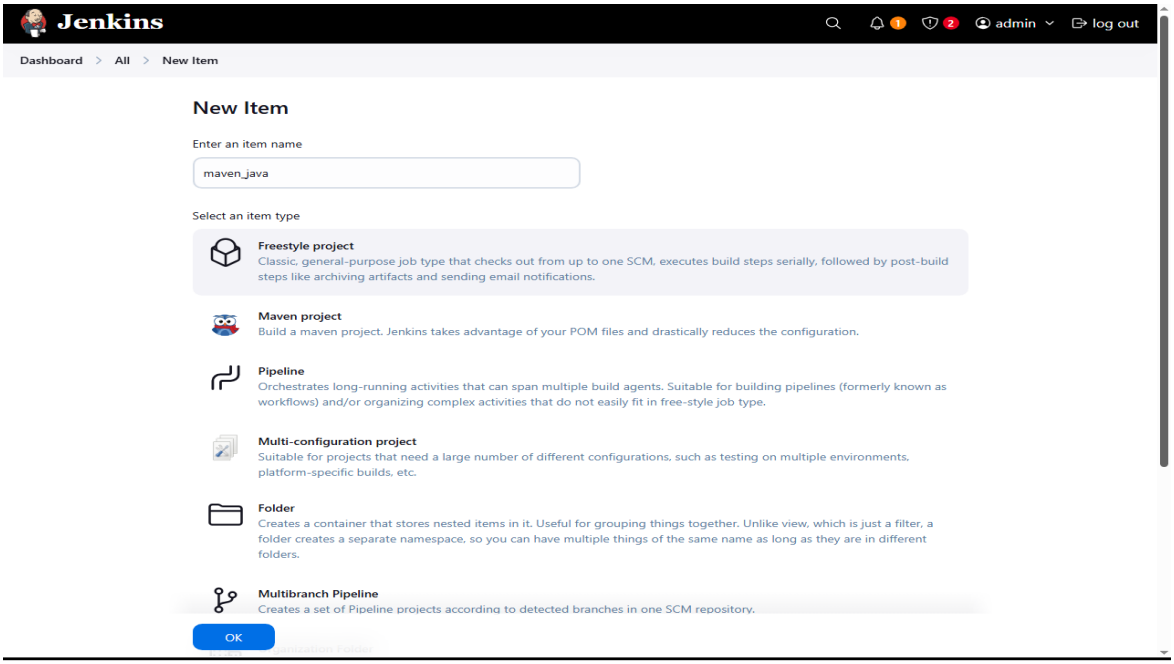
8. Jenkins Automation

Steps for MavenJava Automation

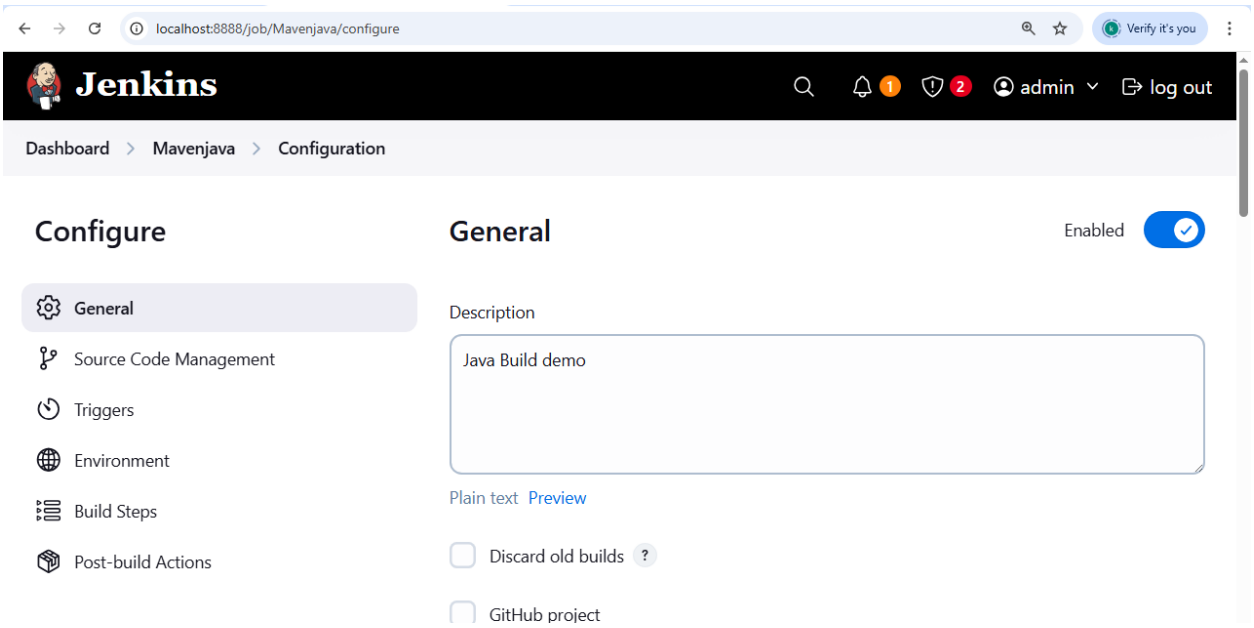
Step 1: Open Jenkins (localhost:8888)

Click on "New Item" (left side menu) and name it as maven_java > select freestyle project > click on “OK”

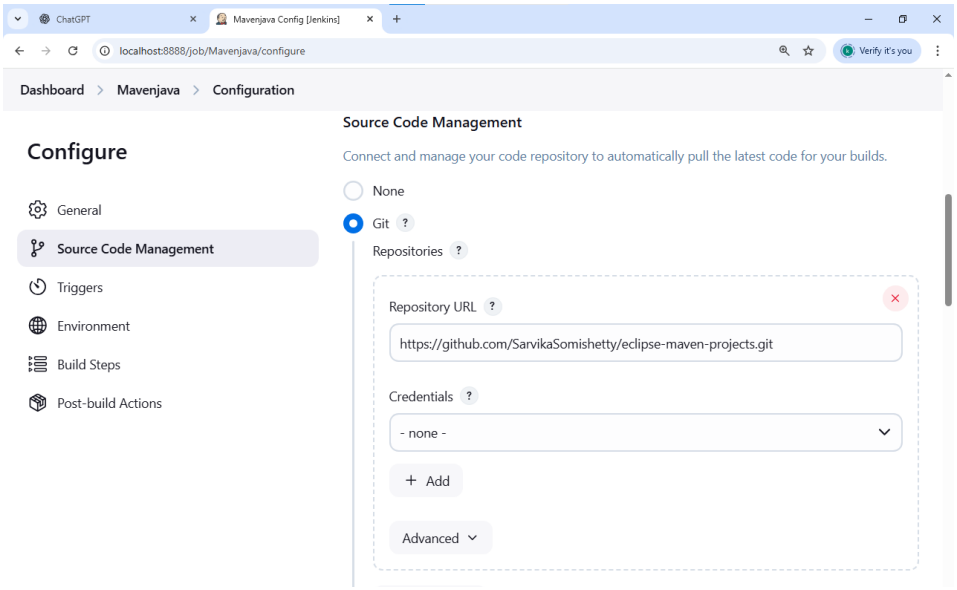


Step 2: Configuration of maven_java project

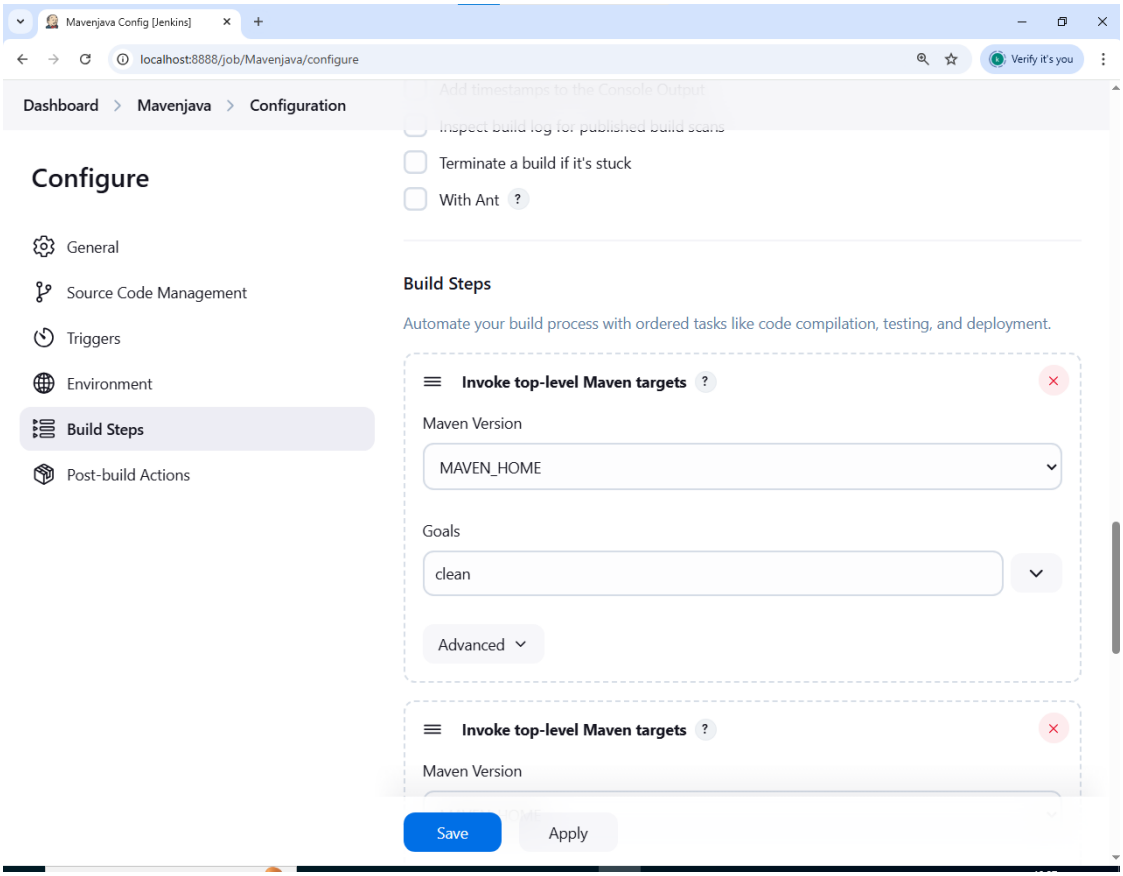
Give the description



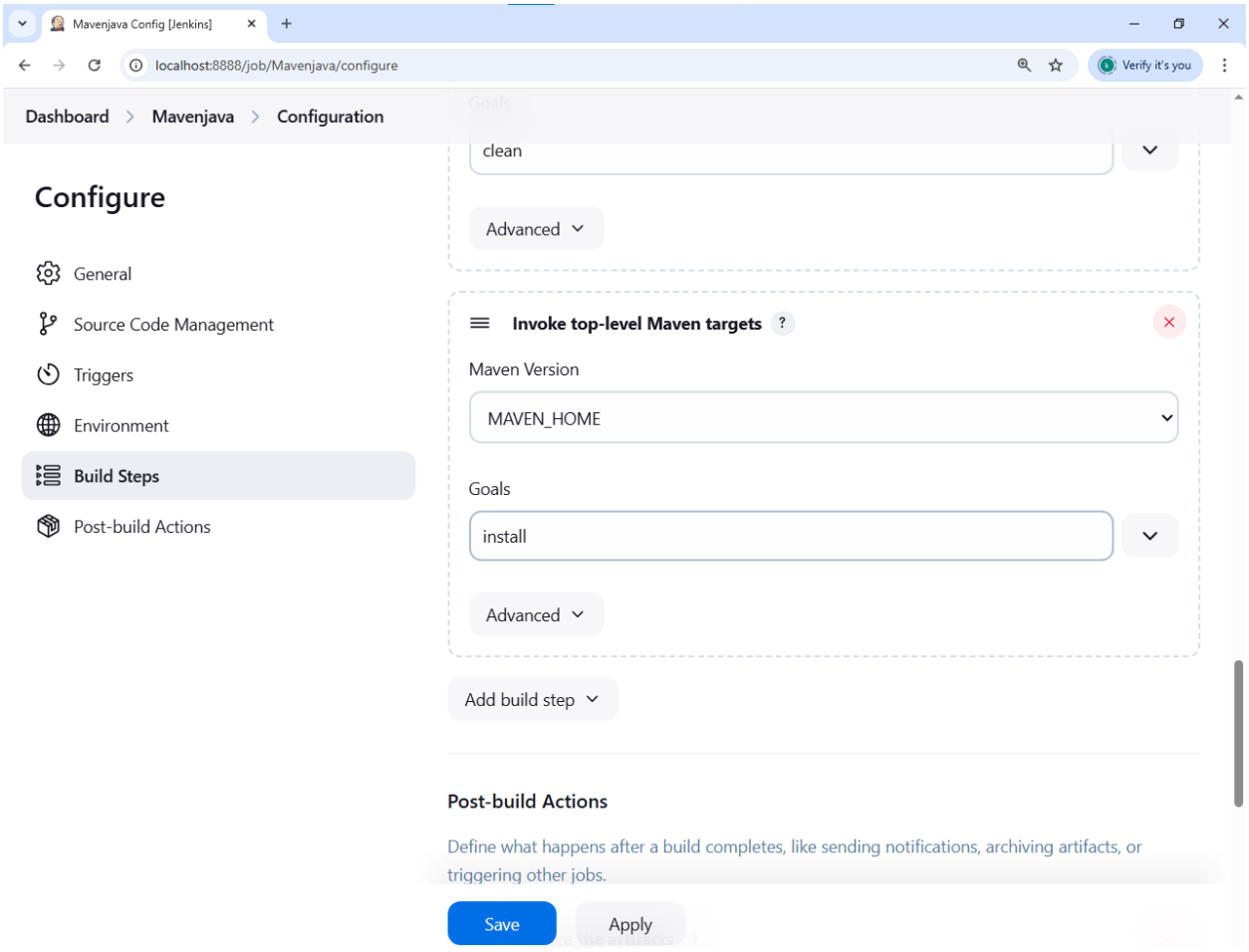
In the source code management select git and give the git repo link



In the build steps click on add build step > give maven version as MAVEN_HOME > select invoke top-level maven targets > goals as clean



In the build steps click on add build step > give maven version as MAVEN_HOME > select invoke top-level maven targets > goals as install

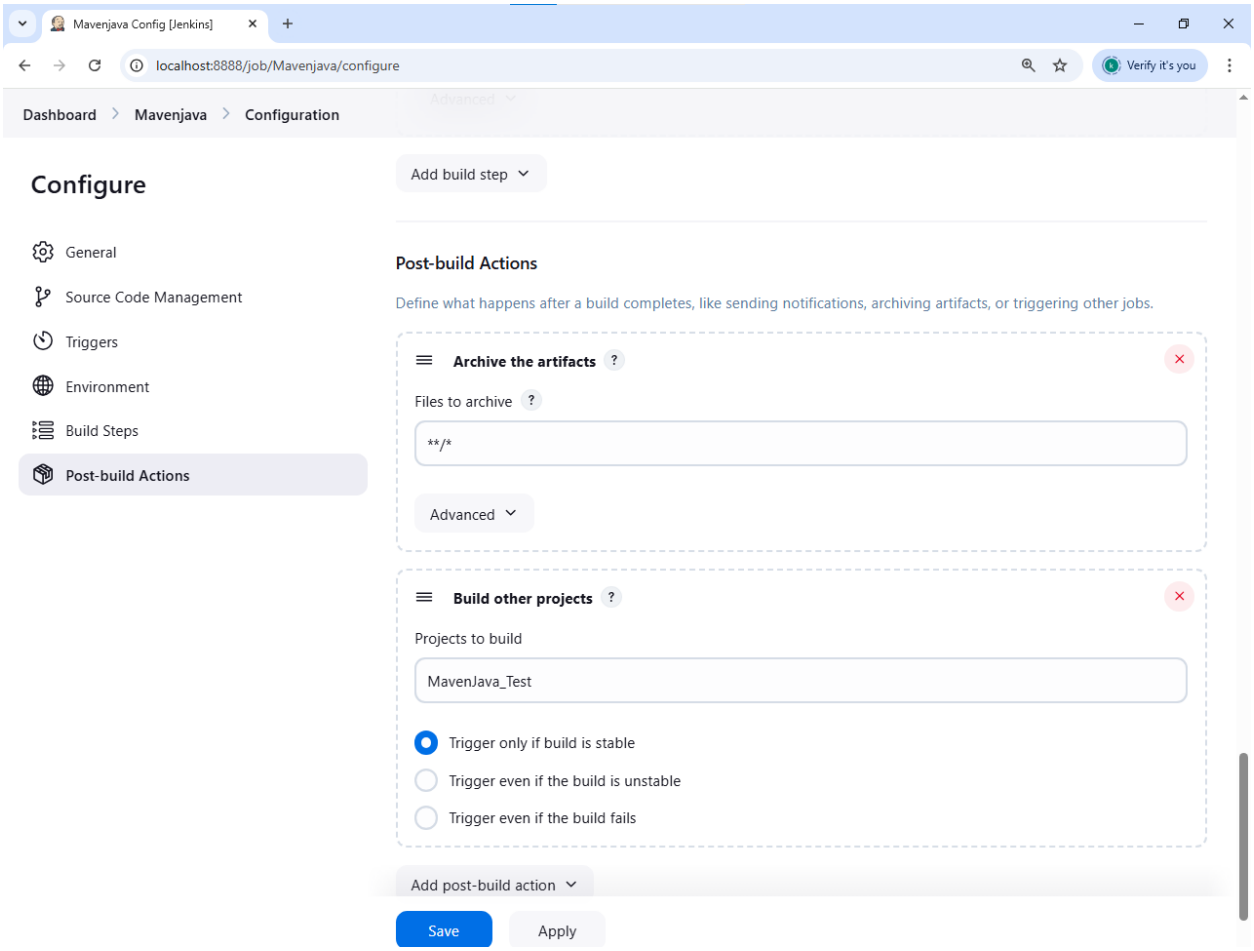


In the post build actions > click on add post build action > select the archive the artifacts > in the file to archive give “**/*”

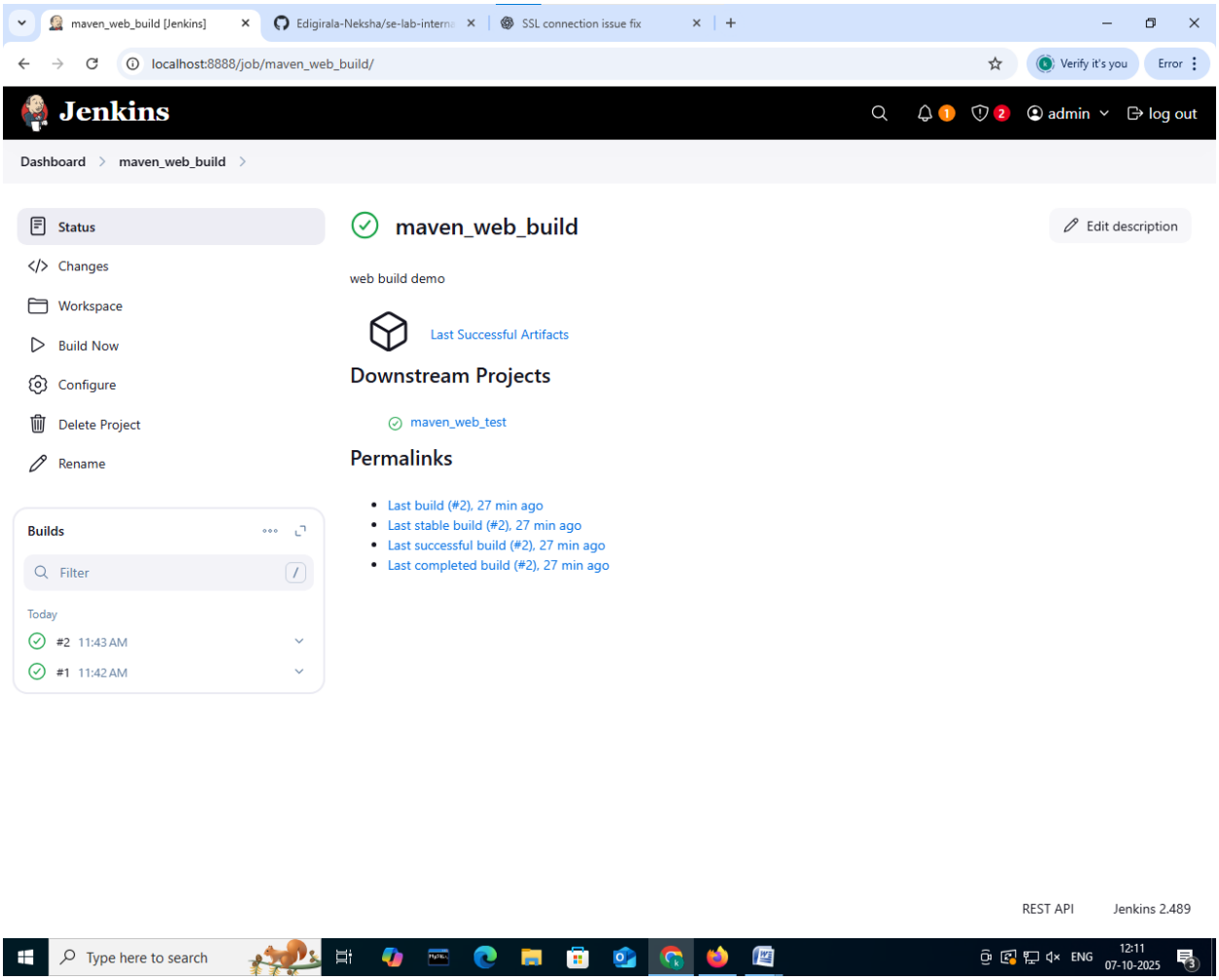
For the second post build action,

In the post build actions > click on add post build action >select build other projects > give projects to build as MavenJava_Test

Click on apply and save

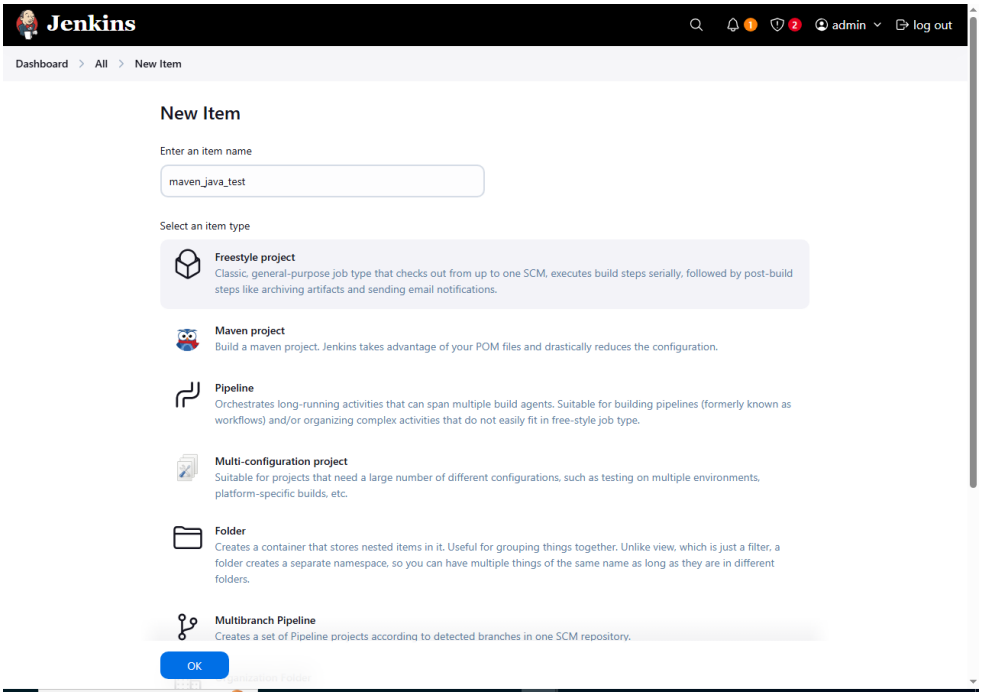


If the build is success:



