkins-backup.tar.gz /var/lib/jenkins/

```
root@ip-172-31-88-11:/home/ubuntu# tar -zcvf jenkins-backup.tar.gz /var/lib/jenkins/
tar: Removing leading `/' from member names
/var/lib/jenkins/
/var/lib/jenkins/org.jenkinsci.plugins.resourcedisposer.AsyncResourceDisposer.xml
/var/lib/jenkins/secrets/
/var/lib/jenkins/secrets/hudson.model.Job.serverCookie
/var/lib/jenkins/secrets/jenkins.model.Jenkins.crumbSalt
/var/lib/jenkins/secrets/org.jenkinsci.plugins.workflow.log.ConsoleAnnotators.consoleAnnotator
/var/lib/jenkins/secrets/initialAdminPassword
/var/lib/jenkins/secrets/master.key
/var/lib/jenkins/secrets/org.jenkinsci.main.modules.instance_identity.InstanceIdentity.KEY
/var/lib/jenkins/secrets/hudson.console.AnnotatedLargeText.consoleAnnotator
/var/lib/jenkins/secrets/hudson.util.Secret
/var/lib/jenkins/secrets/hudson.console.ConsoleNote.MAC
/var/lib/jenkins/hudson.plugins.sonar.SonarGlobalConfiguration.xml
/var/lib/jenkins/plugins/
/var/lib/jenkins/plugins/workflow-support/
/var/lib/jenkins/plugins/workflow-support/WEB-INF/
/var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/
/var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/jboss-marshalling-river-2.1.3.Final.jar
/var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/jboss-marshalling-2.1.3.Final.jar
/var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/workflow-support.jar
/var/lib/jenkins/plugins/workflow-support/WEB-INF/licenses.xml
/var/lib/jenkins/plugins/workflow-support/.timestamp2
/var/lib/jenkins/plugins/workflow-support/META-INF/
/var/lib/jenkins/plugins/workflow-support/META-INF/MANIFEST.MF
/var/lib/jenkins/plugins/workflow-support/META-INF/maven/
/var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/
/var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/workflow-support/
/var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/workflow-support/pom.properties
/var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/workflow-support/pom.xml
/var/lib/jenkins/plugins/workflow-multibranch.jpi
/var/lib/jenkins/plugins/caffeine-api.jpi
/var/lib/jenkins/plugins/cloudbees-folder.jpi
```

✓ All the files & directories under /var/lib/jenkins are compressed and tar file is created jenkins-backup.tar.gz

```
root@ip-172-31-88-11:/home/ubuntu# ls -ltr
total 301648
drwxr-xr-x 4 root root      4096 Sep  3 19:10 Devops_Notes_Assignments
-rw-r--r-- 1 root root       767 Sep  3 19:19 jenkins.sh
-rw-r--r-- 1 root root 308874325 Sep  3 21:21 jenkins-backup.tar.gz
root@ip-172-31-88-11:/home/ubuntu#
```

- **Now we will install AWS CLI**

✓ Get the zip file of aws cli using the below command.

curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"

```
root@ip-172-31-88-11:/home/ubuntu# curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 58.0M  100 58.0M    0     0   107M      0 --:--:-- --:--:-- --:--:--  107M
```

✓ Now we will unzip the file, but before that we need to install **unzip** using the below command.

apt install unzip -y

```
root@ip-172-31-88-11:/home/ubuntu# apt install unzip -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  zip
The following NEW packages will be installed:
  unzip
0 upgraded, 1 newly installed, 0 to remove and 2 not upgraded.
Need to get 175 kB of archives.
After this operation, 384 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 unzip amd64 6.0-28ubuntu4 [175 kB]
Fetched 175 kB in 0s (11.1 MB/s)
Selecting previously unselected package unzip.
(Reading database ... 121174 files and directories currently installed.)
Preparing to unpack .../unzip_6.0-28ubuntu4_amd64.deb ...
Unpacking unzip (6.0-28ubuntu4) ...
Setting up unzip (6.0-28ubuntu4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
  6.8.0-1012-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.8.0-1014-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...
```