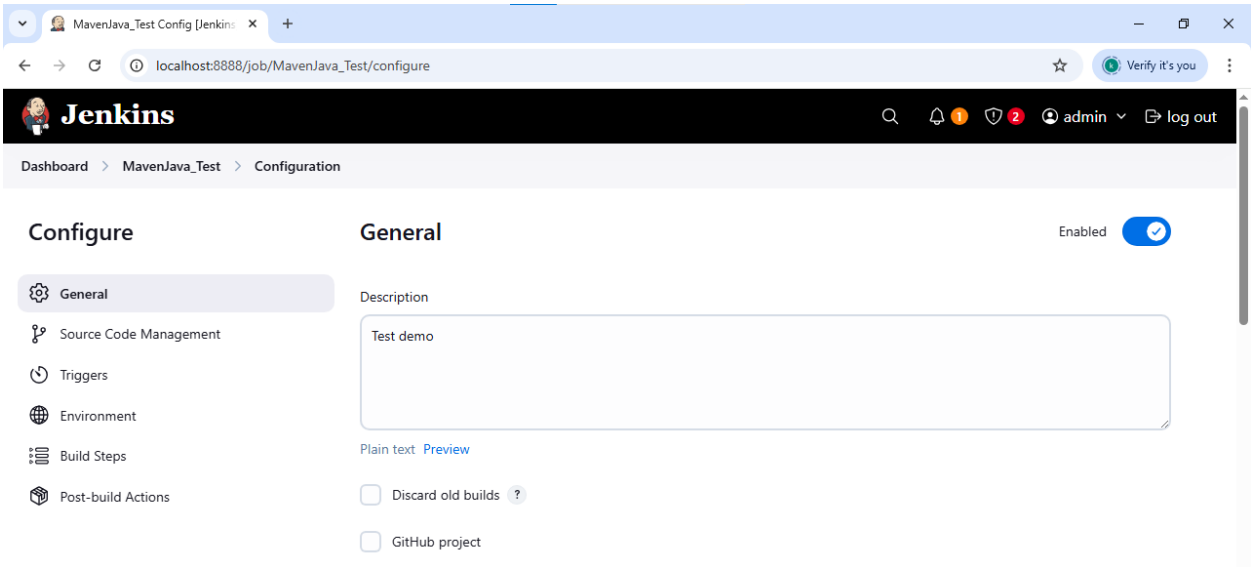
Step 3: Create Freestyle Project (e.g., MavenJava_Test)

Click on new item> give item name as mavaen_java_test or MavenJava_Test and select free style project and click ok

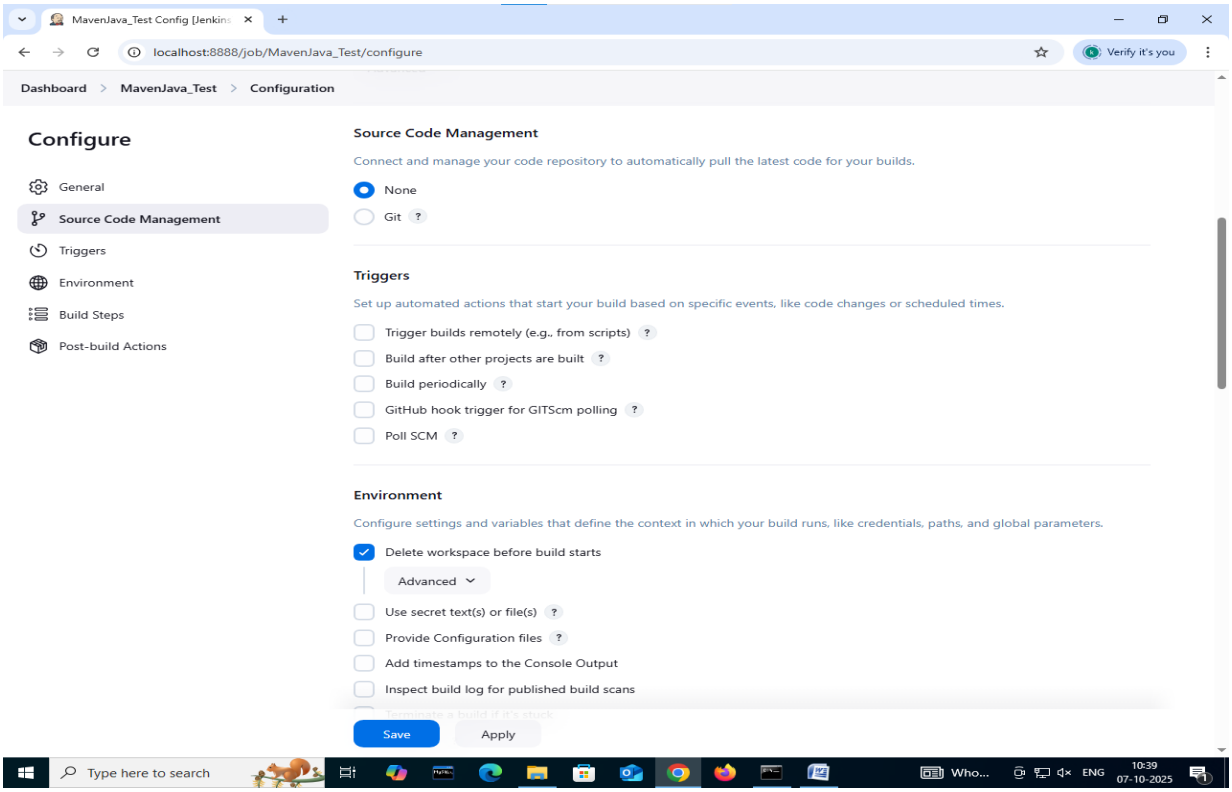


Step 4: Configuration of maven_java project

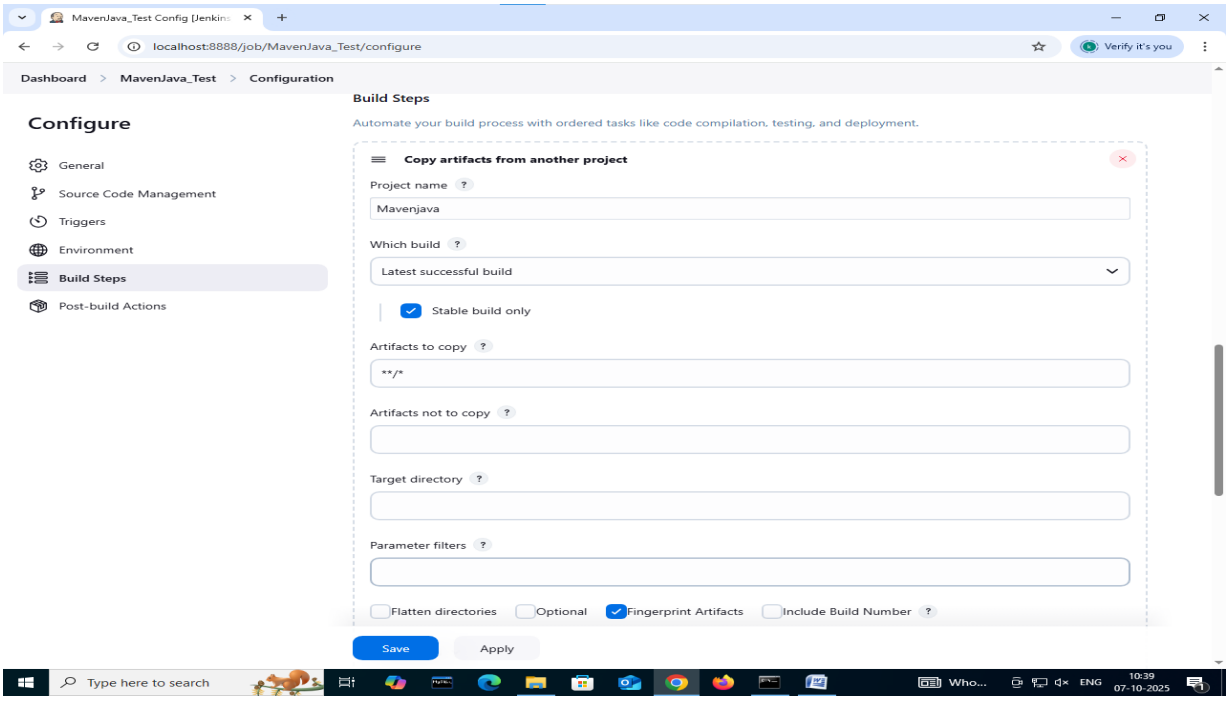
Give the description



In the source code management select none and in environment select “delete workspace before build starts”

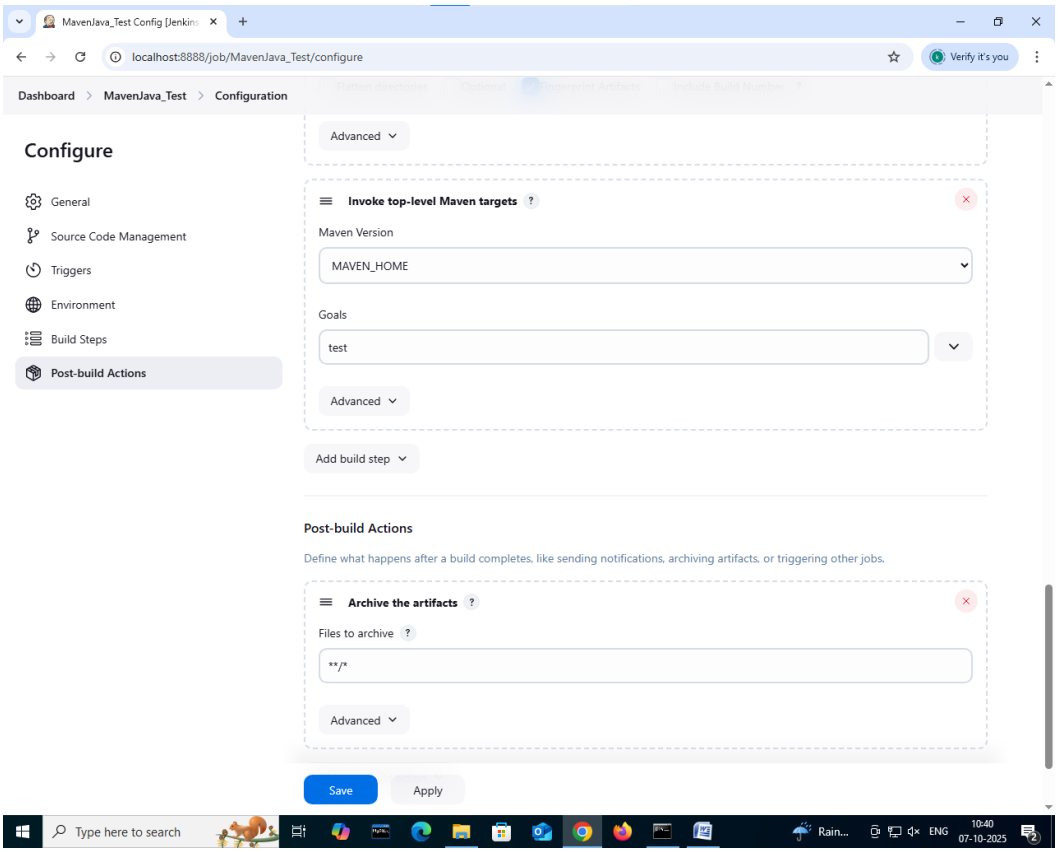


In the build steps> select add a build step> select “copy artifacts from another project” > give project name as Maven java and artifacts to copy as **/*

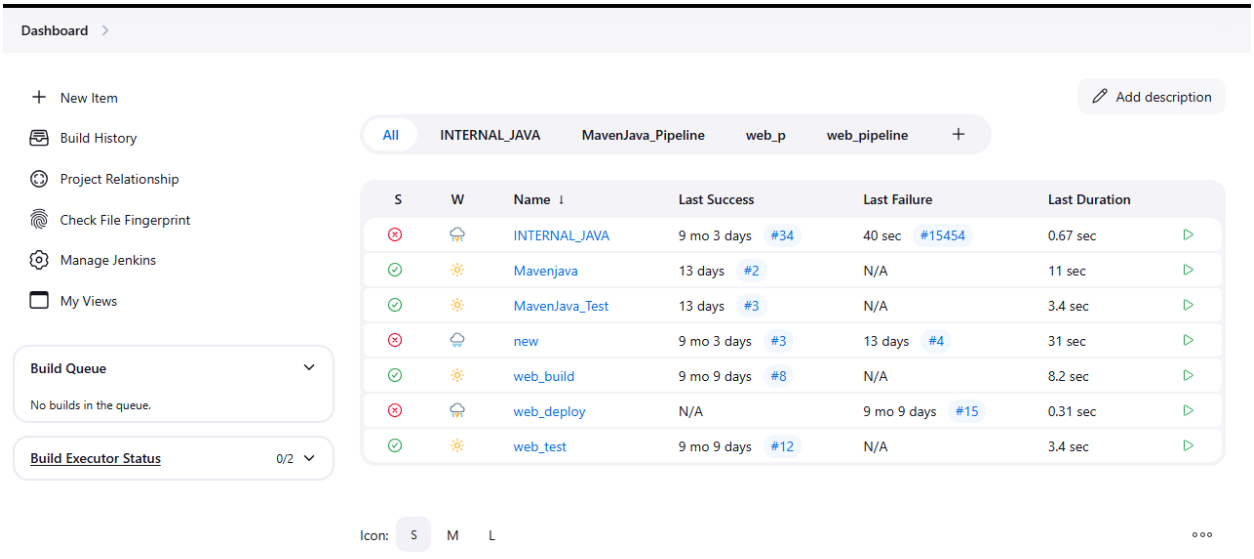


In the post build actions> select archive the artifacts and enter files as **/*

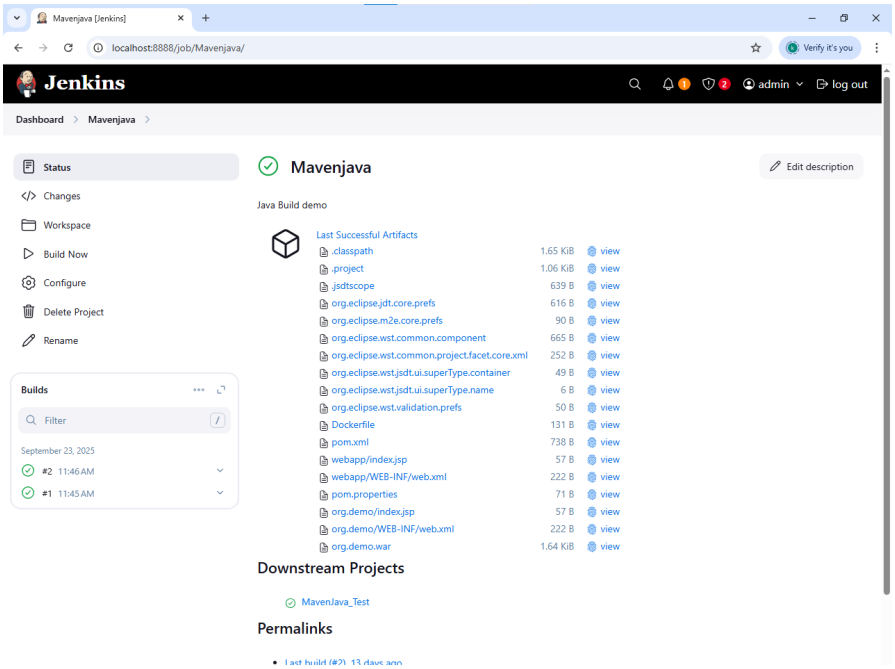
Click on apply and save



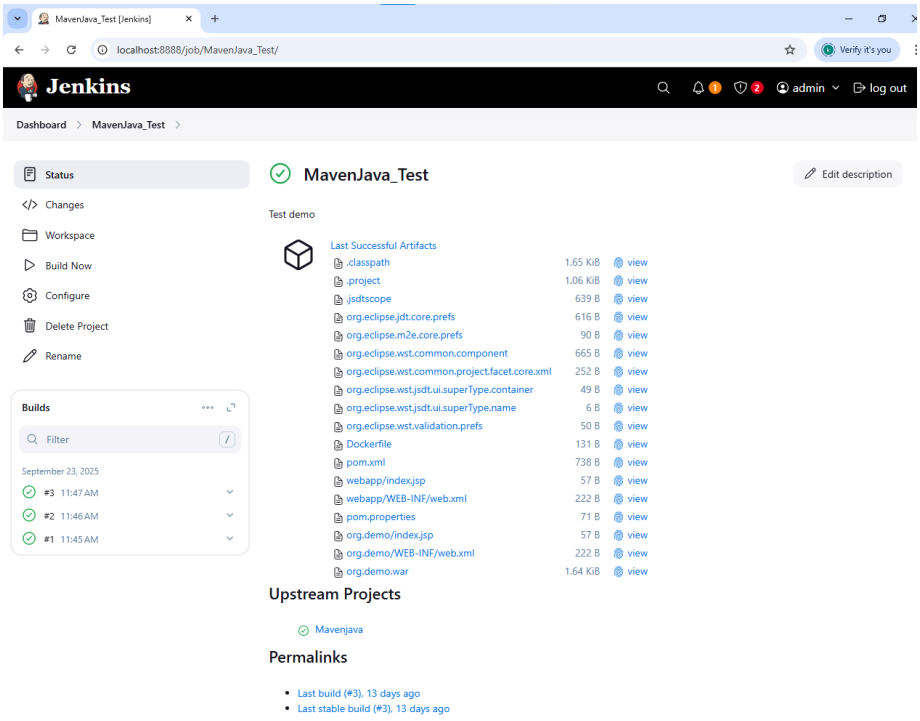
In the dashboard you will find MavenJava and MavenJava_Test



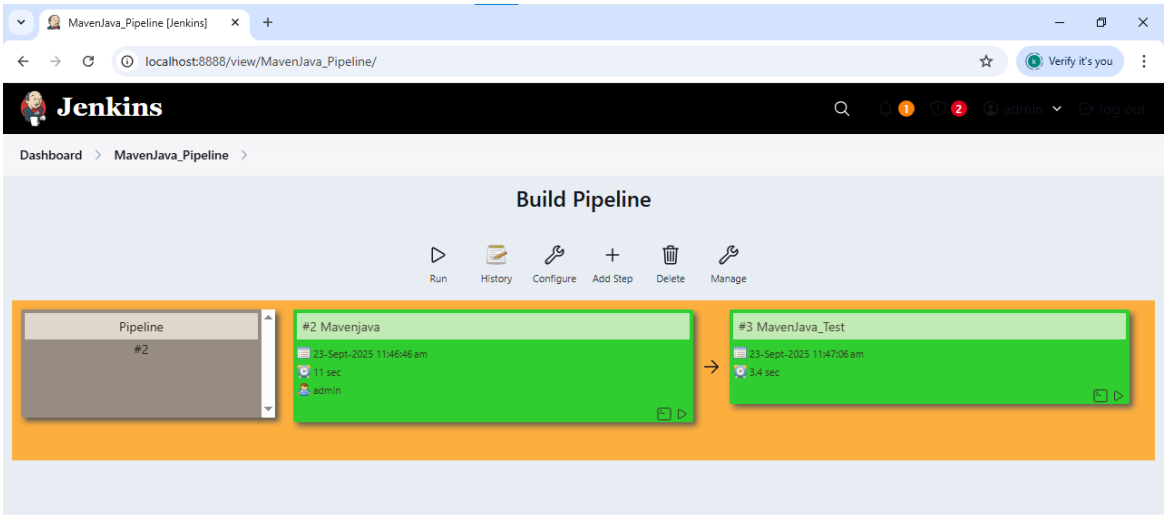
If you open the MavenJava file the following will be shown in case on no errors



If you open the MavenJava_Test file the following will be shown in case on no errors



MavenJava_pipeline

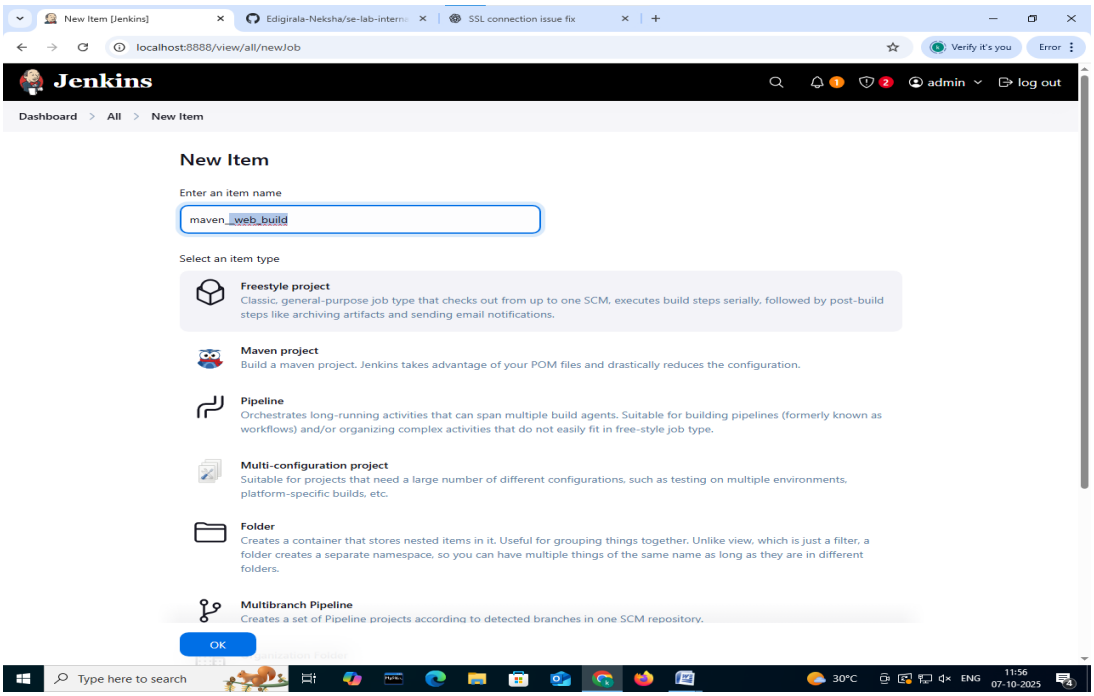


II. Maven Web Automation Steps:

Create Freestyle Project (e.g., MavenWeb_Build)

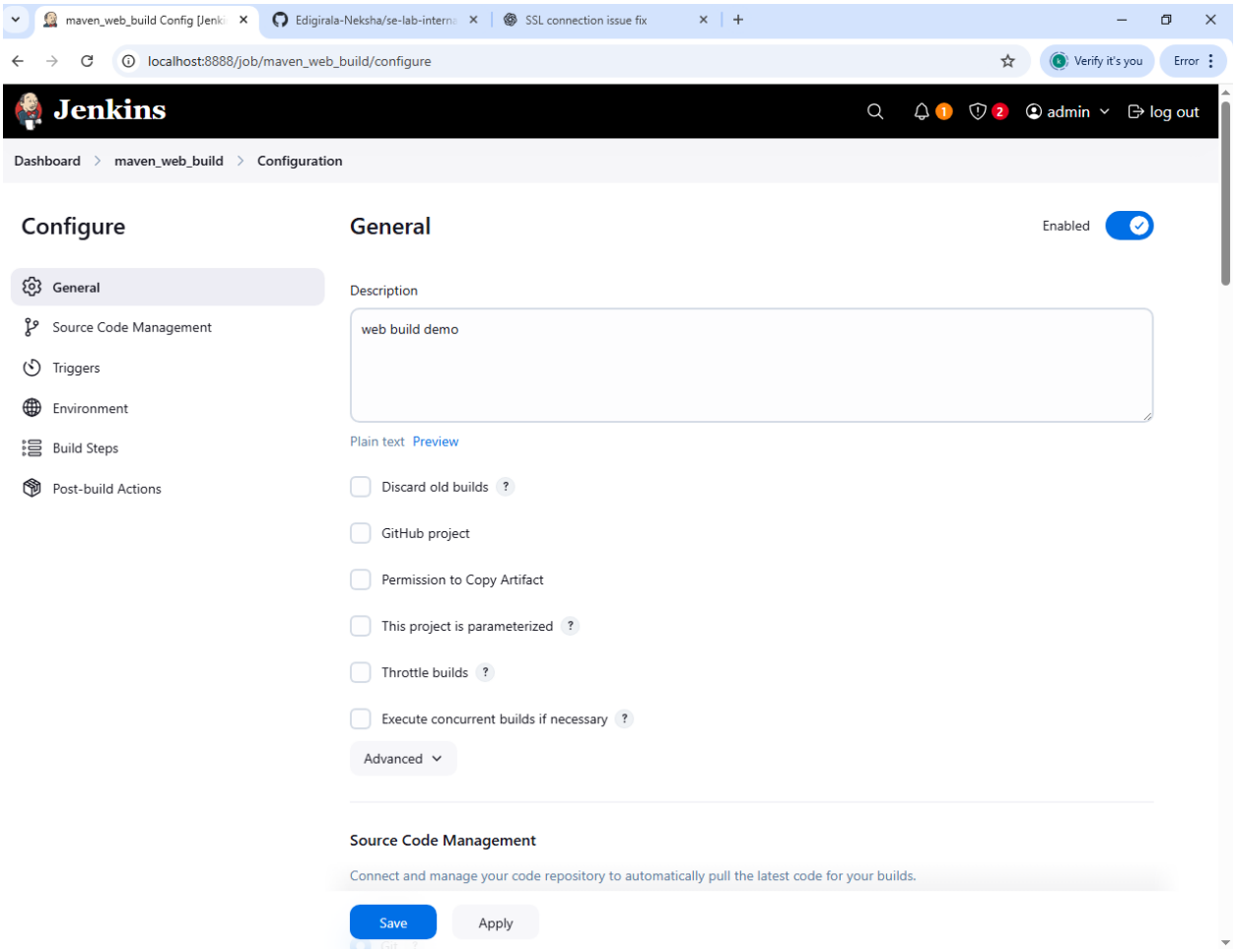
Step 1: Open Jenkins (localhost:8888)

Click on "New Item" (left side menu) and name it as maven_web_build> select freestyle project > click on “OK”

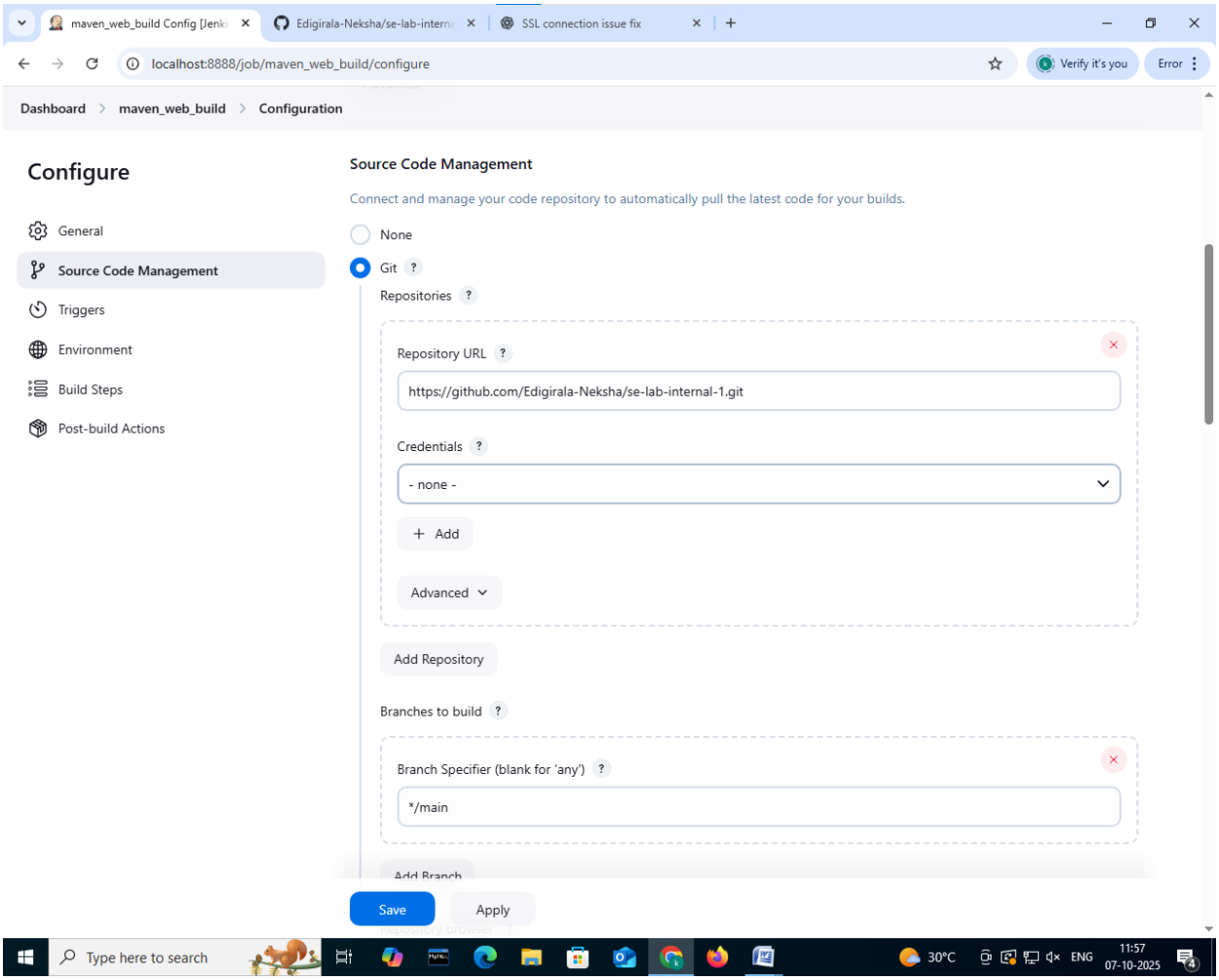


Step 2: Configuration of maven_web_build project

Give the description



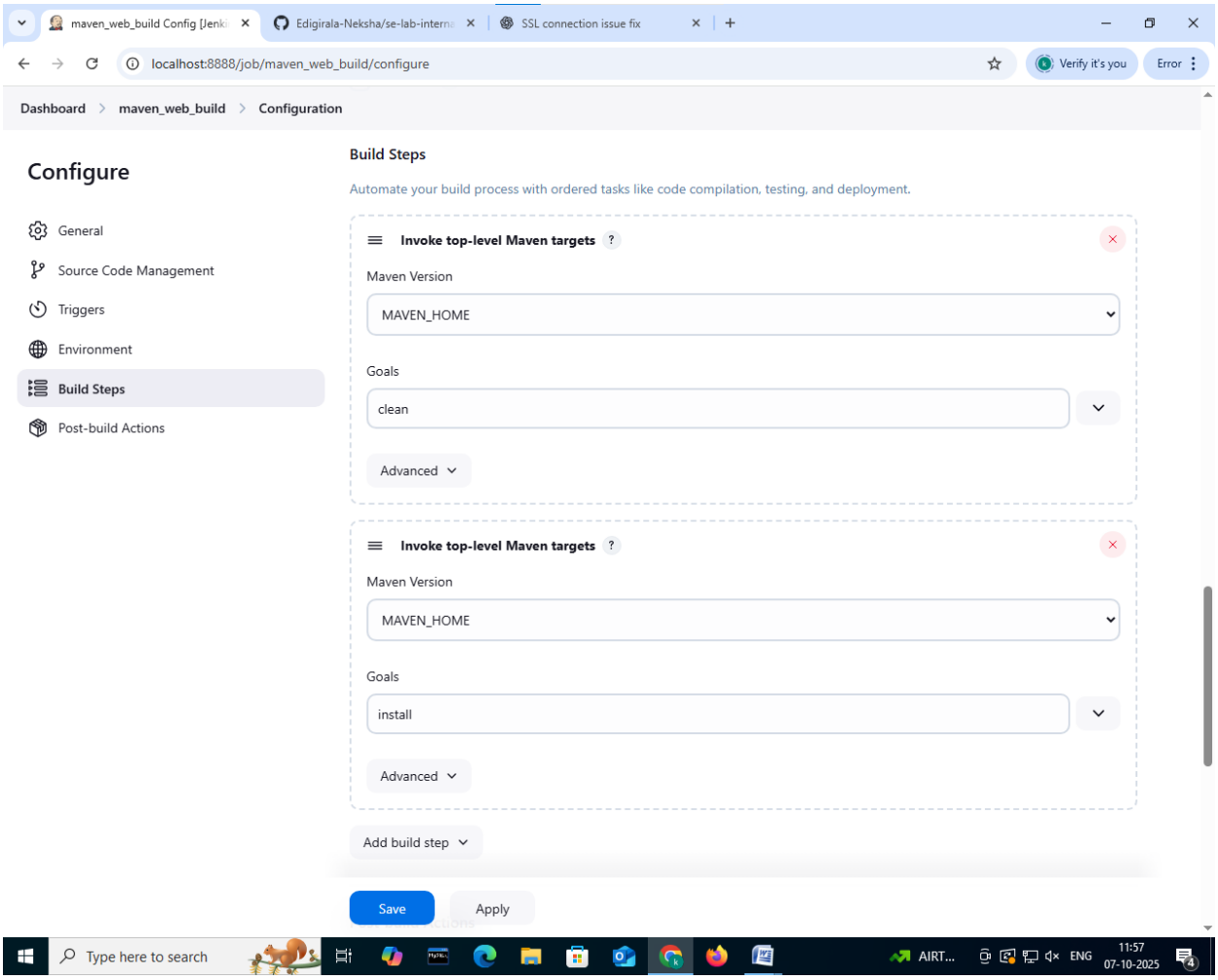
In the source code management select git and give the git repo link



In the build steps click on add build step > give maven version as MAVEN_HOME > select invoke top-level maven targets > goals as clean

For the second build step,

In the build steps click on add build step > give maven version as MAVEN_HOME > select invoke top-level maven targets > goals as install

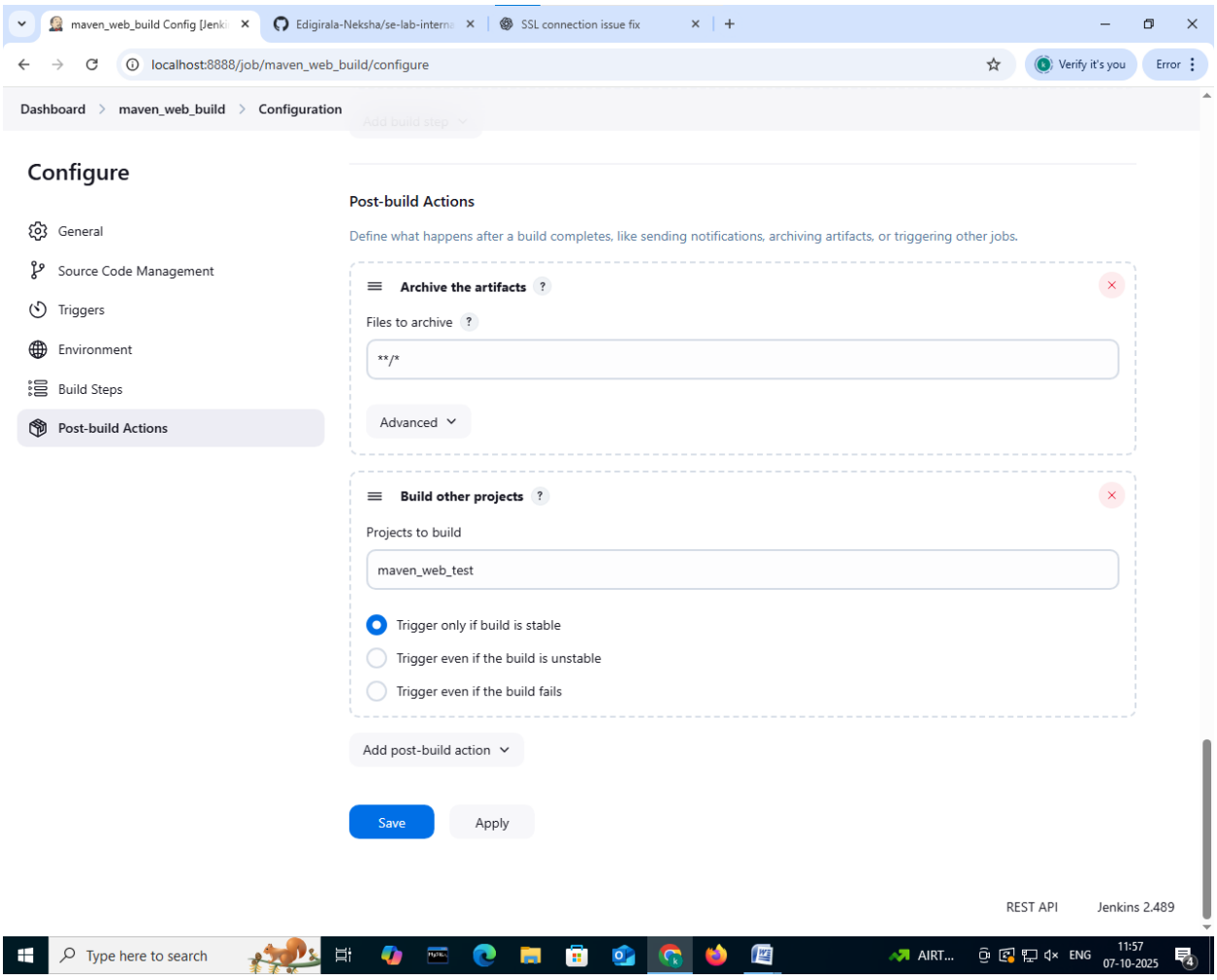


In the post build actions > click on add post build action > select the archive the artifacts > in the file to archive give “**/*”

For the second post build action,

In the post build actions > click on add post build action >select build other projects > give projects to build as maven_web_test

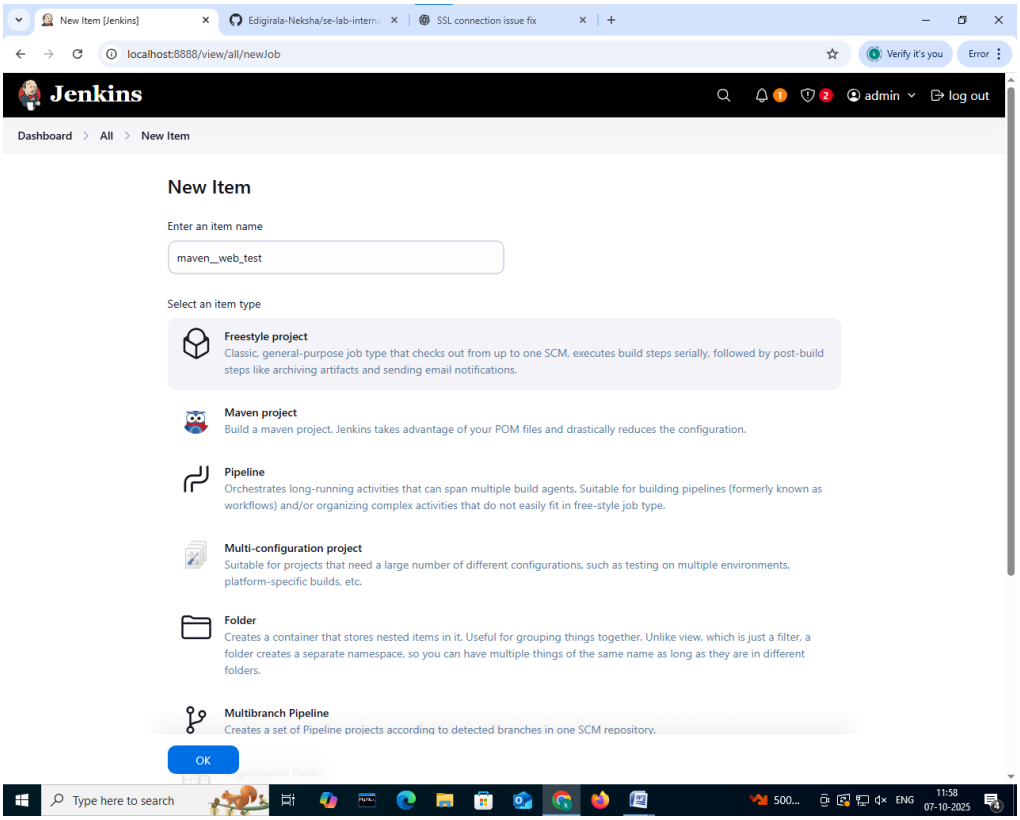
Click on apply and save



Create Freestyle Project (e.g., MavenWeb_Test):

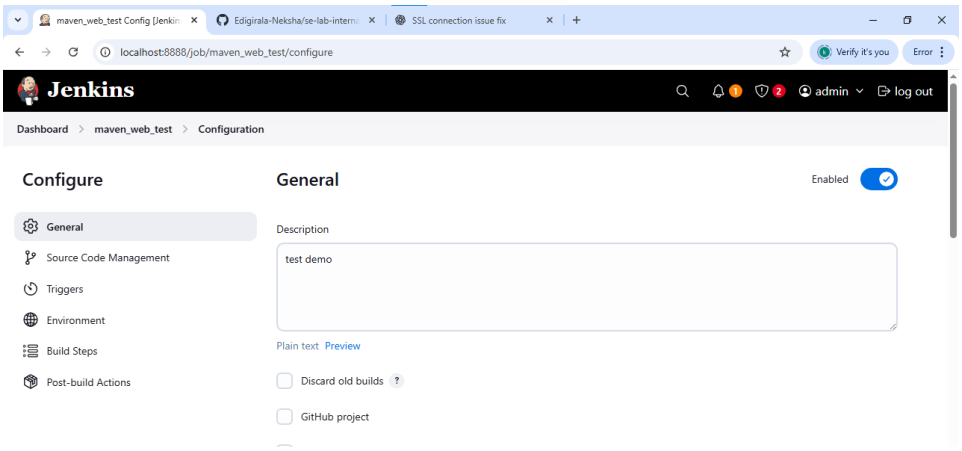
Step 1: Open Jenkins (localhost:8888)

Click on "New Item" (left side menu) and name it as maven_web_test> select freestyle project > click on “OK”