✓ Now we will unzip the file **awscliv2.zip** using the below command

unzip awscliv2.zip

```
root@ip-172-31-88-11:/home/ubuntu# unzip awscliv2.zip
Archive:  awscliv2.zip
   creating: aws/
   creating: aws/dist/
  inflating: aws/README.md
  inflating: aws/THIRD_PARTY_LICENSES
  inflating: aws/install
   creating: aws/dist/awscli/
   creating: aws/dist/cryptography/
   creating: aws/dist/docutils/
   creating: aws/dist/lib-dynload/
  inflating: aws/dist/aws
  inflating: aws/dist/aws_completer
  inflating: aws/dist/libpython3.11.so.1.0
  inflating: aws/dist/_awscrt.abi3.so
  inflating: aws/dist/_cffi_backend.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/_ruamel_yaml.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/libz.so.1
  inflating: aws/dist/liblzma.so.0
  inflating: aws/dist/libbz2.so.1
  inflating: aws/dist/libffi.so.5
  inflating: aws/dist/libsqlite3.so.0
  inflating: aws/dist/base_library.zip
  inflating: aws/dist/lib-dynload/_pickle.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_hashlib.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_sha3.cpython-311-x86_64-linux-gnu.so
```

✓ Once the unzip of the file is done, we need to run the below command to install AWS CLI

./aws/install

```
root@ip-172-31-88-11:/home/ubuntu# ./aws/install
You can now run: /usr/local/bin/aws --version
```

✓ We can verify that AWS CLI installed by typing below command.

aws

```
root@ip-172-31-88-11:/home/ubuntu# aws

usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

  aws help
  aws <command> help
  aws <command> <subcommand> help

aws: error: the following arguments are required: command
```

- **Now we will upload the jenkins-backup.tar.gz using the below command via AWS CLI.**

  aws s3 cp <file-name> s3://<bucket-name>/<file-name>

  In our case

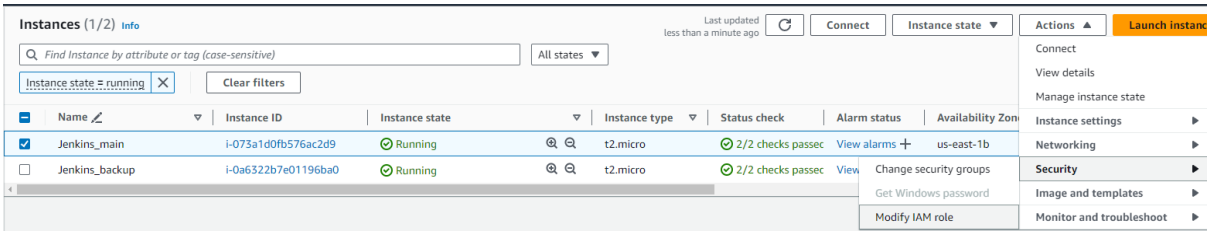
  aws s3 cp jenkins-backup.tar.gz s3://jenkinss3backup/jenkins-backup.tar.gz

```
root@ip-172-31-88-11:/home/ubuntu# aws s3 cp jenkins-backup.tar.gz s3://jenkinss3backup/jenkins-backup.tar.gz
upload failed: ./jenkins-backup.tar.gz to s3://jenkinss3backup/jenkins-backup.tar.gz Unable to locate credentials
```

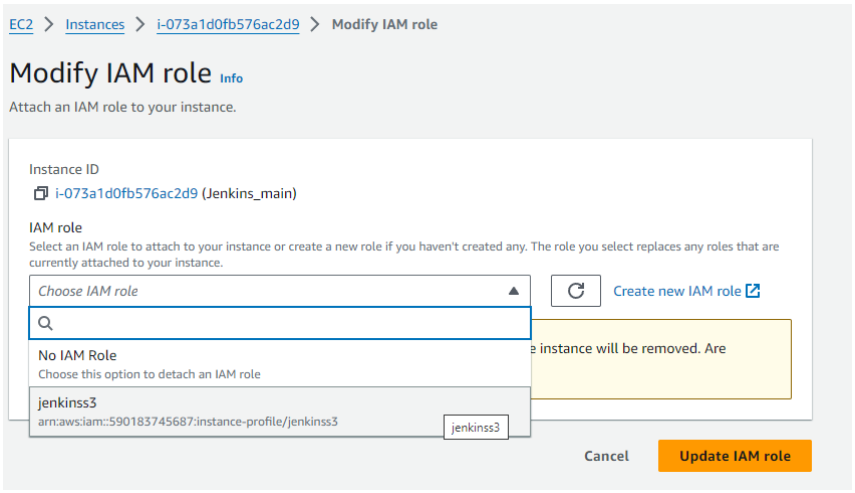
- Upload failed due to credentials were not located.

- **Now we will attach the IAM Role to Jenkins Primary EC2 Instance**

✓ We will go to the below path and modify IAM Role



✓ We will choose the Role from dropdown and select jenkinss3 and click update IAM role



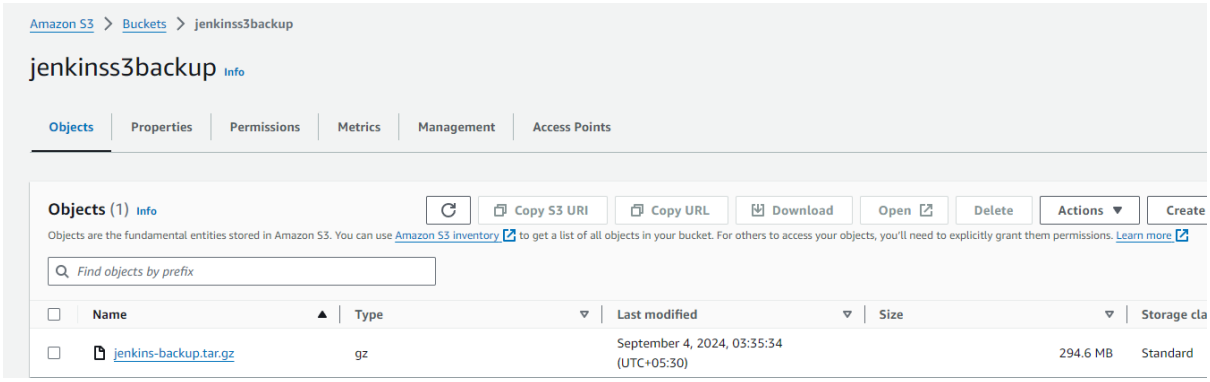
- **Now we will again upload the jenkins-backup.tar.gz using the below command via AWS CLI.**

  aws s3 cp jenkins-backup.tar.gz s3://jenkinss3backup/jenkins-backup.tar.gz

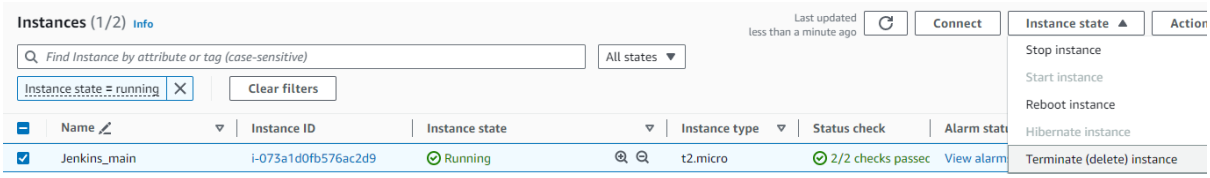
  Now file got uploaded successfully.

```
root@ip-172-31-88-11:/home/ubuntu# aws s3 cp jenkins-backup.tar.gz s3://jenkinss3backup/jenkins-backup.tar.gz
upload: ./jenkins-backup.tar.gz to s3://jenkinss3backup/jenkins-backup.tar.gz
root@ip-172-31-88-11:/home/ubuntu#
```

- Verified on S3 bucket also that upload was successful.