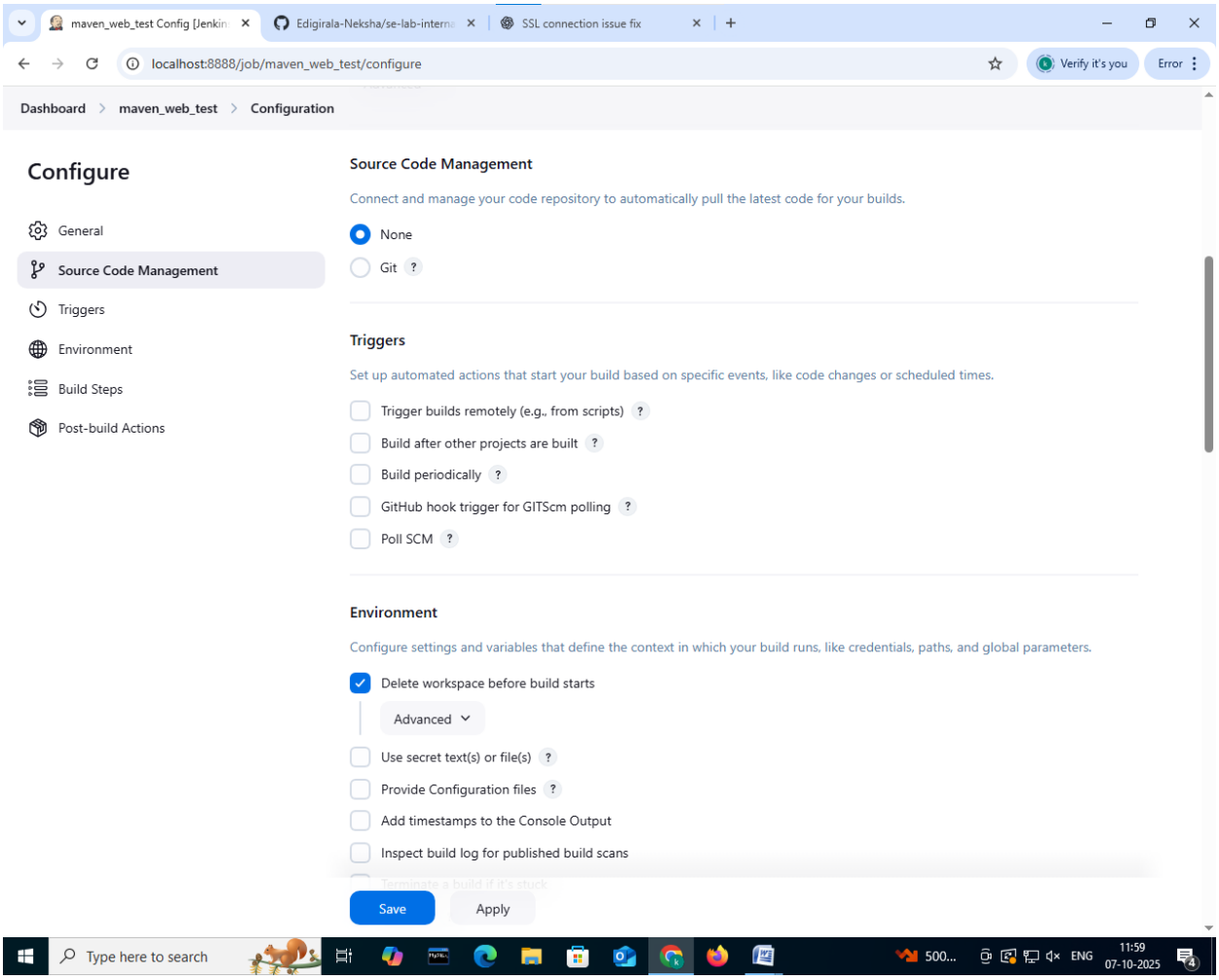


Step 2: Configuration of maven_web_test project

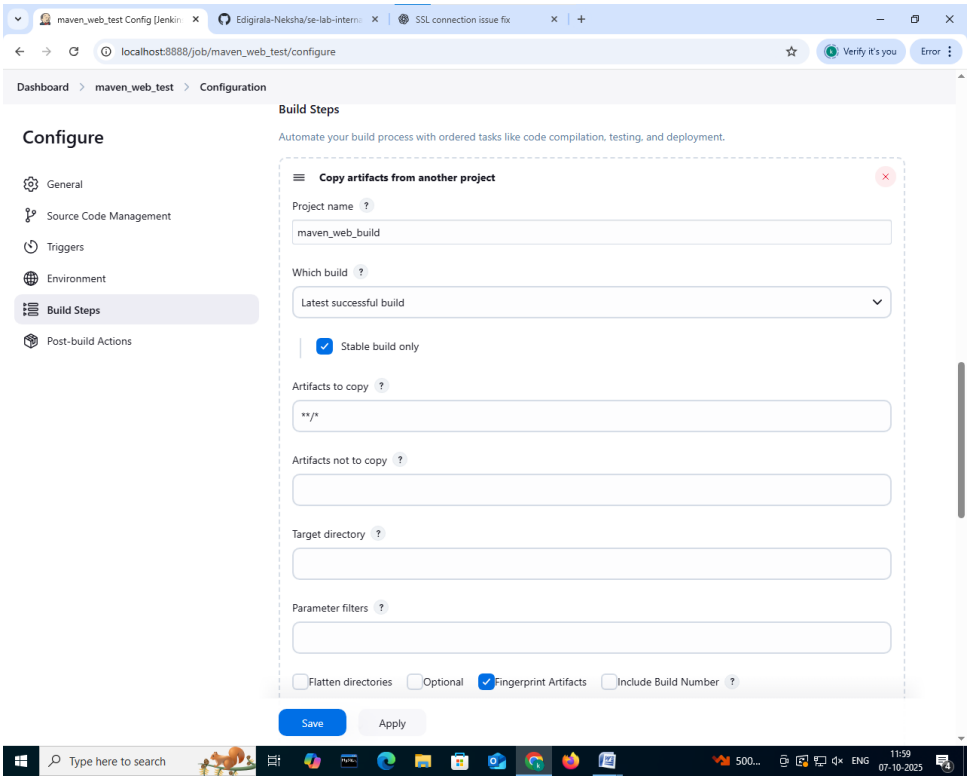
Give the description



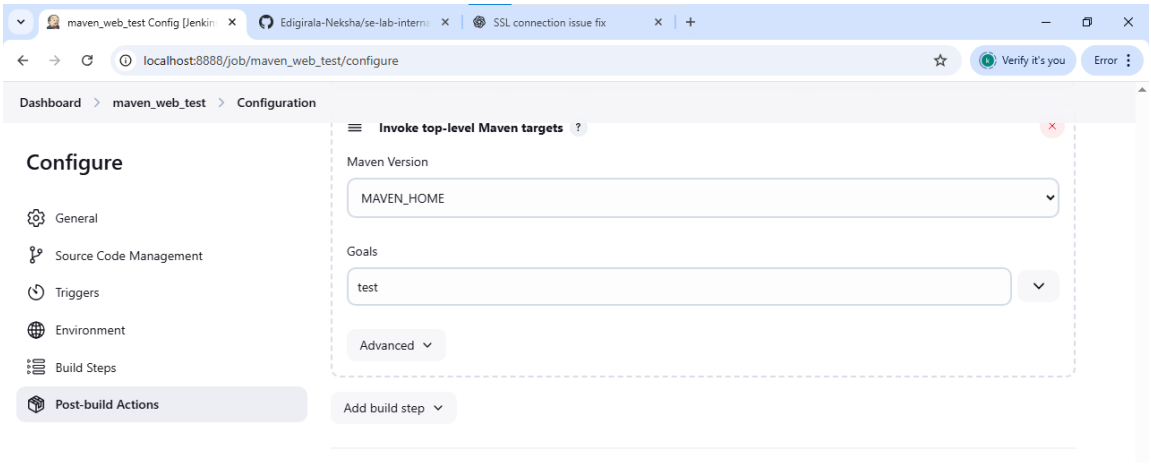
In the source code management select none and in environment select “delete workspace before build starts”



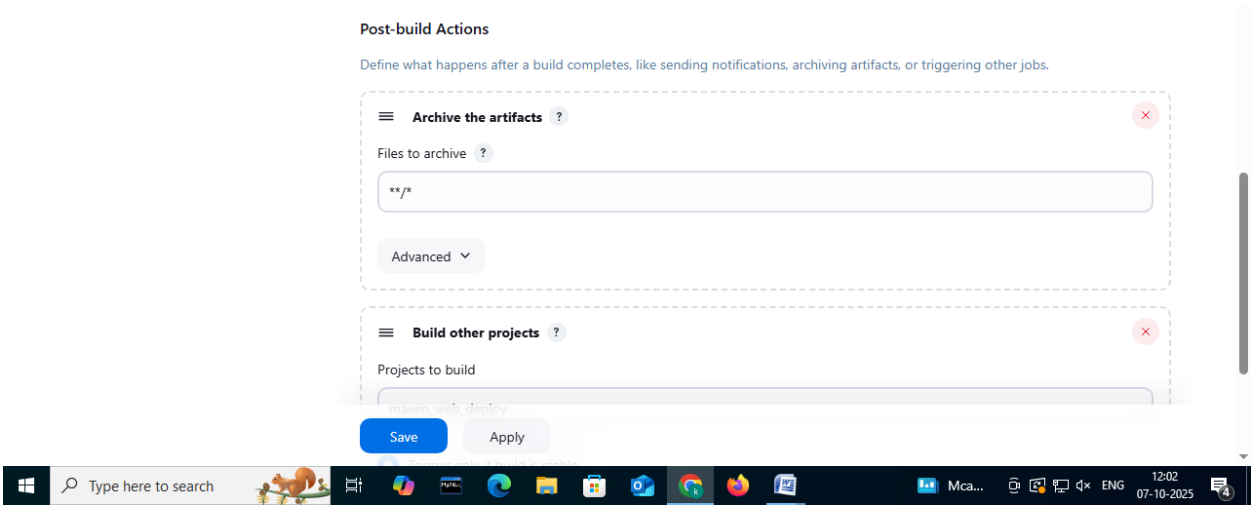
In the build steps click on add build step > select copy artifacts from another project > give project name as maven_web_build > give artifacts to copy as **/*



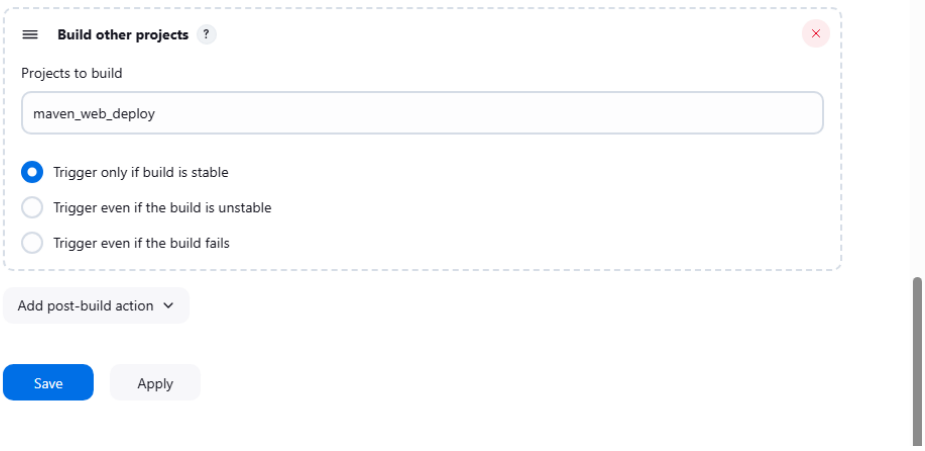
In the build steps click on add build step > give maven version as MAVEN_HOME > select invoke top-level maven targets > goals as test



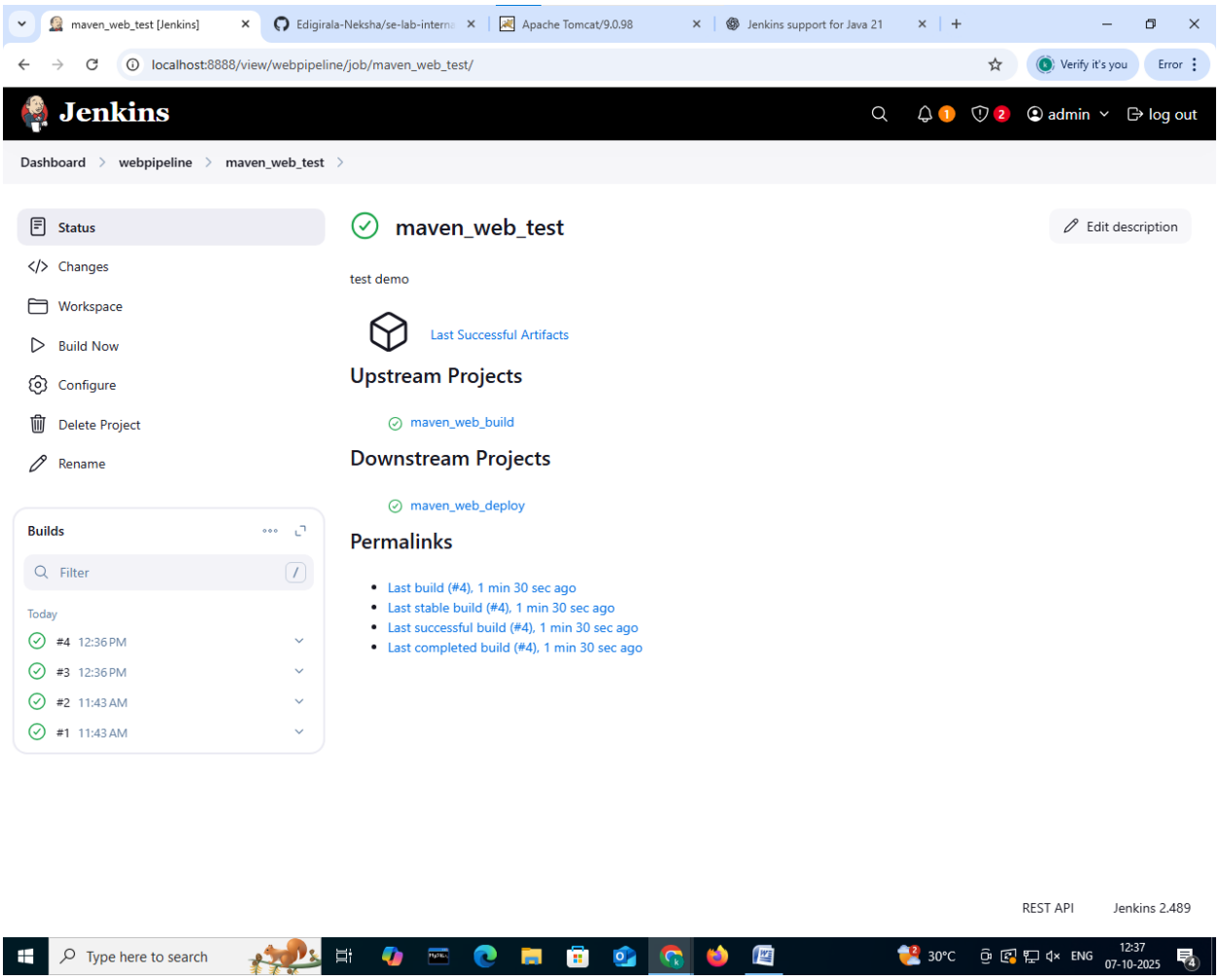
In the post build actions > click on add post build action > select the archive the artifacts > in the file to archive give **/*



In the post build actions > click on add post build action >select build other projects > give name as maven_web_deploy> select “trigger only if build is stable”



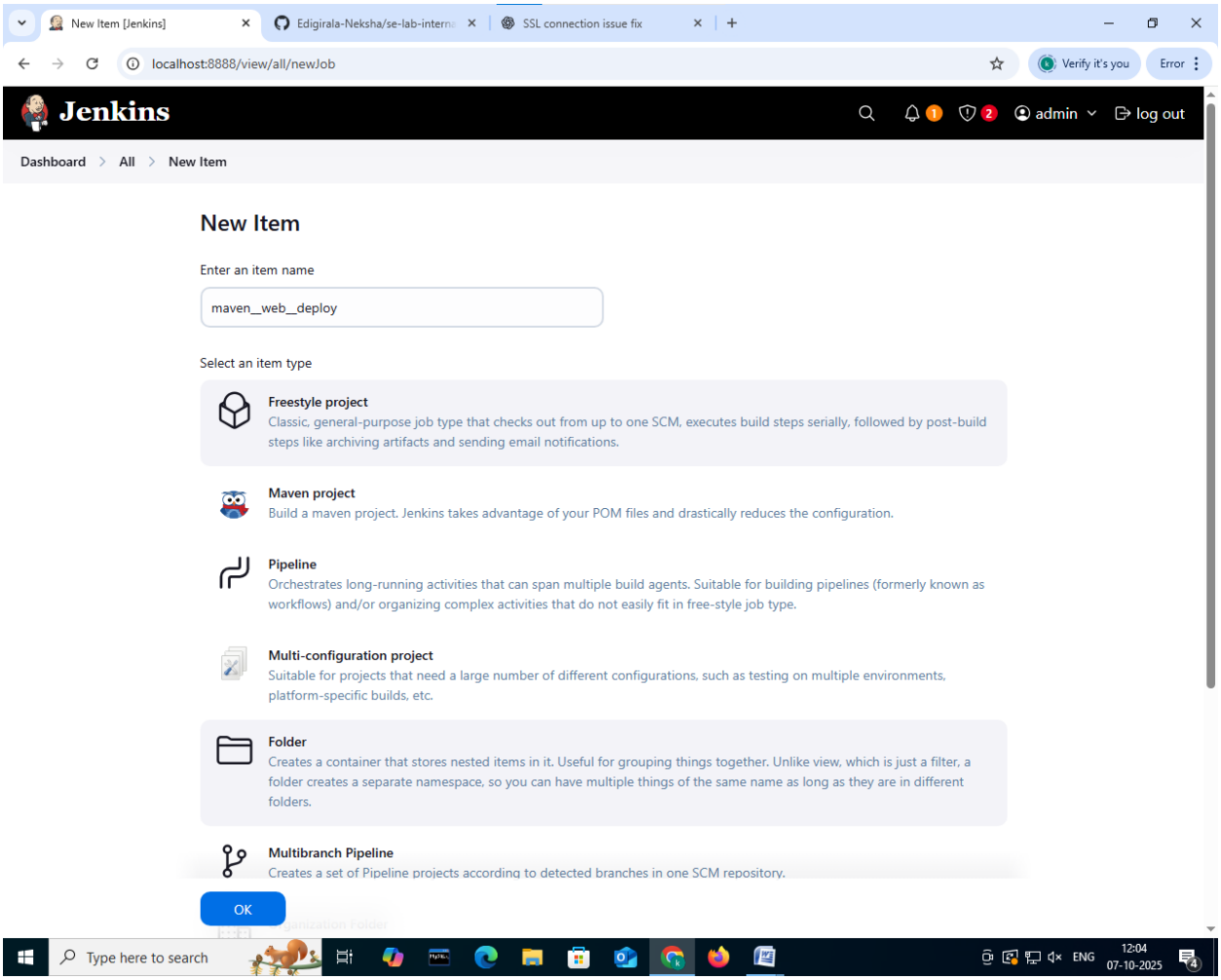
If the build is success:



Create Freestyle Project (e.g., MavenWeb_Deploy):

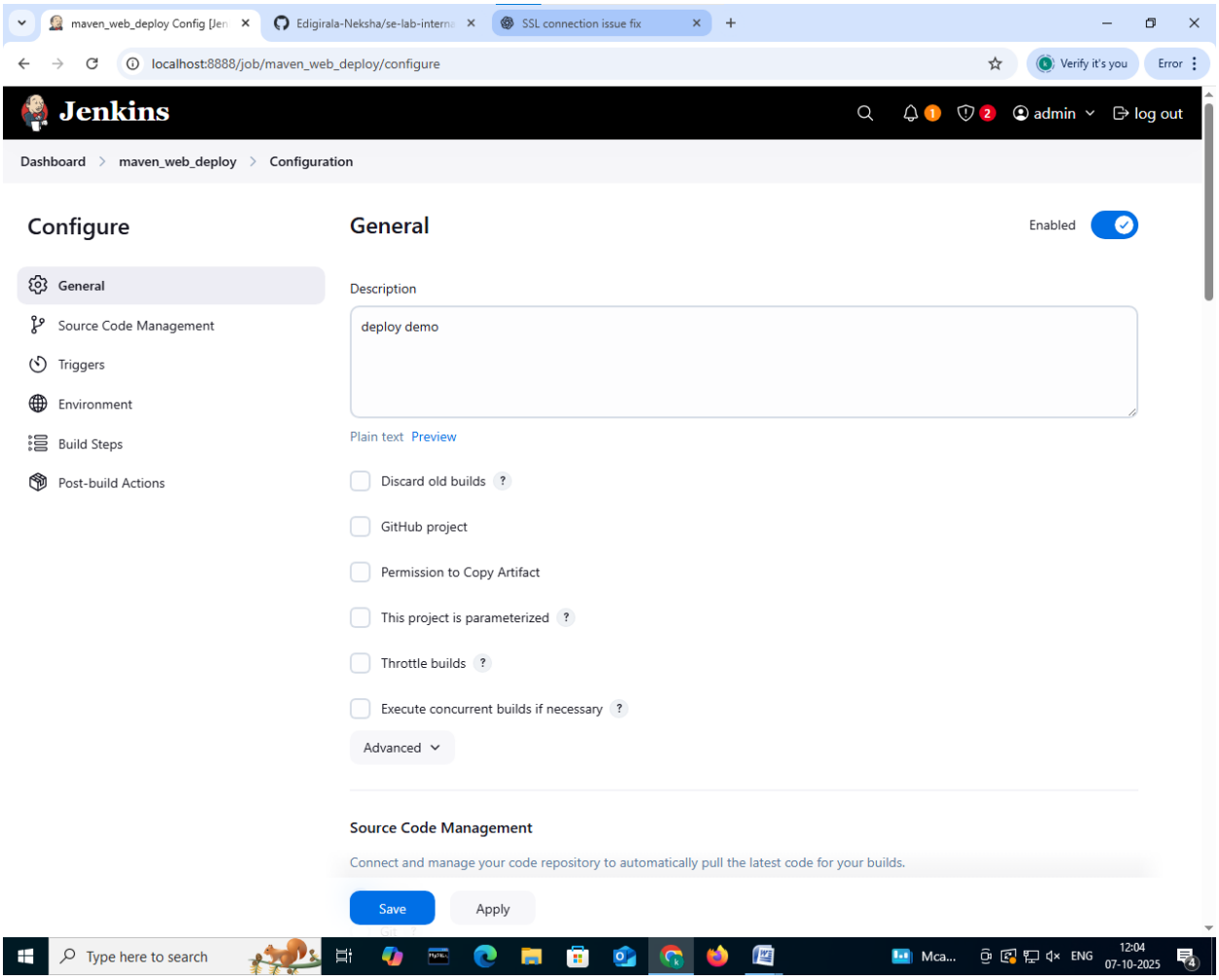
Step 1: Open Jenkins (localhost:8888)

Click on "New Item" (left side menu) and name it as maven_web_deploy > select freestyle project > click on “OK”

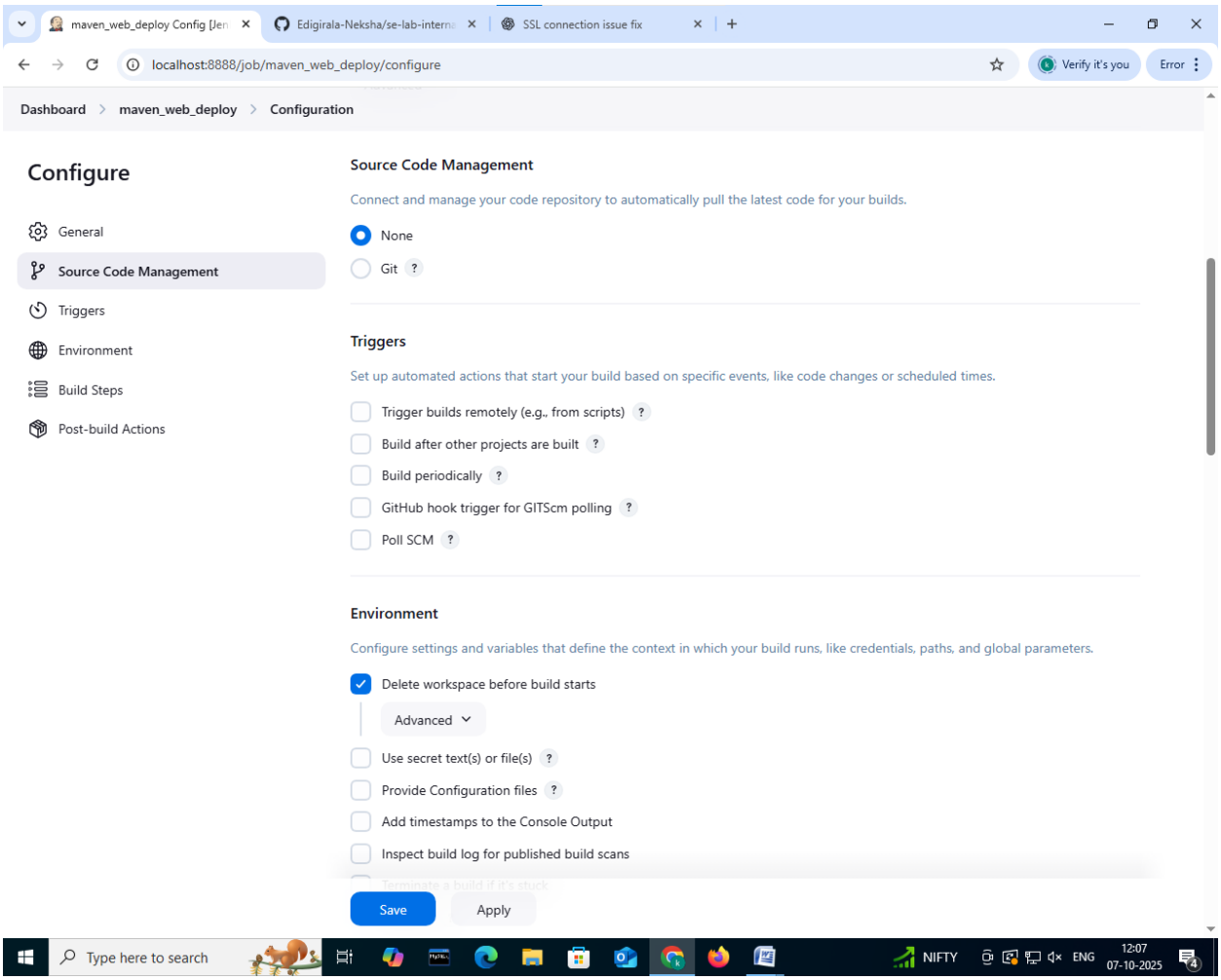


Step 2: Configuration of maven_web_deploy project

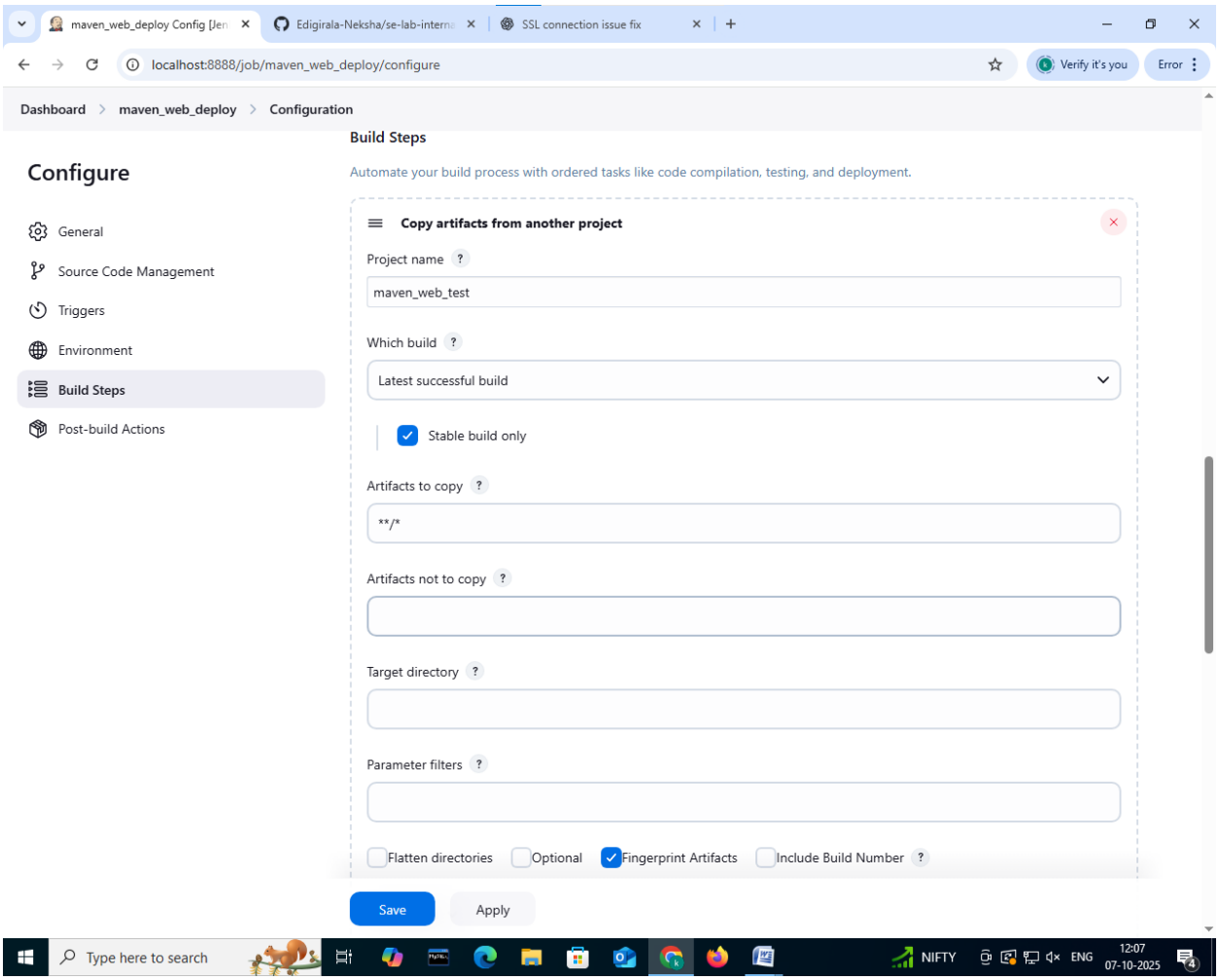
Give the description



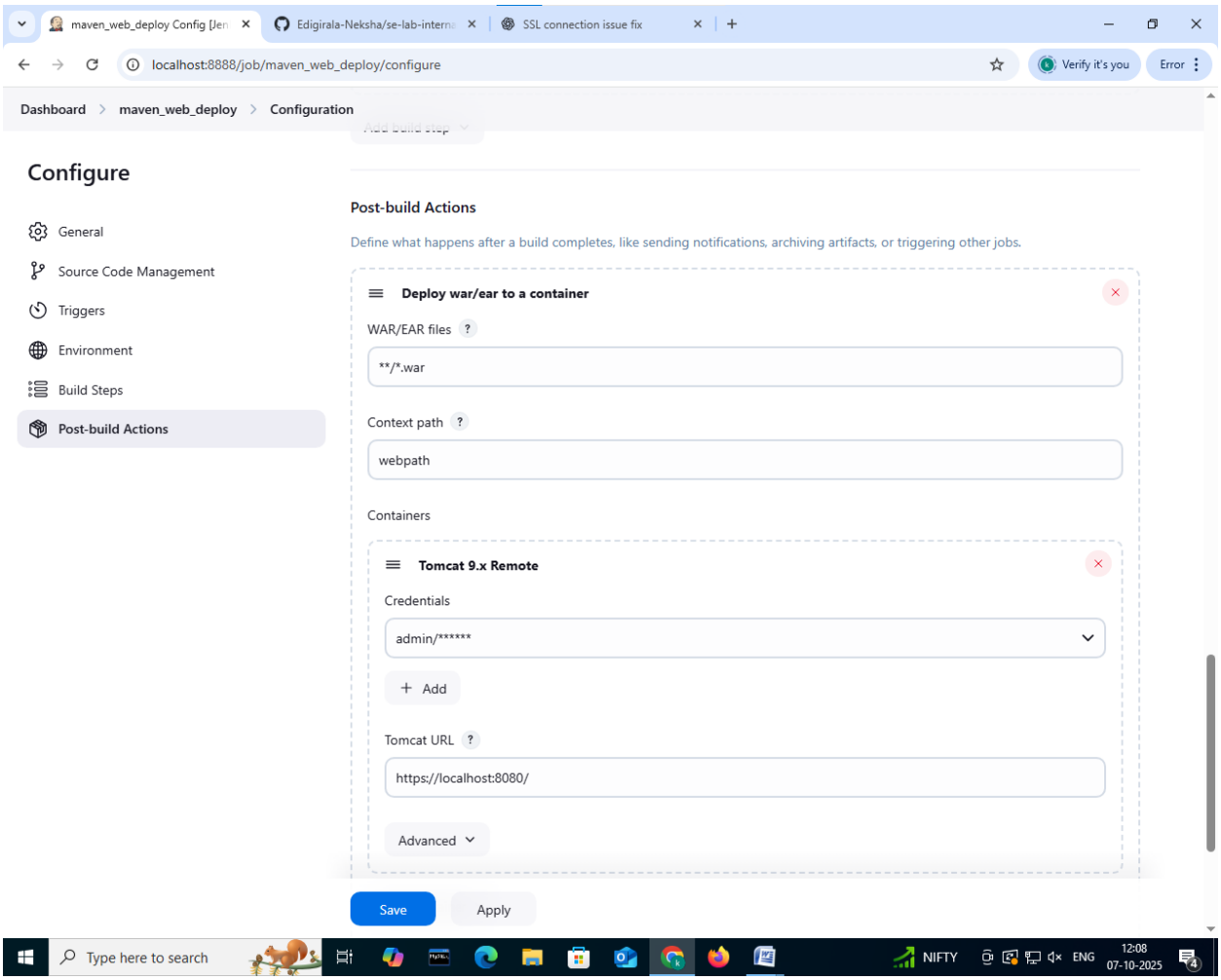
In the source code management select none and in environment select “delete workspace before build starts”



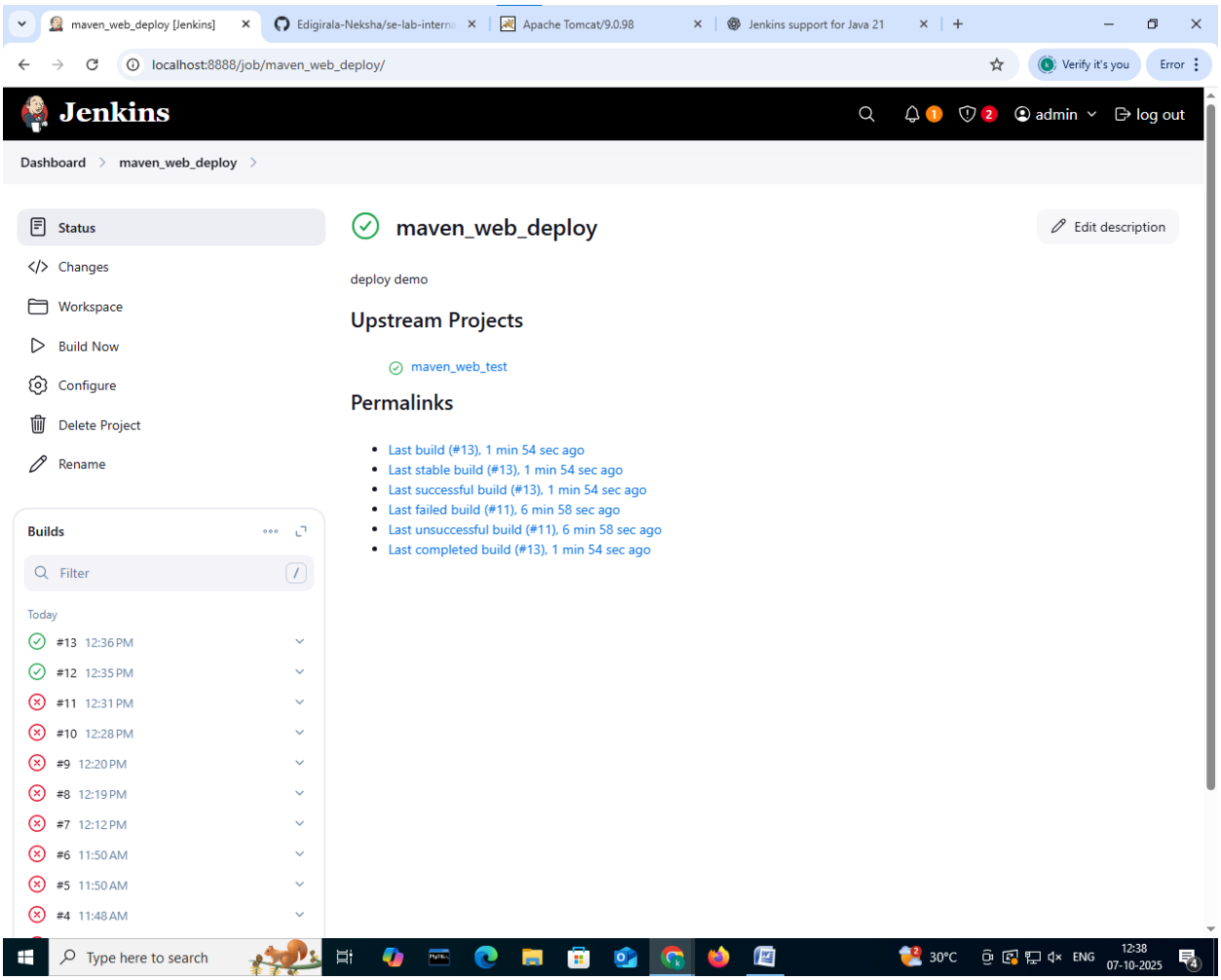
In the build steps click on add build step > select copy artifacts from another project > give project name as maven_web_test > give artifacts to copy as **/*



In the post build actions > click on add post build actions > select deploy war/ear to a container > enter war/ear files as **/*.war> context path as webpath > give the credentials and tomcat URL



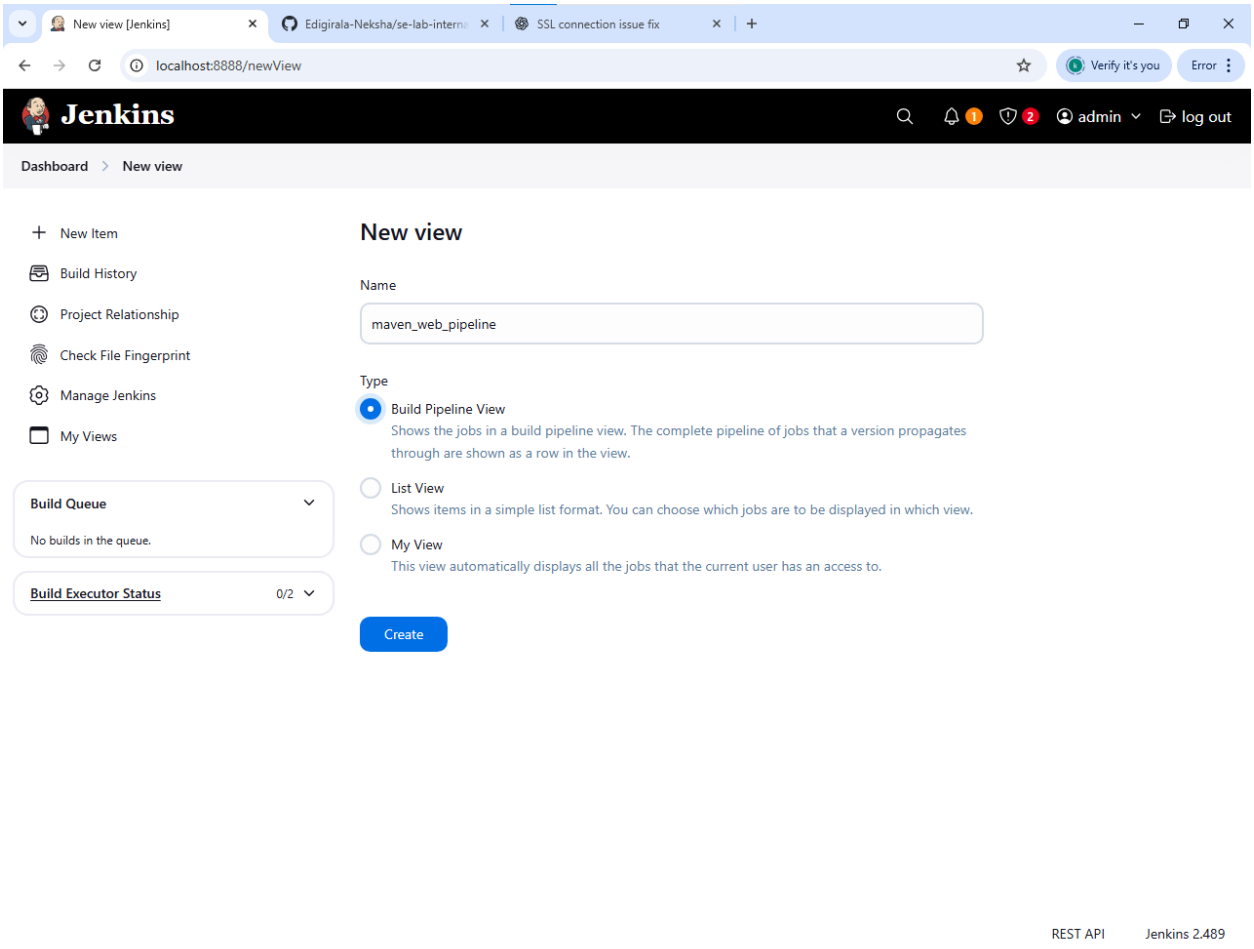
If the build is success:



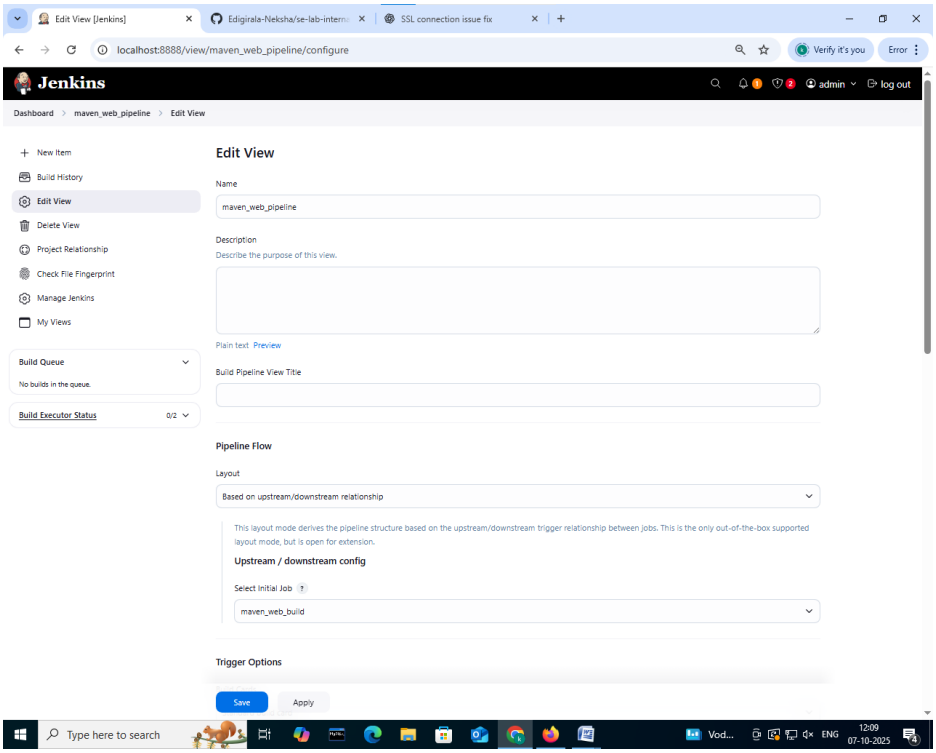
Create Pipeline View for MavenWeb

Click "+" beside "All" on the dashboard and Enter name as maven_web_pipeline

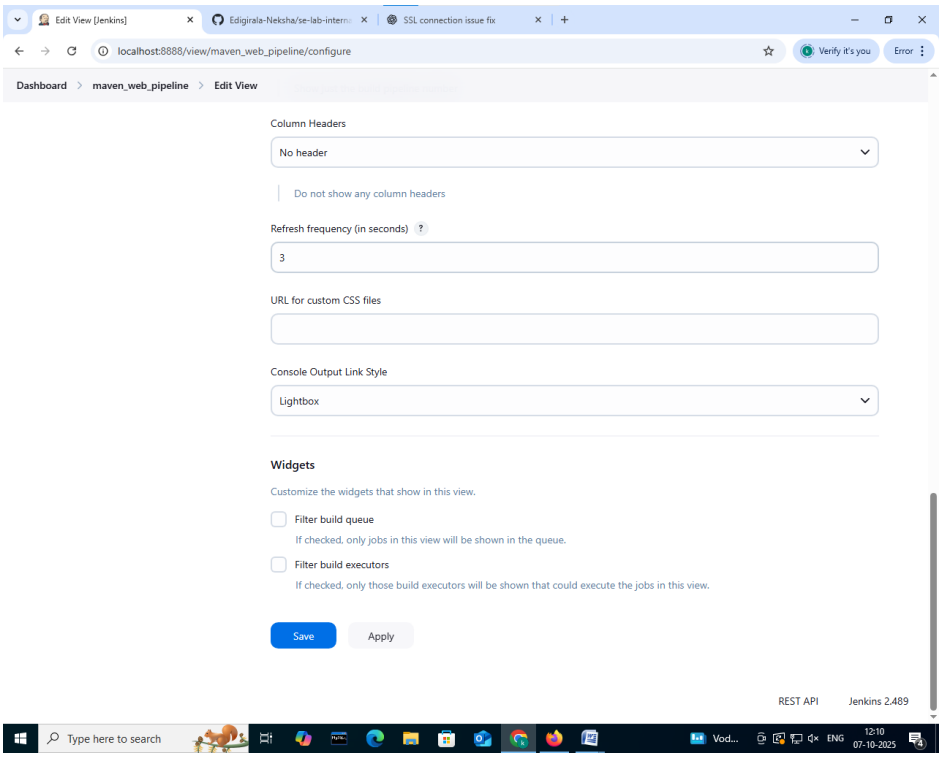
Select type as build pipeline view



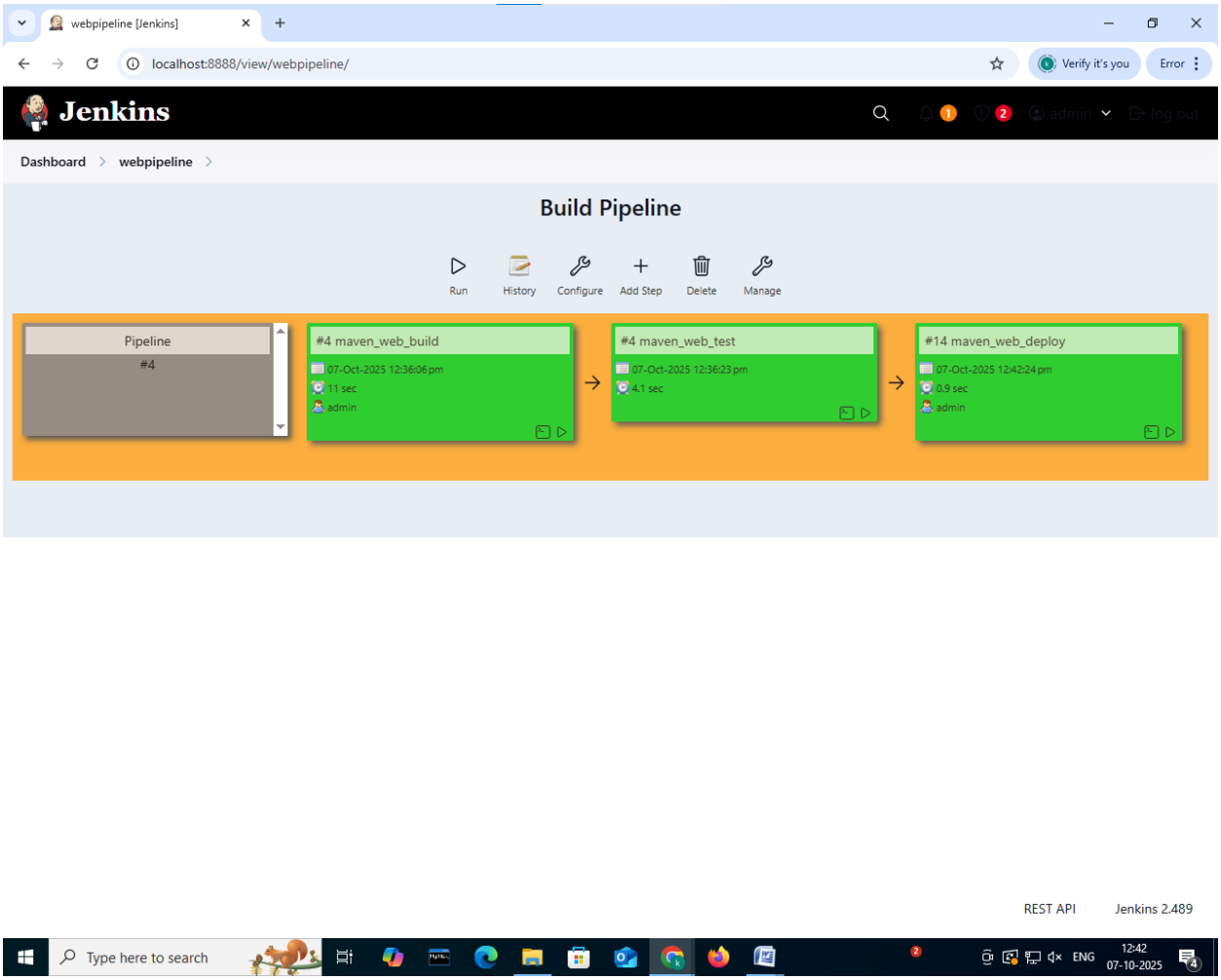
Give the description and in the upstream directly the maven_web_build will be shown