- **Now will terminate the first Jenkins instance.**



- **Now we will go on the Second Jenkins instance for restoring backup of first Jenkins instance.**

✓ We will have install AWS CLI again by following the same method.

Get the zip file of aws cli using the below command.

curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"

```
root@ip-172-31-80-117:/home/ubuntu# curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 58.0M  100 58.0M    0     0   106M      0 --:--:-- --:--:-- --:--:--  106M
root@ip-172-31-80-117:/home/ubuntu#
```

✓ Now we will unzip the file, but before that we need to install **unzip** using the below command.

apt install unzip -y

```
root@ip-172-31-88-11:/home/ubuntu# apt install unzip -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  zip
The following NEW packages will be installed:
  unzip
0 upgraded, 1 newly installed, 0 to remove and 2 not upgraded.
Need to get 175 kB of archives.
After this operation, 384 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 unzip amd64 6.0-28ubuntu4 [175 kB]
Fetched 175 kB in 0s (11.1 MB/s)
Selecting previously unselected package unzip.
(Reading database ... 121174 files and directories currently installed.)
Preparing to unpack .../unzip_6.0-28ubuntu4_amd64.deb ...
Unpacking unzip (6.0-28ubuntu4) ...
Setting up unzip (6.0-28ubuntu4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Pending kernel upgrade!
Running kernel version:
  6.8.0-1012-aws
Diagnostics:
  The currently running kernel version is not the expected kernel version 6.8.0-1014-aws.

Restarting the system to load the new kernel will not be handled automatically, so you should consider rebooting.

Restarting services...
```

✓ Now we will unzip the file **awscliv2.zip** using the below command
unzip awscliv2.zip

```
root@ip-172-31-88-11:/home/ubuntu# unzip awscliv2.zip
Archive:  awscliv2.zip
   creating: aws/
   creating: aws/dist/
  inflating: aws/README.md
  inflating: aws/THIRD_PARTY_LICENSES
  inflating: aws/install
   creating: aws/dist/awscli/
   creating: aws/dist/cryptography/
   creating: aws/dist/docutils/
   creating: aws/dist/lib-dynload/
  inflating: aws/dist/aws
  inflating: aws/dist/aws_completer
  inflating: aws/dist/libpython3.11.so.1.0
  inflating: aws/dist/_awscrt.abi3.so
  inflating: aws/dist/_cffi_backend.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/_ruamel_yaml.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/libz.so.1
  inflating: aws/dist/liblzma.so.0
  inflating: aws/dist/libbz2.so.1
  inflating: aws/dist/libffi.so.5
  inflating: aws/dist/libsqlite3.so.0
  inflating: aws/dist/base_library.zip
  inflating: aws/dist/lib-dynload/_pickle.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_hashlib.cpython-311-x86_64-linux-gnu.so
  inflating: aws/dist/lib-dynload/_sha3.cpython-311-x86_64-linux-gnu.so
```

✓ Once the unzip of the file is done, we need to run the below command to install AWS CLI

./aws/install

```
root@ip-172-31-88-11:/home/ubuntu# ./aws/install
You can now run: /usr/local/bin/aws --version
```

✓ We can verify that AWS CLI installed by typing below command.

aws

```
root@ip-172-31-88-11:/home/ubuntu# aws

usage: aws [options] <command> <subcommand> [<subcommand> ...] [parameters]
To see help text, you can run:

  aws help
  aws <command> help
  aws <command> <subcommand> help

aws: error: the following arguments are required: command
```

✓ we need to stop the jenkins service by using the below command

systemctl stop jenkins

```
root@ip-172-31-88-11:/home/ubuntu# systemctl stop jenkins
root@ip-172-31-88-11:/home/ubuntu#
```

✓ We need to remove the home path /var/lib/jenkins of current jenkins setup

rm –rf /var/lib/jenkins

```
root@ip-172-31-80-117:/home/ubuntu# rm -rf /var/lib/jenkins/
root@ip-172-31-80-117:/home/ubuntu#
```

- **Now we will download the jenkins-backup.tar.gz using the below command via AWS CLI.**

aws s3 cp s3://bucketname/filename filename

In our case

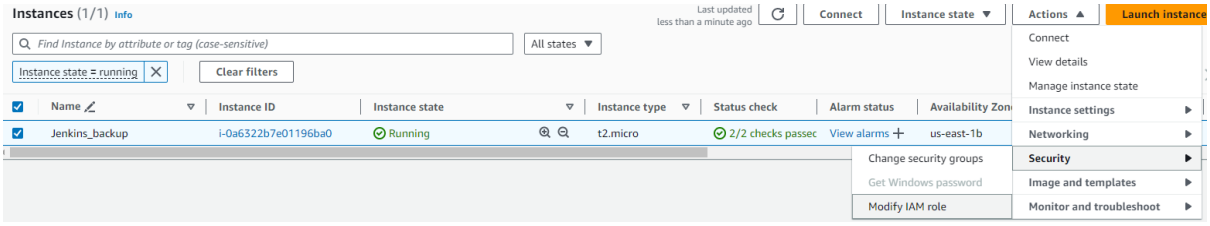
aws s3 cp s3://jenkinss3backup/jenkins-backup.tar.gz jenkins-backup.tar.gz

```
root@ip-172-31-80-117:/home/ubuntu# aws s3 cp s3://jenkinss3backup/jenkins-backup.tar.gz jenkins-backup.tar.gz
fatal error: Unable to locate credentials
```

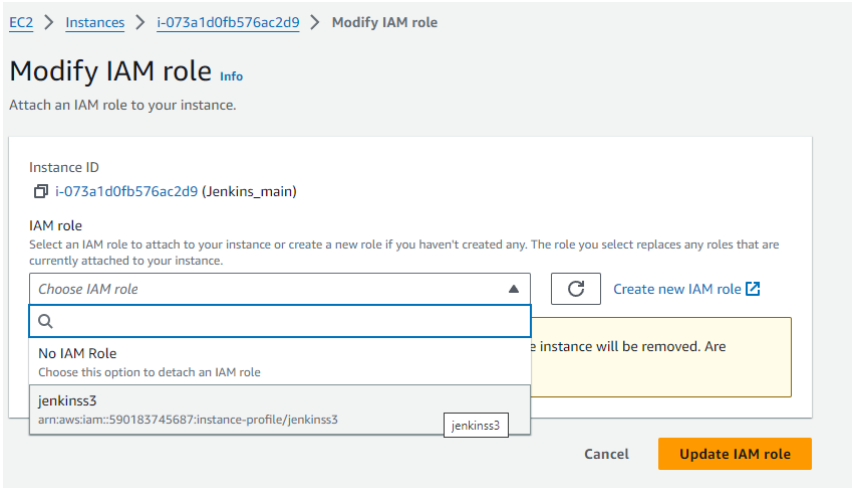
- Unable to locate credentials.