Click on apply and save

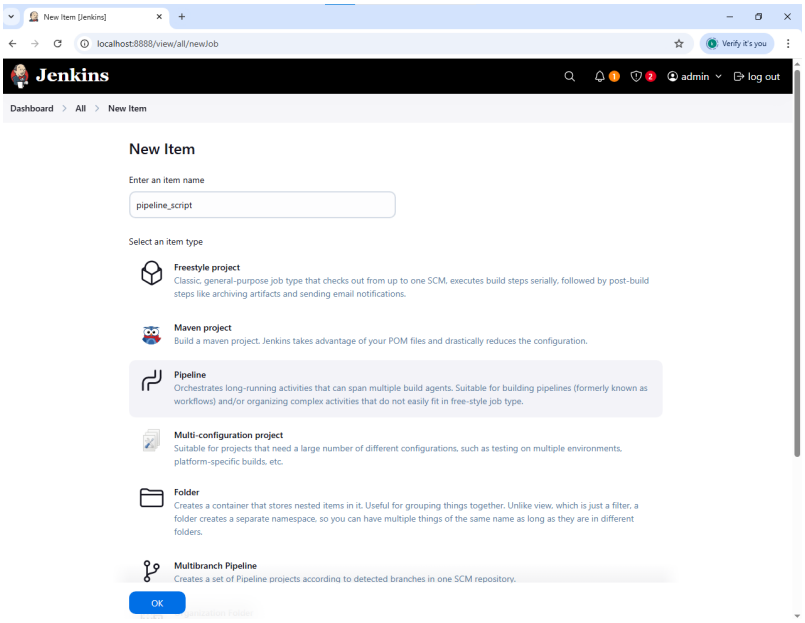


In the stage view it we be shown as:

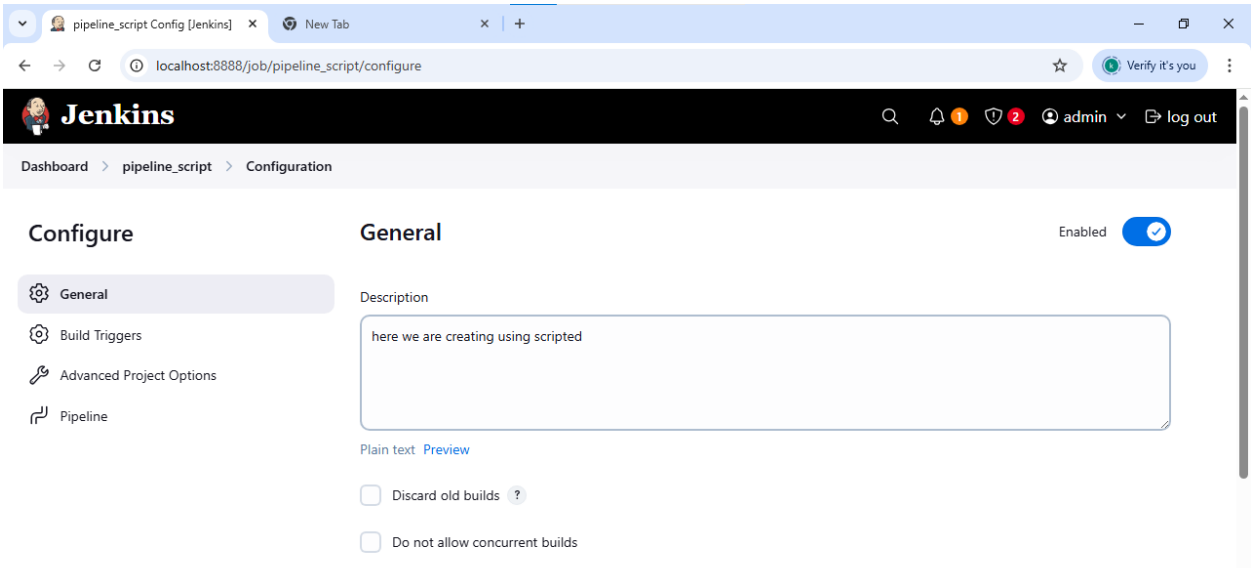


9.Pipeline Creation using script

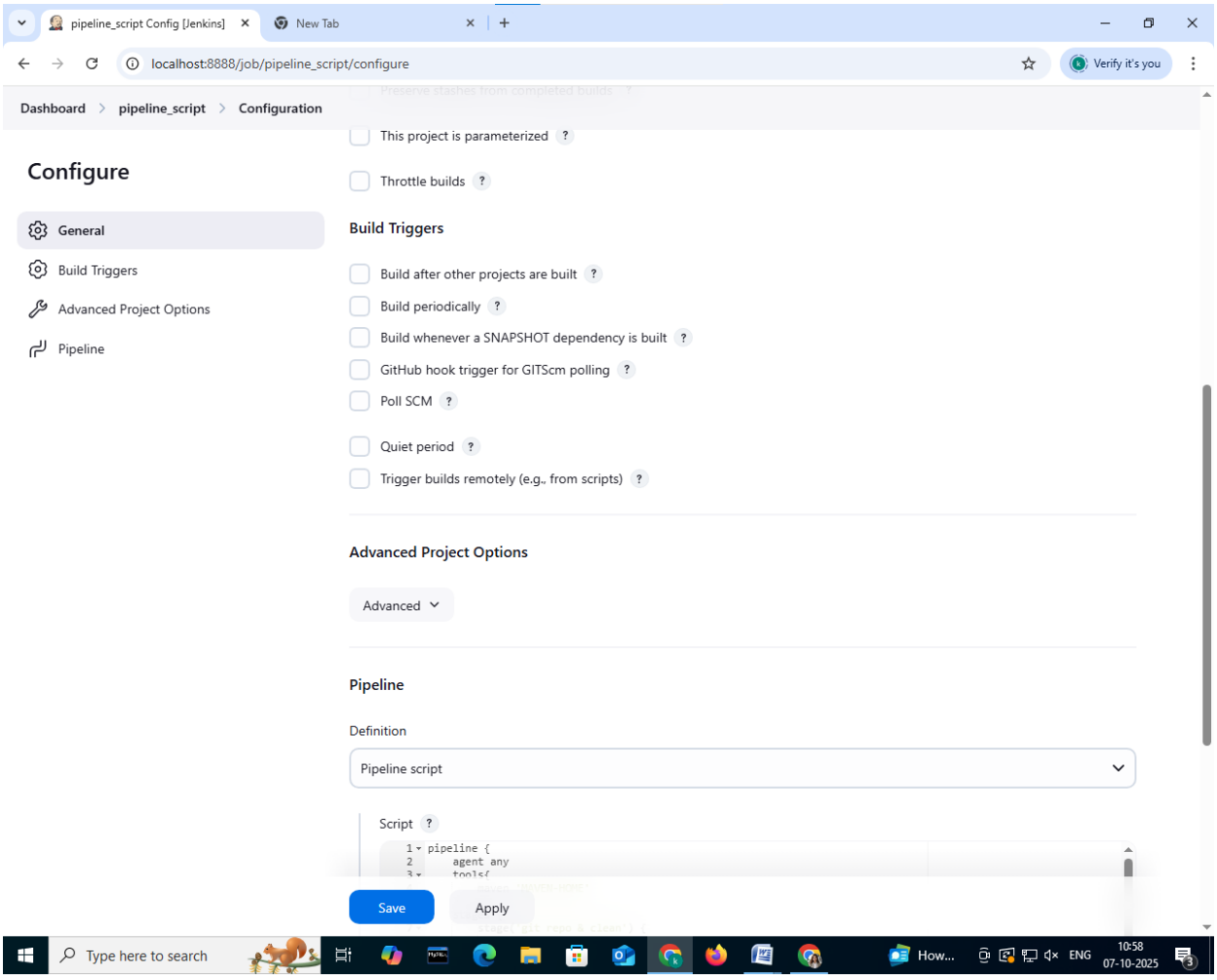
Step 1: In the Jenkins select the new item and give the name as pipeline_script and select pipeline and click ok



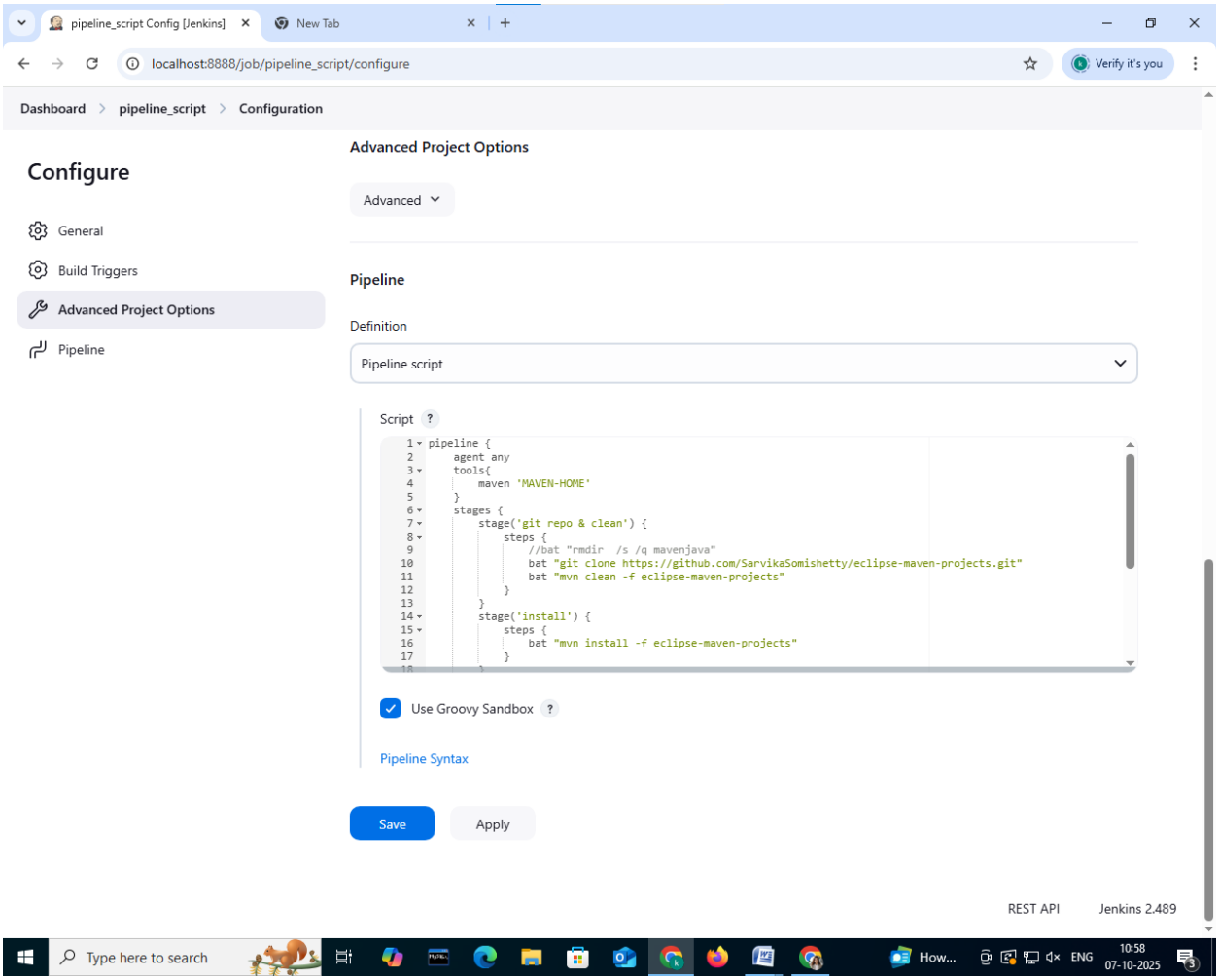
Step 2: In the configuration, give the description



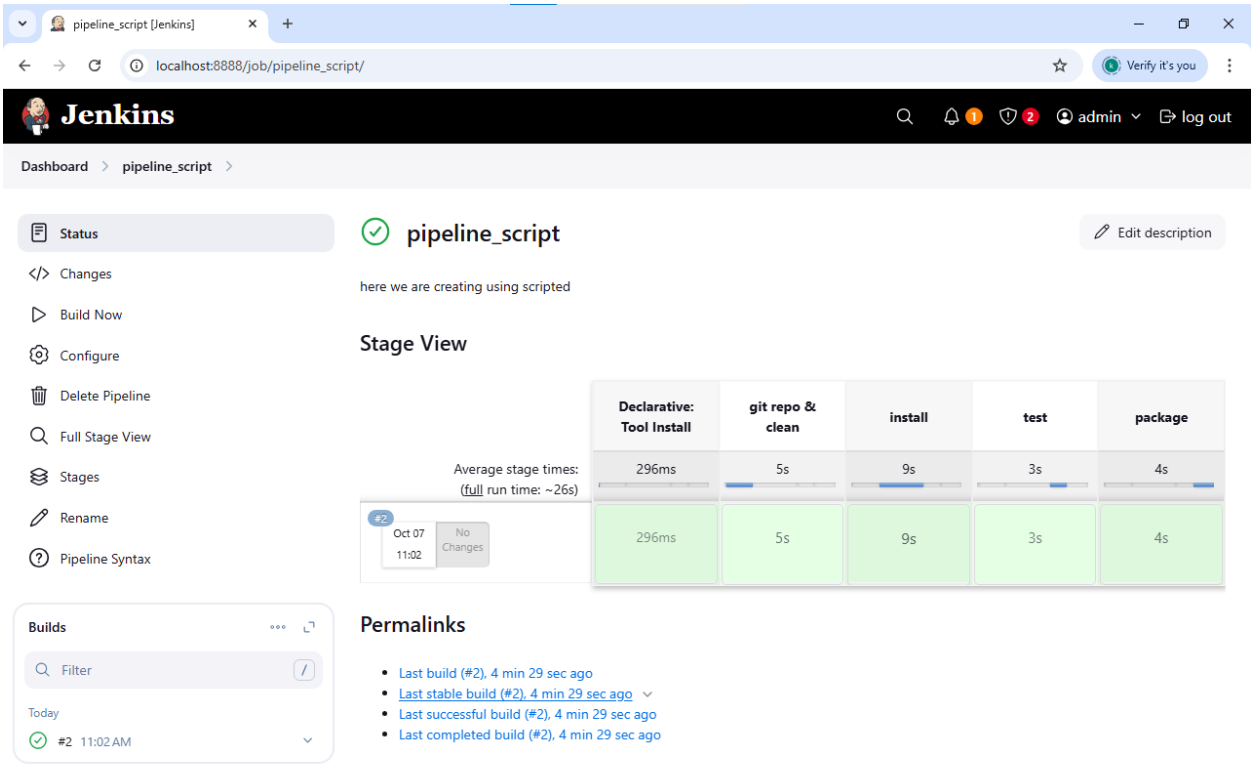
Step 3: In the pipeline section give definition as pipeline script and enter the script with git reop link and project name



Step 4: click on apply and then save



Step 8: Check the stage view. If is successful.



10. Kubernetes Using Minikube:

Step -1:

Start Minikube : Command- minikube start

- First, you need to start your Kubernetes cluster using Minikube.
- When you start it, Minikube sets up a lightweight virtual machine on your system and runs a local Kubernetes node inside it.

Step-2: Then check for the status Minikube status

Step-3:Create an image

```
PS C:\Users\User>
PS C:\Users\User> kubectl delete deployment mynginx
deployment.apps "mynginx" deleted
PS C:\Users\User> kubectl create deployment mynginx --image=nginx
deployment.apps/mynginx created
PS C:\Users\User> kubectl expose deployment mynginx --type=NodePort --port=80
service/mynginx exposed
PS C:\Users\User> kubectl get pods
NAME                       READY   STATUS    RESTARTS   AGE
mynginx-79bb8756c7-wpslj   1/1     Running   0          34s
```

Step-4: Check the NGINX Service Details

- After creating the service, check its details to see which port Kubernetes assigned to it.

```
    DownwardAPI:               true
QoS Class:                     BestEffort
Node-Selectors:                <none>
Tolerations:                   node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                               node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type    Reason     Age   From               Message
  ----    ------     ----  ----               -------
  Normal  Scheduled  68s   default-scheduler  Successfully assigned default/mynginx-79bb8756c7-wpslj to minikube
  Normal  Pulling    67s   kubelet            Pulling image "nginx"
  Normal  Pulled     65s   kubelet            Successfully pulled image "nginx" in 2.416s (2.416s including waiting). Image size: 159974475 bytes.
  Normal  Created    65s   kubelet            Created container nginx
  Normal  Started    64s   kubelet            Started container nginx
PS C:\Users\User> kubectl scale deployment mynginx --replicas=4
deployment.apps/mynginx scaled
PS C:\Users\User> kubectl get service myngnix
Error from server (NotFound): services "myngnix" not found
PS C:\Users\User> kubectl port-forward svc/mynginx 8081:80
Forwarding from 127.0.0.1:8081 -> 80
Forwarding from [::1]:8081 -> 80
```

Step-5:check the detail of the kubectl .

```
PS C:\Users\User> kubectl describe pods
Name:             mynginx-79bb8756c7-wpslj
Namespace:        default
Priority:         0
Service Account:  default
Node:             minikube/192.168.49.2
Start Time:       Tue, 14 Oct 2025 12:38:19 +0530
Labels:           app=mynginx
                  pod-template-hash=79bb8756c7
Annotations:      <none>
Status:           Running
IP:               10.244.0.16
IPs:
  IP:           10.244.0.16
Controlled By:  ReplicaSet/mynginx-79bb8756c7
Containers:
  nginx:
    Container ID:   docker://675066efbd98a54ba39177103943b196de2c61f01d820ede859b48578f3e245e
    Image:          nginx
    Image ID:       docker-pullable://nginx@sha256:3b7732505933ca591ce4a6d860cb713ad96a3176b82f7979a8dfa9973486a0d6
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Tue, 14 Oct 2025 12:38:22 +0530
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-nh2rw (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-nh2rw:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
QoS Class:                   BestEffort
Node-Selectors:              <none>
Tolerations:                 node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
```

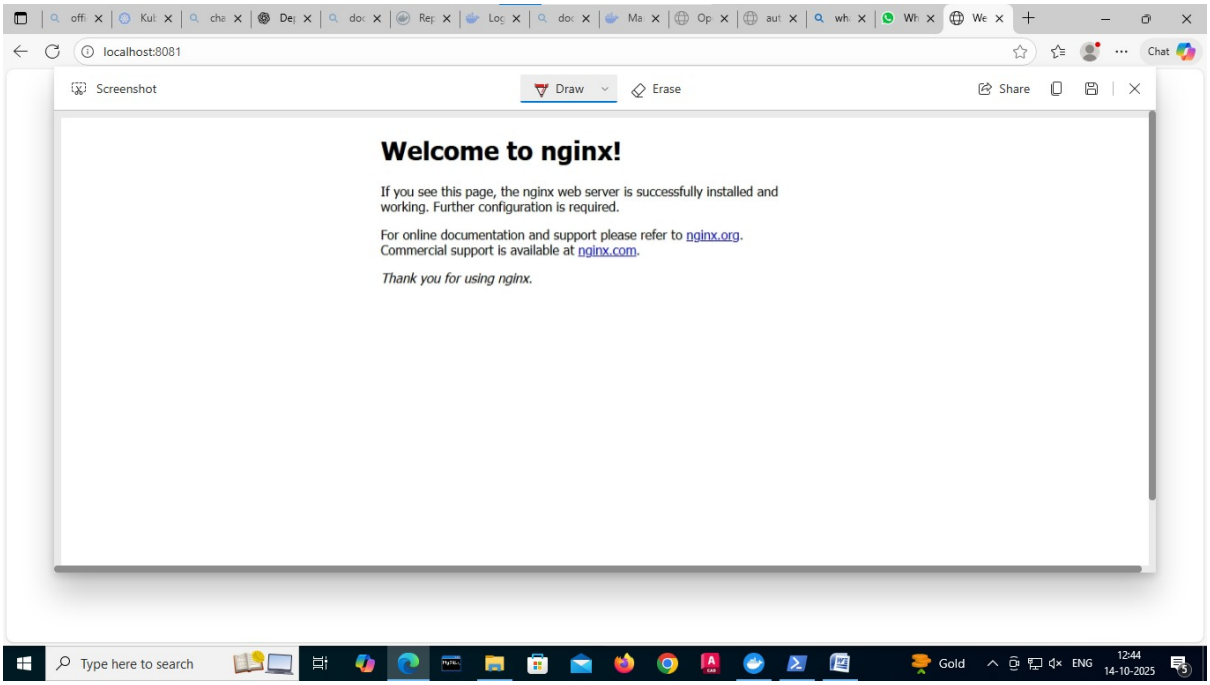
Step-6:Check the NGINX Service Details

- After creating the service, check its details to see which port Kubernetes assigned to it.

```
PS C:\Users\User> kubectl port-forward svc/mynginx 8081:80
Forwarding from 127.0.0.1:8081 -> 80
Forwarding from [::1]:8081 -> 80
Handling connection for 8081
Handling connection for 8081
Handling connection for 8081
Handling connection for 8081
```

Step-7: Open NGINX in the Browser

- Now that your service is exposed, you can open NGINX in your browser.