- **Now we will attach the IAM Role to Jenkins Secondary EC2 Instance**

✓ We will go to the below path and modify IAM Role



✓ We will choose the Role from dropdown and select jenkinss3 and click update IAM role



- **Now we will again download the jenkins-backup.tar.gz using the below command via AWS CLI.**

aws s3 cp s3://jenkinss3backup/jenkins-backup.tar.gz jenkins-backup.tar.gz

File downloaded successfully on the Instance

```
root@ip-172-31-80-117:/home/ubuntu# aws s3 cp s3://jenkinss3backup/jenkins-backup.tar.gz jenkins-backup.tar.gz
download: s3://jenkinss3backup/jenkins-backup.tar.gz to ./jenkins-backup.tar.gz
root@ip-172-31-80-117:/home/ubuntu#
```

✓ File is present on the location.

```
root@ip-172-31-80-117:/home/ubuntu# ls -ltr
total 361056
drwxr-xr-x 3 root root      4096 Aug 30 18:21 aws
drwxr-xr-x 4 root root      4096 Sep  3 19:10 Devops_Notes_Assignments
-rw-r--r-- 1 root root 308874325 Sep  3 22:05 jenkins-backup.tar.gz
-rw-r--r-- 1 root root  60830141 Sep  3 22:14 awscliv2.zip
```

✓ Now will untar the file, so that it creates /etc/var/jenkins directory and copies all the backup of the primary instance.

tar -zxvf <file-name> -C /

In our case

tar -zxvf jenkins-backup.tar.gz -C /

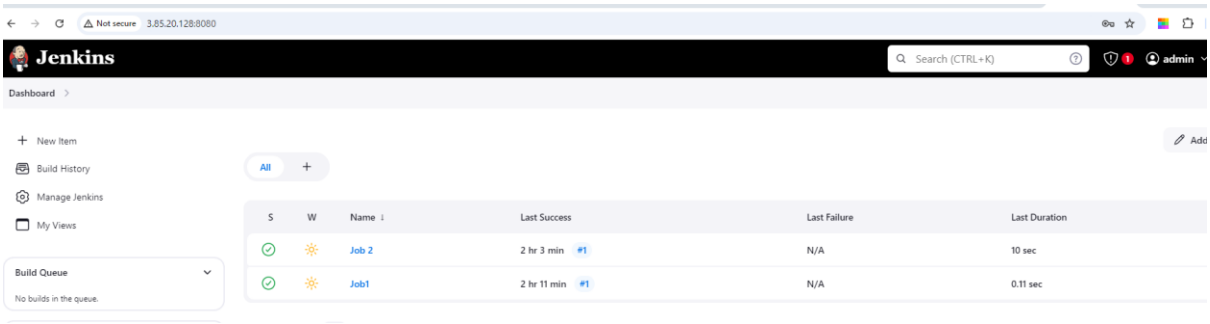
```
root@ip-172-31-80-117:/home/ubuntu# tar -zxvf jenkins-backup.tar.gz -C /
var/lib/jenkins/
var/lib/jenkins/org.jenkinsci.plugins.resourcedisposer.AsyncResourceDisposer.xml
var/lib/jenkins/secrets/
var/lib/jenkins/secrets/hudson.model.Job.serverCookie
var/lib/jenkins/secrets/jenkins.model.Jenkins.crumbSalt
var/lib/jenkins/secrets/org.jenkinsci.plugins.workflow.log.ConsoleAnnotators.consoleAnnotator
var/lib/jenkins/secrets/initialAdminPassword
var/lib/jenkins/secrets/master.key
var/lib/jenkins/secrets/org.jenkinsci.main.modules.instance_identity.InstanceIdentity.KEY
var/lib/jenkins/secrets/hudson.console.AnnotatedLargeText.consoleAnnotator
var/lib/jenkins/secrets/hudson.util.Secret
var/lib/jenkins/secrets/hudson.console.ConsoleNote.MAC
var/lib/jenkins/hudson.plugins.sonar.SonarGlobalConfiguration.xml
var/lib/jenkins/plugins/
var/lib/jenkins/plugins/workflow-support/
var/lib/jenkins/plugins/workflow-support/WEB-INF/
var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/
var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/jboss-marshalling-river-2.1.3.Final.jar
var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/jboss-marshalling-2.1.3.Final.jar
var/lib/jenkins/plugins/workflow-support/WEB-INF/lib/workflow-support.jar
var/lib/jenkins/plugins/workflow-support/WEB-INF/licenses.xml
var/lib/jenkins/plugins/workflow-support/.timestamp2
var/lib/jenkins/plugins/workflow-support/META-INF/
var/lib/jenkins/plugins/workflow-support/META-INF/MANIFEST.MF
var/lib/jenkins/plugins/workflow-support/META-INF/maven/
var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/
var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/workflow-support/
var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/workflow-support/pom.properties
var/lib/jenkins/plugins/workflow-support/META-INF/maven/org.jenkins-ci.plugins.workflow/workflow-support/pom.xml
var/lib/jenkins/plugins/workflow-multibranch.jpi
var/lib/jenkins/plugins/caffeine-api.jpi
var/lib/jenkins/plugins/cloudbees-folder.jpi
var/lib/jenkins/plugins/junit.jpi
```

✓ Now we will start jenkins using the below command.

systemctl start jenkins

```
root@ip-172-31-80-117:/home/ubuntu# systemctl start jenkins
root@ip-172-31-80-117:/home/ubuntu#
```

✓ Now we will verify, if we are able to login to the Jenkins using the credentials which we set for Jenkins Primary Instance.



✓ We are able to login and we also see the jobs created in Jenkins Primary Instance.

- So, Jenkins Backup and Restore have been completed successfully.

9) How to Secure Your Jenkins Server with HTTPS.

- **Before starting, we need to have the following prerequisites**

  - **A Jenkins Server Installed on EC2 Instance**
  - **Open SSL Installed on the same EC2 Instance**

- A EC2 Instance is ready with Jenkins Server installed on it using the below shell script

Jenkins_Installation_Configuration_script.sh

- **Open SSL is installed using the below command.**

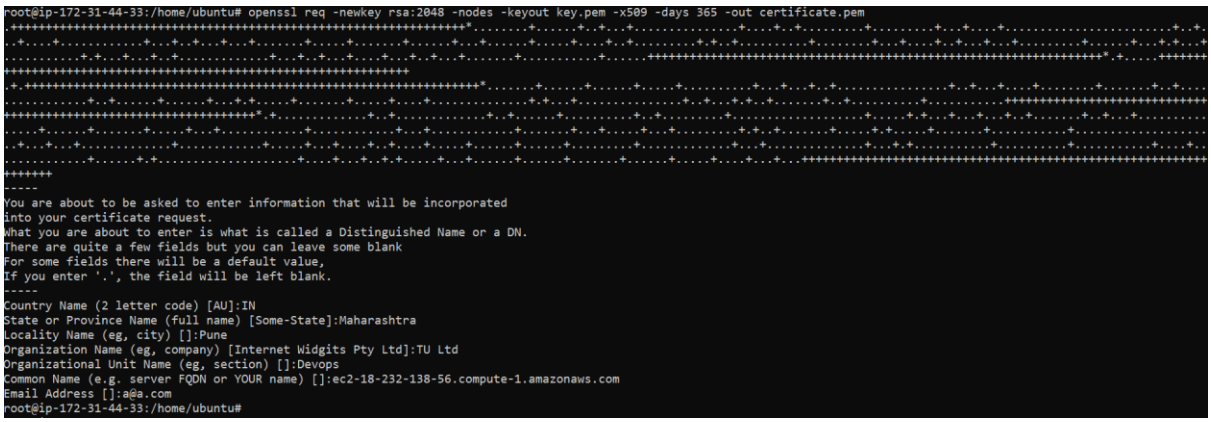
  apt install openssl -y

```
root@ip-172-31-44-33:/home/ubuntu# apt install openssl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
openssl is already the newest version (3.0.13-0ubuntu3.4).
openssl set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
```

- **Generate SSL Certificate**

  - The first step is to generate a self-signed SSL certificate using OpenSSL. Execute the following command: and follow the steps in below screenshot

    openssl req -newkey rsa:2048 -nodes -keyout key.pem -x509 -days 365 -out certificate.pem



- **Create PKCS12 File**

  - Now we will create a PKCS12 file from the generated key and certificate: Set an export password when prompted, verify it.

    openssl pkcs12 -inkey key.pem -in certificate.pem -export -out certificate.p12

```
root@ip-172-31-44-33:/home/ubuntu# openssl pkcs12 -inkey key.pem -in certificate.pem -export -out certificate.p12
Enter Export Password:
Verifying - Enter Export Password:
```

- **Import Certificate to Keystore**

  - Now Import the PKCS12 file into a Java keystore (JKS) using the keytool command:

    You'll need to enter the destination keystore password and the source keystore password.

keytool -importkeystore -srckeystore ./certificate.p12 -srcstoretype pkcs12 -destkeystore jenkinsbuilddevops.jks -deststoretype JKS

```
root@ip-172-31-16-10:/home/ubuntu# keytool -importkeystore -srckeystore ./certificate.p12 -srcstoretype pkcs12 -destkeystore jenkinsbuilddevops.jks -deststoretype JKS
Importing keystore ./certificate.p12 to jenkinsbuilddevops.jks...
Enter destination keystore password:
Re-enter new password:
Enter source keystore password:
Entry for alias 1 successfully imported.
Import command completed:  1 entries successfully imported, 0 entries failed or cancelled

Warning:
The JKS keystore uses a proprietary format. It is recommended to migrate to PKCS12 which is an industry standard format using "keytool -importkeystore -srckeystore jenkinsb
uilddevops.jks -destkeystore jenkinsbuilddevops.jks -deststoretype pkcs12".
root@ip-172-31-16-10:/home/ubuntu#
```

- **Move Keystore to Jenkins Directory**

  - Now Move keystore to jenkins directory. Copy the generated keystore file to the Jenkins directory:

    cp -pav jenkinsbuilddevops.jks /var/lib/jenkins/

```
root@ip-172-31-44-33:/home/ubuntu# cp -pav jenkinsbuilddevops.jks /var/lib/jenkins/
'jenkinsbuilddevops.jks' -> '/var/lib/jenkins/jenkinsbuilddevops.jks'
root@ip-172-31-44-33:/home/ubuntu#
```

  - Set Jenkins HTTPS Configuration**:** Modify the Jenkins service configuration to enable HTTPS. Edit the Jenkins service file:

    vi /usr/lib/systemd/system/jenkins.service

```
root@ip-172-31-44-33:/home/ubuntu# vi /usr/lib/systemd/system/jenkins.service
root@ip-172-31-44-33:/home/ubuntu#
```

    Change the below line

    Environment="JENKINS_PORT=8080"

```
# add the CAP_NET_BIND_SERVICE ca
# directive below.
Environment="JENKINS_PORT=8080"
```

Environment="JENKINS_HTTPS_PORT=8090"
Environment="JENKINS_HTTPS_KEYSTORE=/var/lib/jenkins/jenkinsbuilddevops.jks"
Environment="JENKINS_HTTPS_KEYSTORE_PASSWORD=abc123"

```
Environment="JENKINS_HTTPS_PORT=8090"
Environment="JENKINS_HTTPS_KEYSTORE=/var/lib/jenkins/jenkinsbuilddevops.jks"
Environment="JENKINS_HTTPS_KEYSTORE_PASSWORD=abc123"
```

- Reload the systemd daemon and restart the Jenkins service:

systemctl daemon-reload

systemctl restart jenkins

```
root@ip-172-31-44-33:/home/ubuntu# systemctl daemon-reload
root@ip-172-31-44-33:/home/ubuntu#
root@ip-172-31-44-33:/home/ubuntu#
root@ip-172-31-44-33:/home/ubuntu#
root@ip-172-31-44-33:/home/ubuntu# systemctl restart jenkins
root@ip-172-31-44-33:/home/ubuntu#
```

- Now we will verify the https connection on the new port for Jenkins i.e. port 8090