11. Jenkins-CI/CD

Setting Up Jenkins CI------using GitHub Webhook with Jenkins

Step 1: Take the authentication key from the ngrok and setup in ngrok terminal

```
  tcp             start a TCP tunnel
  tls             start a TLS endpoint
  update          update ngrok to the latest version
  version         print the version string

EXAMPLES:
# forward http traffic from assigned public URL to local port 80
ngrok http 80
# port 8080 available at baz.ngrok.dev
ngrok http --url baz.ngrok.dev 8080
# tunnel arbitrary TCP traffic to port 22
ngrok tcp 22
# secure your app with oauth
ngrok http 80 --oauth=google --oauth-allow-email=foo@foo.com

Paid Features:
  ngrok http 80 --url mydomain.com                          # run ngrok with your own custom domain
  ngrok http 80 --cidr-allow 2600:8c00::a03c:91ee:fe69:9695/32  # run ngrok with IP policy restrictions
  Upgrade your account at https://dashboard.ngrok.com/billing/subscription to access paid features

Upgrade your account at https://dashboard.ngrok.com/billing/subscription to access paid features

Flags:
  -h, --help      help for ngrok

Use "ngrok [command] --help" for more information about a command.

ngrok is a command line application, try typing 'ngrok.exe http 80'
at this terminal prompt to expose port 80.
C:\Windows\System32>ngrok config add-authtoken 34gKWhQDcoITj34K6eN73XoYG6J_58fBgmpjM5ikZVdKVDyCe
```

Step-2: Execute the following command using the port number on which Jenkins is running

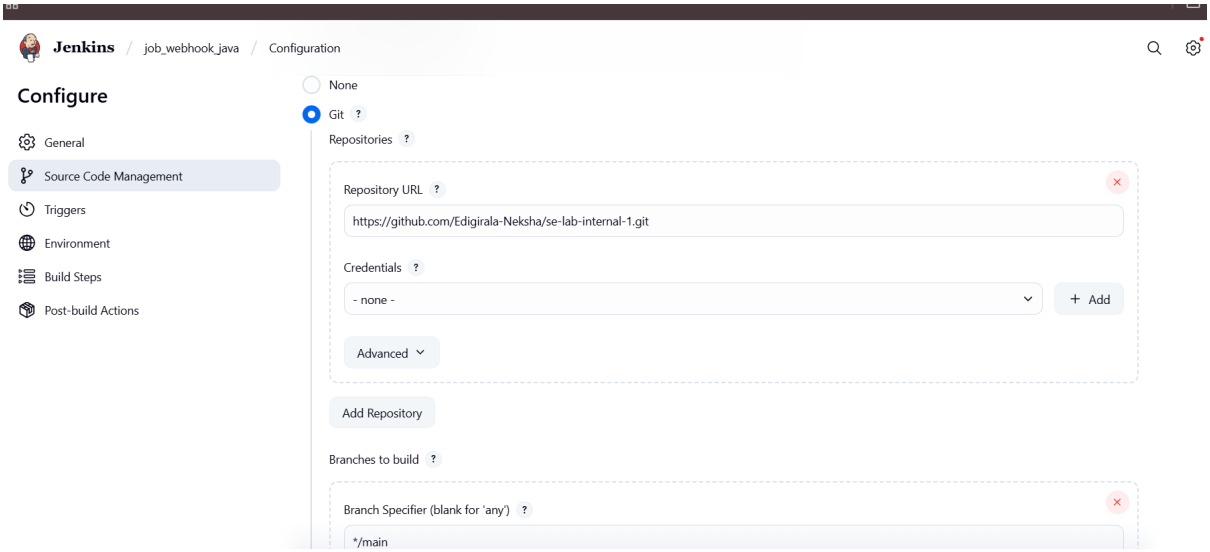
```
C:\Windows\System32>ngrok.exe http 8888
```

- Following output will be given:

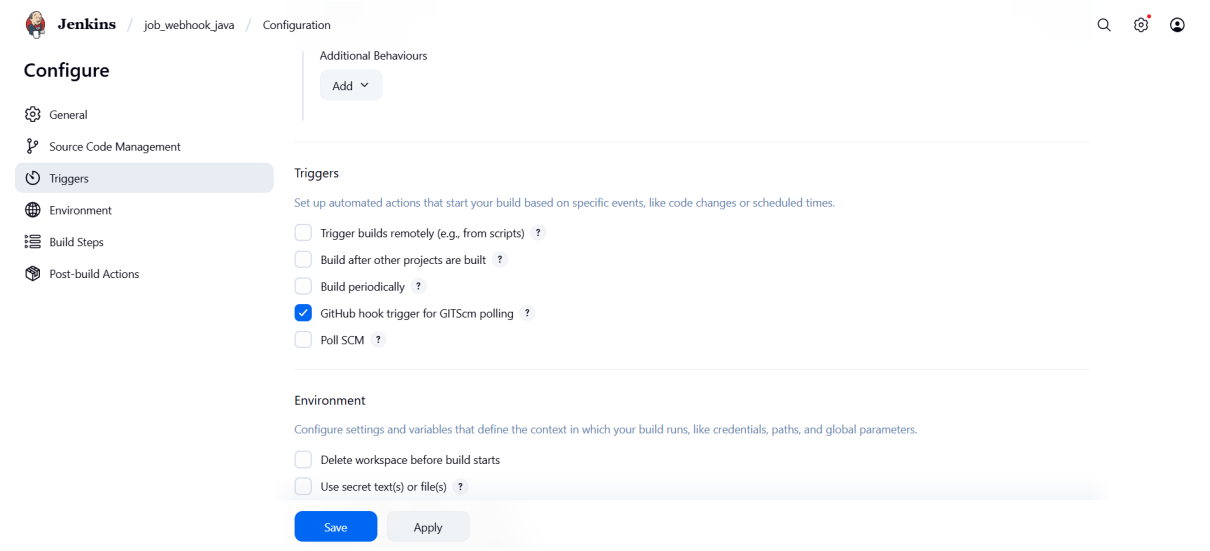
```
ngrok                                                                                    (Ctrl+C to quit)

◆  Block threats before they reach your services with new WAF actions → https://ngrok.com/r/waf

Session Status                online
Account                       Neksha Edigirala (Plan: Free)
Update                        update available (version 3.32.0, Ctrl-U to update)
Version                       3.24.0-msix
Region                        India (in)
Latency                       147ms
Web Interface                 http://127.0.0.1:4040
Forwarding                    https://corkier-darla-handsome.ngrok-free.dev -> http://localhost:8888

Connections                   ttl     opn     rt1     rt5     p50     p90
                              2       0       0.00    0.00    30.28   30.47

HTTP Requests
-------------

11:35:59.377 IST POST /github-webhook/          200 OK
11:34:29.479 IST POST /github-webhook/          200 OK
```

Go to Jenkins:

Step-3: Create the Jenkins job in the source code management select the git and enter git repo url and make sure the branch is same (i.e., main)

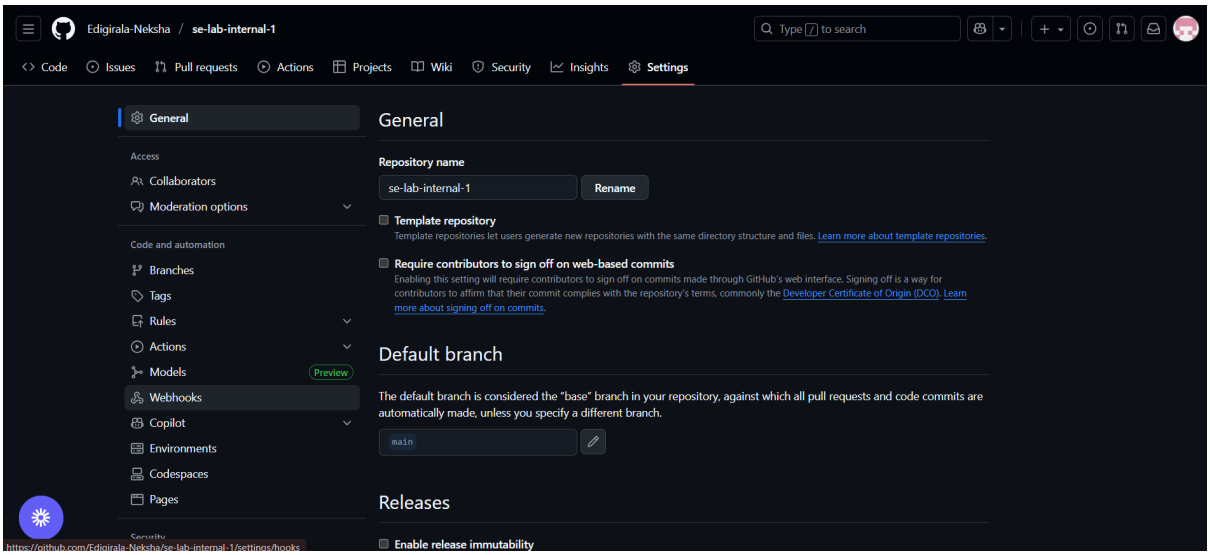


Step-4: In the triggers section select “Github hook trigger for GITScm polling”



Click on apply and save

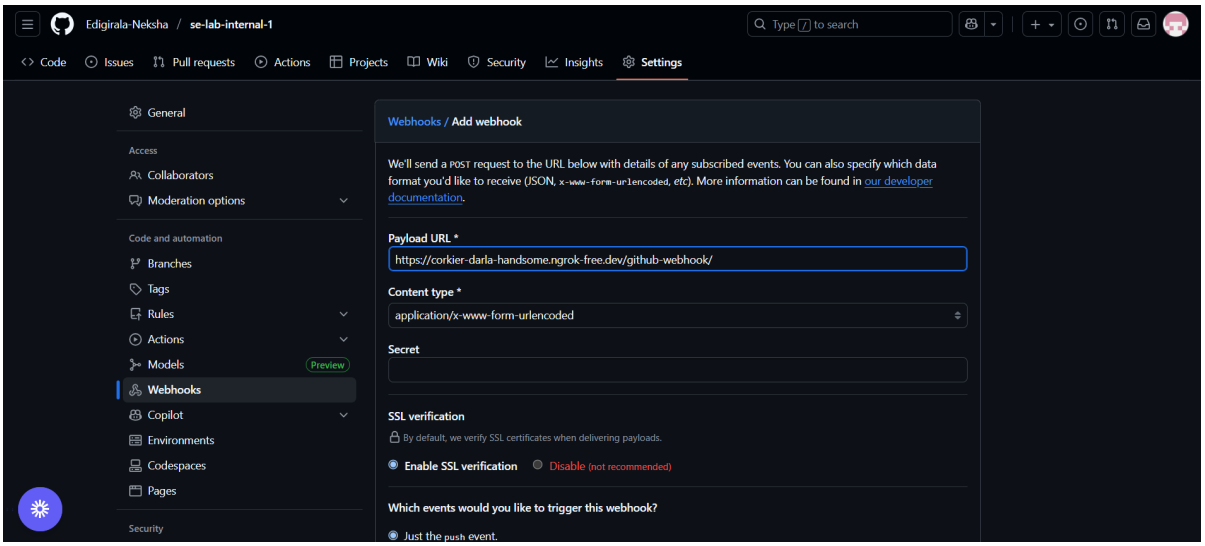
Step-6: open the git hub repo open setting of repo and then go to webhooks



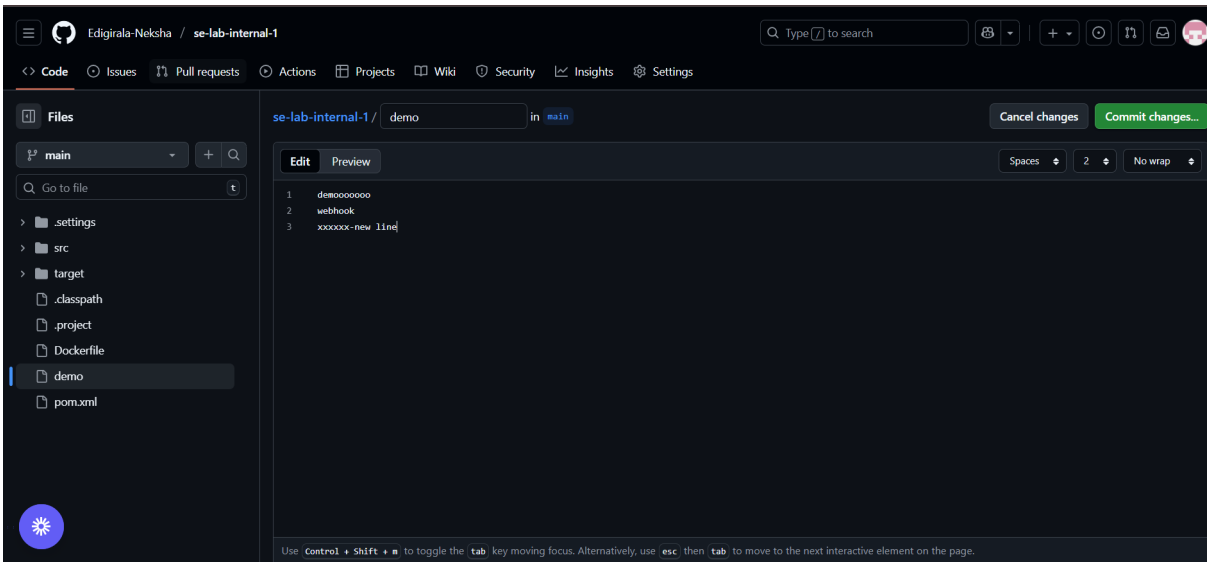
Step-7: Click on add a webhook and take the forwarding URL from ngrok and paste in payload URL and add /github-webhook/ along with the forwarding url

Forwarding URL: https://corkier-darla-handsome.ngrok-free.dev

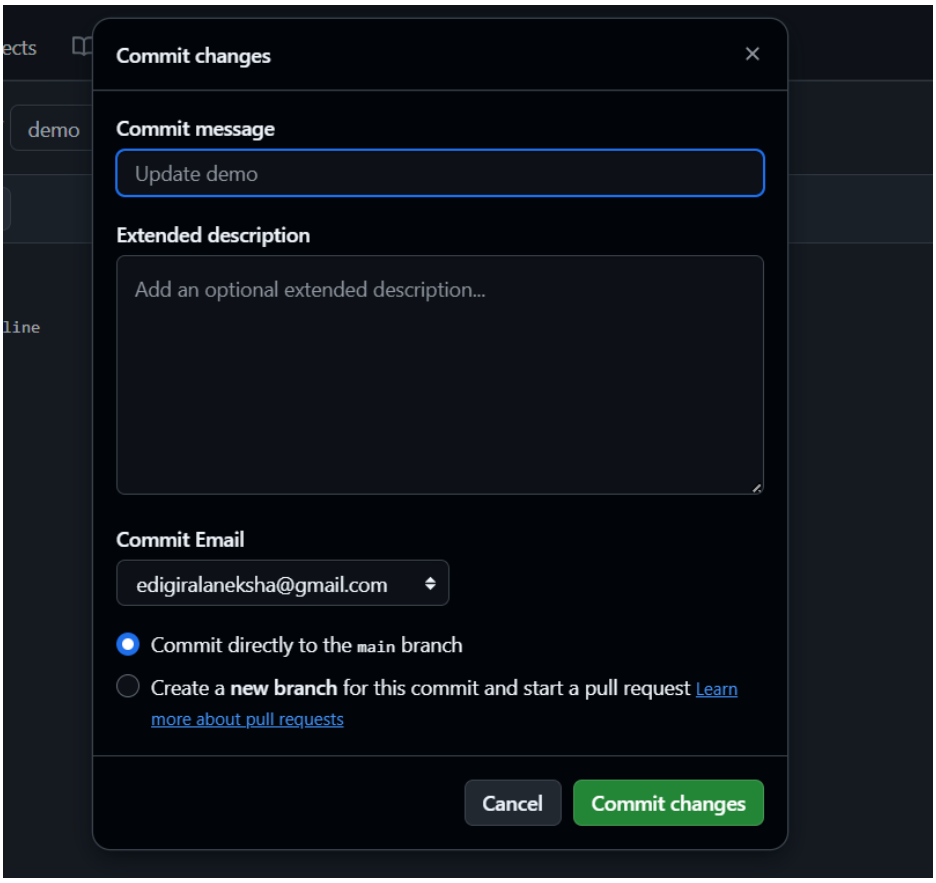
Payload url: https://corkier-darla-handsome.ngrok-free.dev/github-webhook/



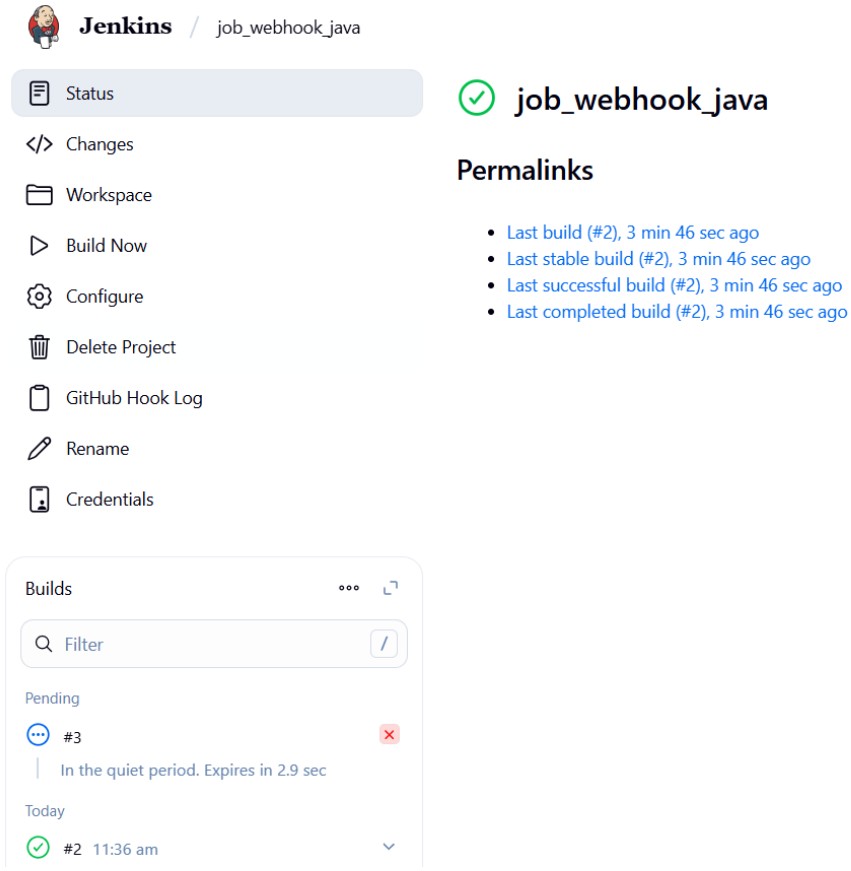
Step 8: make changes in the files in github

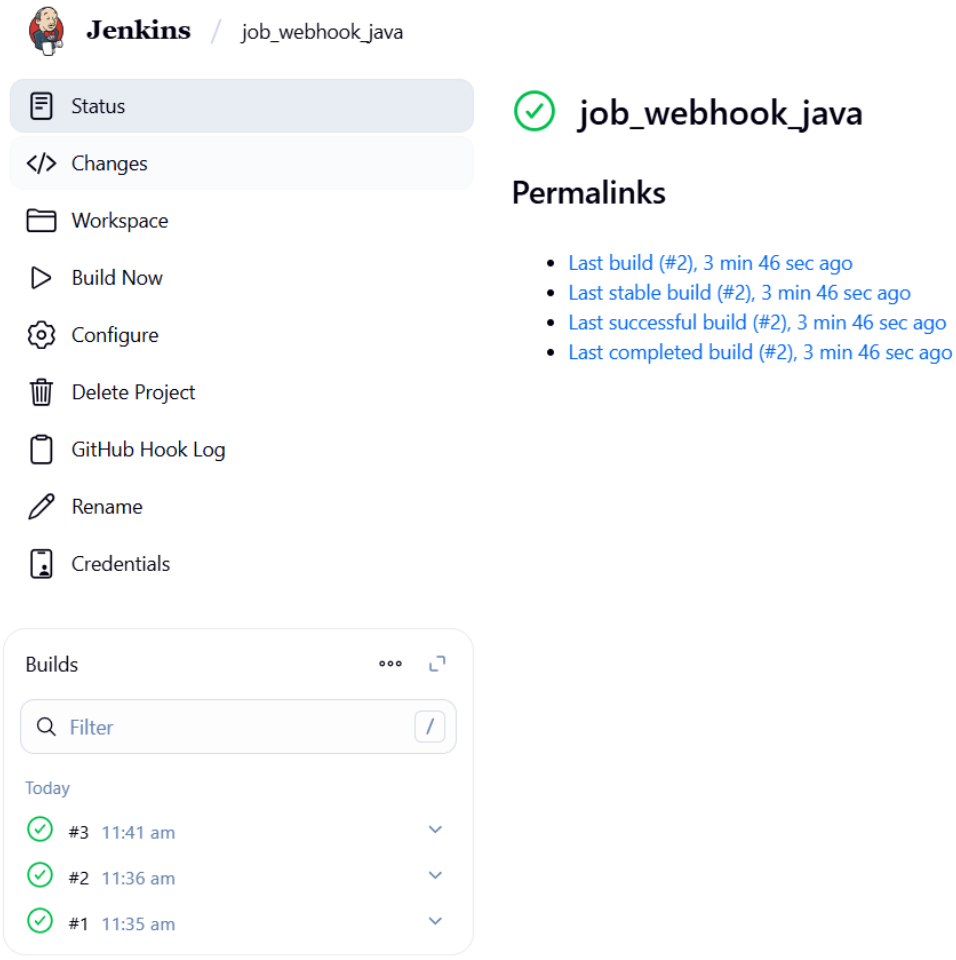


Step 9: click on commit changes

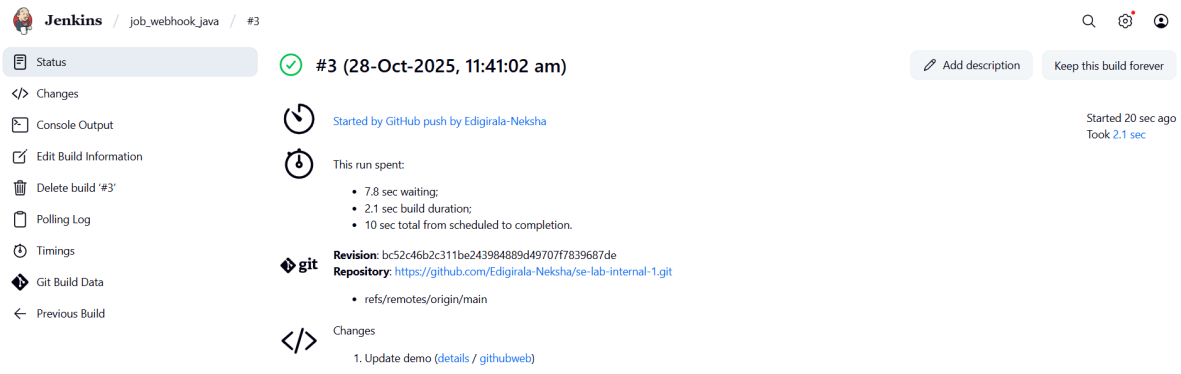


Step 10: open Jenkins the build will start automatically





You can check status : started by git hub push



Setting Up Jenkins Email Notification Setup (Using Gmail with AppPassword)

Step-1: Creation of app password

Gmail: Enable App Password (for 2-Step Verification)

ii. Enable 2-Step Verification

iii. Generate App Password for Jenkins

- Go to:
    - Security → App passwords
- Select:
    - **App**: Other (Custom name)
    - **Name**: Jenkins-Demo
- Click **Generate**
- Copy the **16-digit app password**
    - Save it in a secure location (e.g., Notepad)

2. Jenkins Plugin Installation

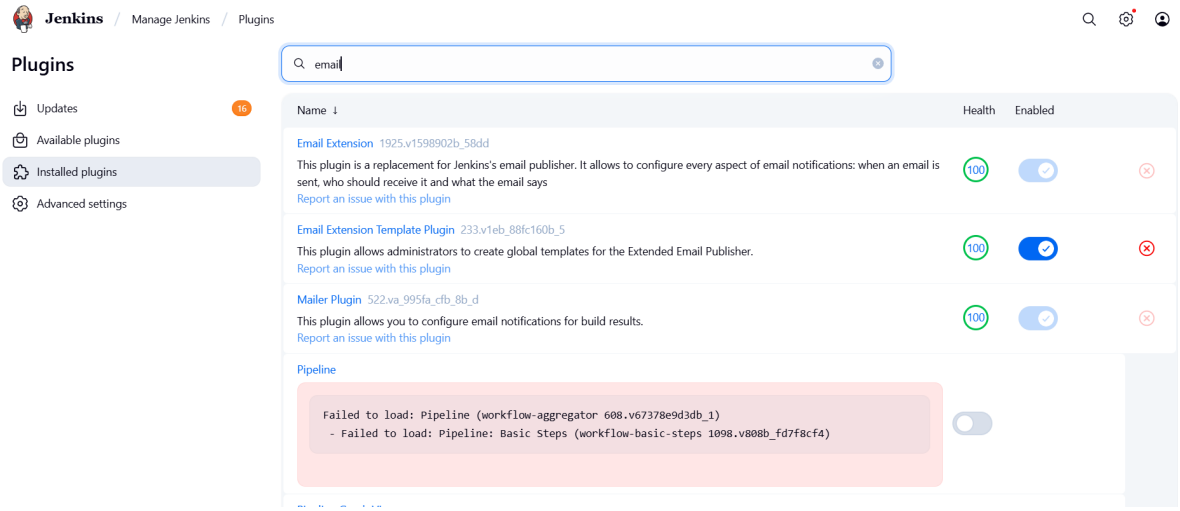
i. Open Jenkins Dashboard

ii. Navigate to:

- Manage Jenkins → Manage Plugins

iii. Install Plugin:

- Search for and install:
    - Email Extension Plugin



3. Configure Jenkins Global Email Settings

Go to:

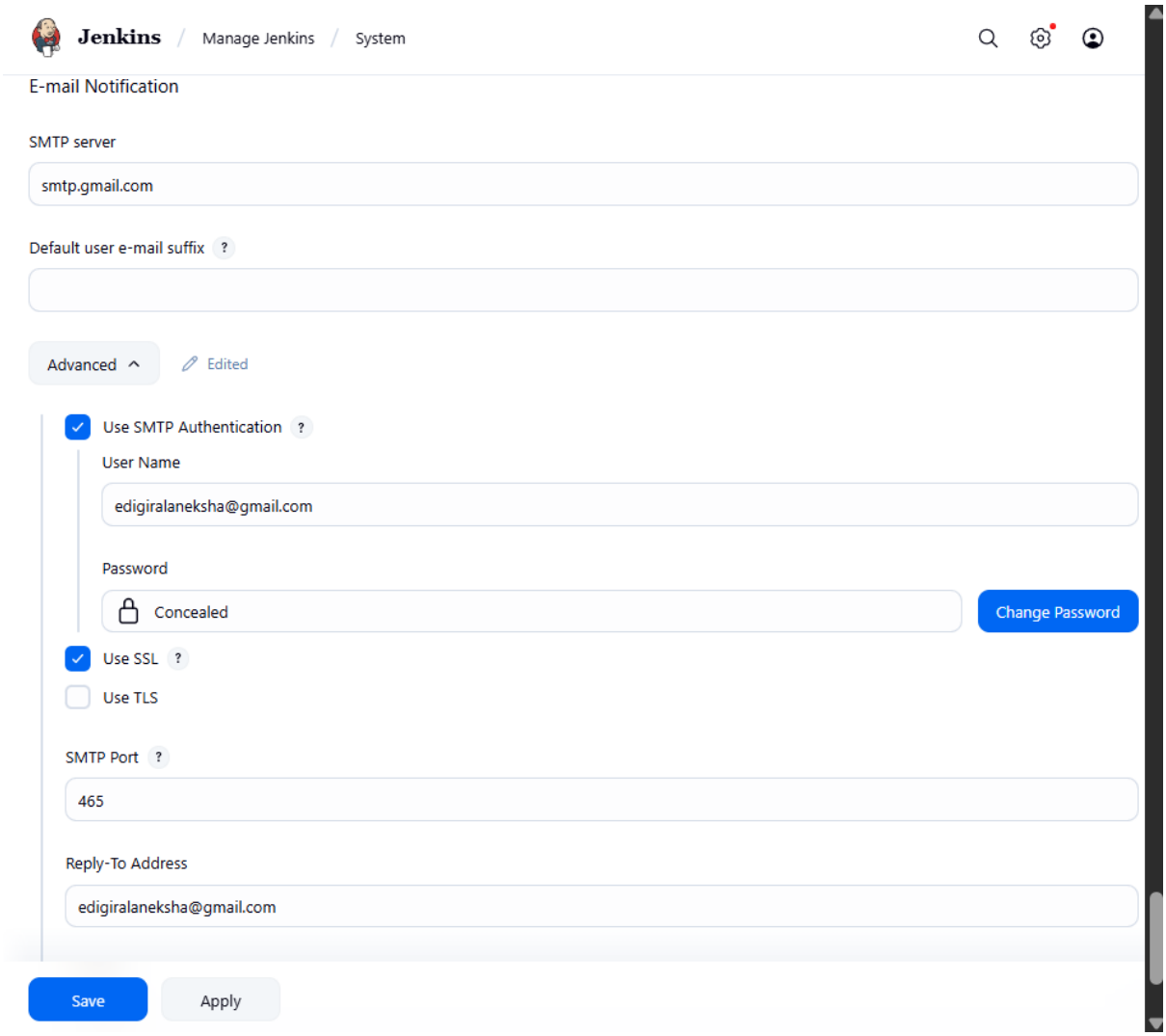
- Manage Jenkins → Configure System

A. E-mail Notification Section

Field	Value
SMTP Server	smtp.gmail.com
Use SMTP Auth	✅ Enabled
User Name	Your Gmail ID (e.g., archanareddykmit@gmail.com)
Password	Paste the 16-digit App Password
Use SSL	✅ Enabled
SMTP Port	465
Reply-To Address	Your Gmail ID (same as above)

➤ Test Configuration

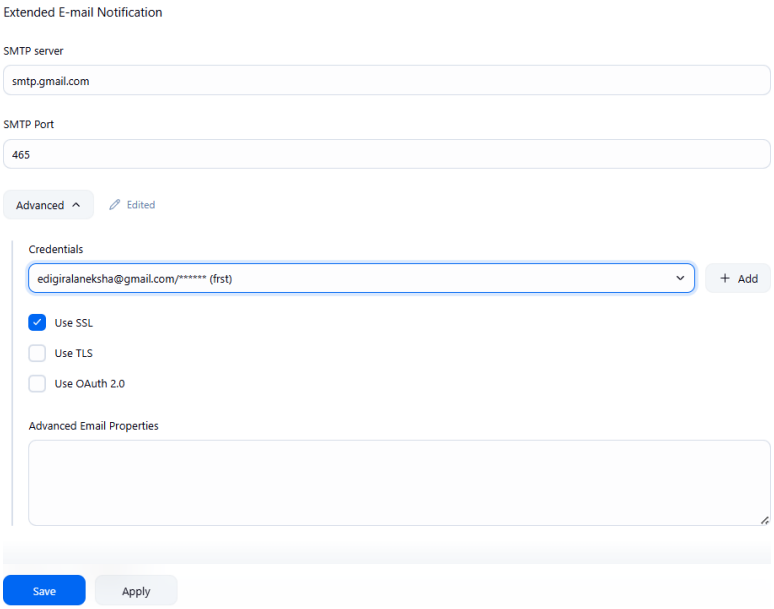
- Click: Test configuration by sending test e-mail
- Provide a valid email address to receive a test mail
- ✅ Should receive email from Jenkins

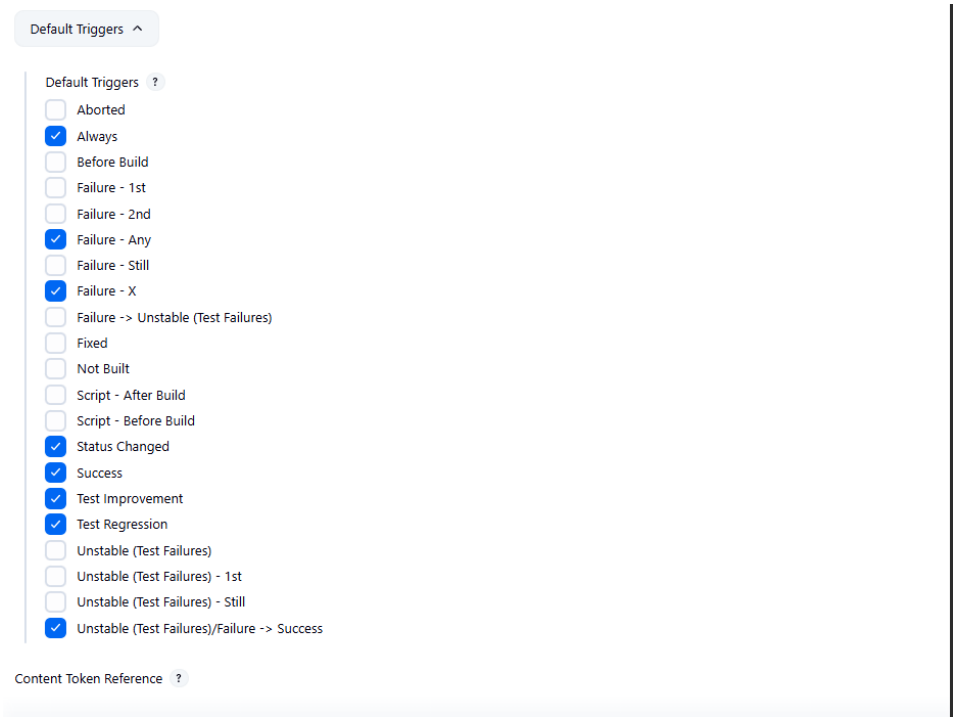


-

B. Extended E-mail Notification Section

Field	Value
SMTP Server	smtp.gmail.com
SMTP Port	465
Use SSL	✅ Enabled
Credentials	Add Gmail ID and App Password as Jenkins credentials
Default Content Type	text/html or leave default
Default Recipients	Leave empty or provide default emails
Triggers	Select as per needs (e.g., Failure)

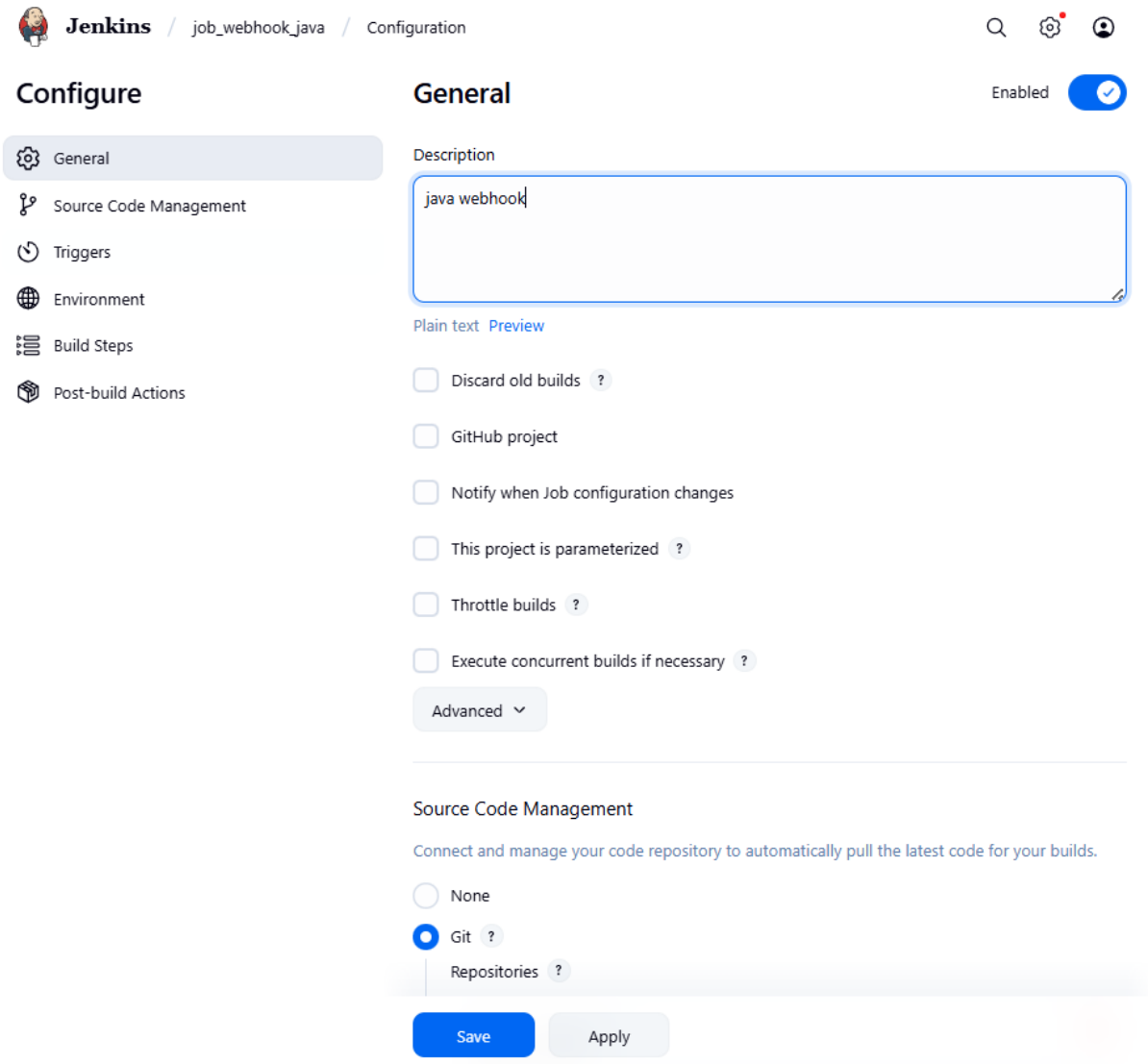




4. Configure Email Notifications for a Jenkins Job

i. Go to:

- Jenkins → Select a Job → Configure



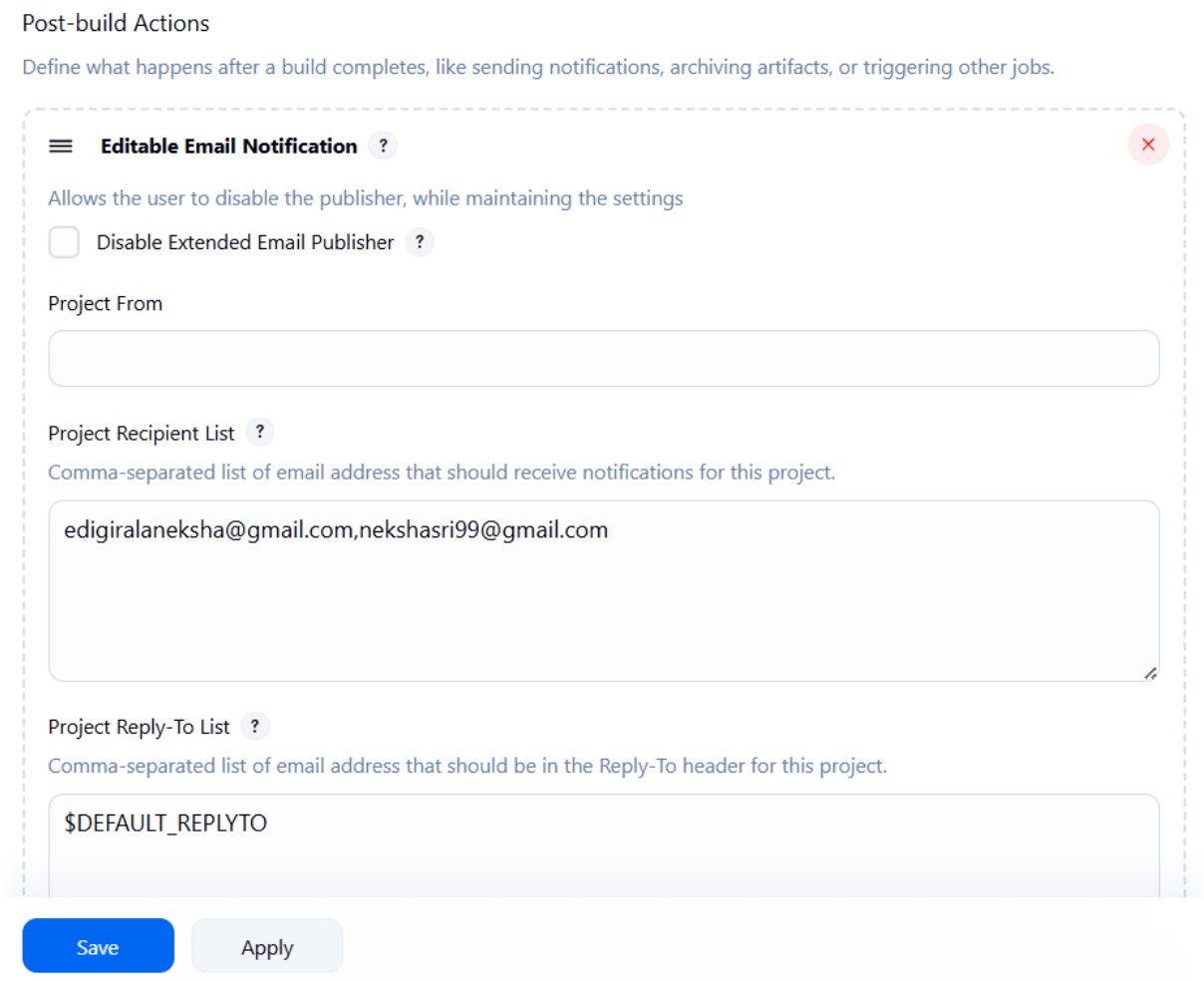
ii. In the Post-build Actions section:

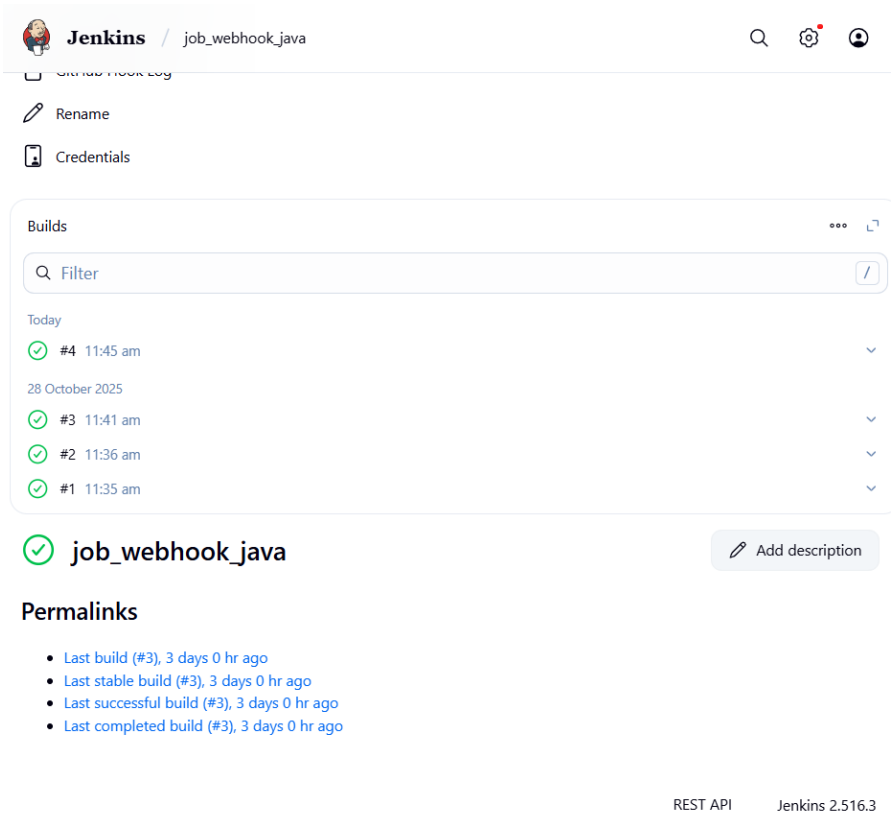
- Click: Add post-build action → **Editable Email Notification**

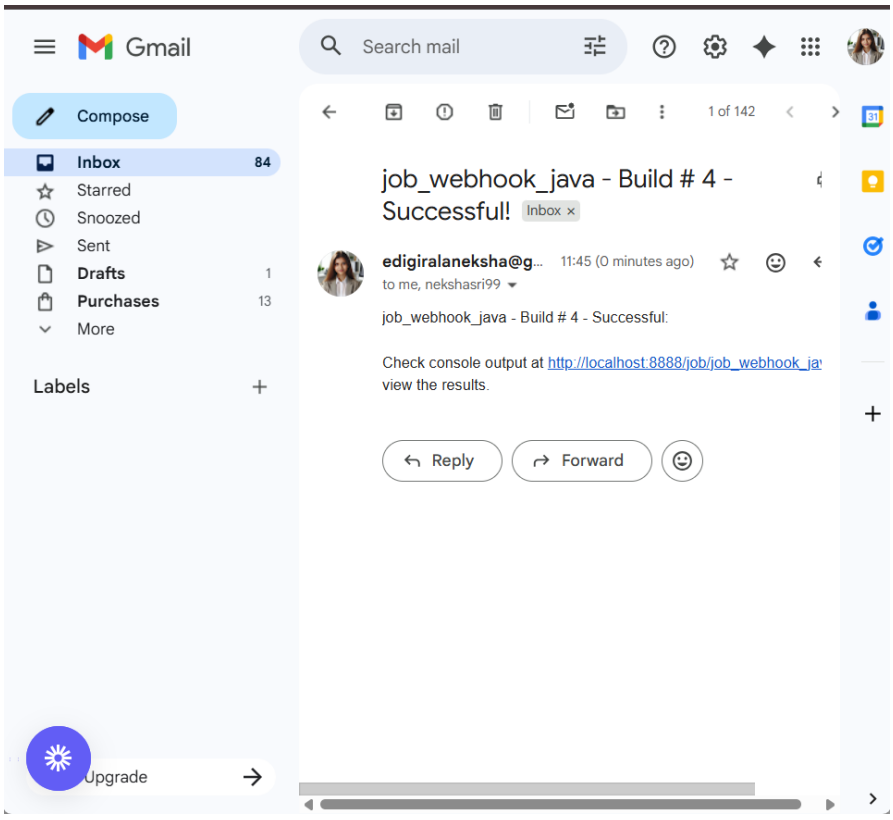
A. Fill in the fields:

Field	Value
Project Recipient List	Add recipient email addresses (comma-separated)
Content Type	Default (text/plain) or text/html
Triggers	Select events (e.g., Failure, Success, etc.)
Attachments	(Optional) Add logs, reports, etc.

iii. Click Save



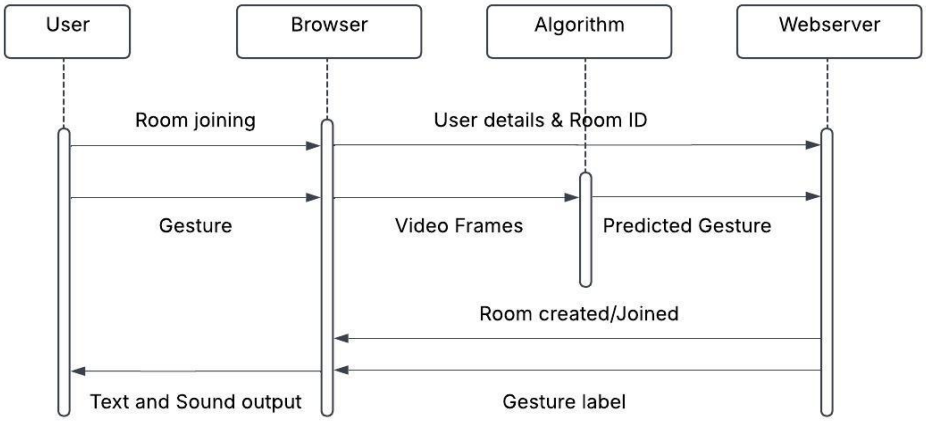




TUNEORA – A Music Web

App

1. Sequence Diagram:
A sequence diagram shows how objects interact in a particular scenario of a use case.
It focuses on the time order of messages exchanged between different components in a system.



2. Class Diagram:
A class diagram represents the static structure of a system by showing classes, their attributes, methods, and relationships.
It is mainly used for object-oriented design and modeling data structures.

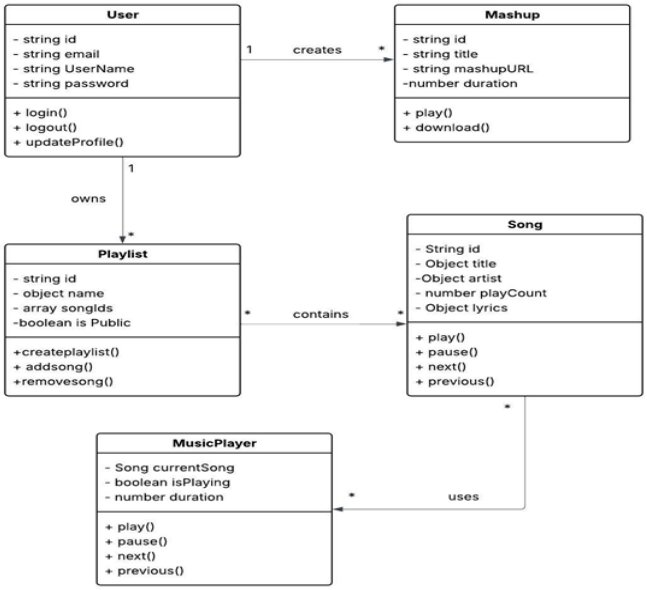
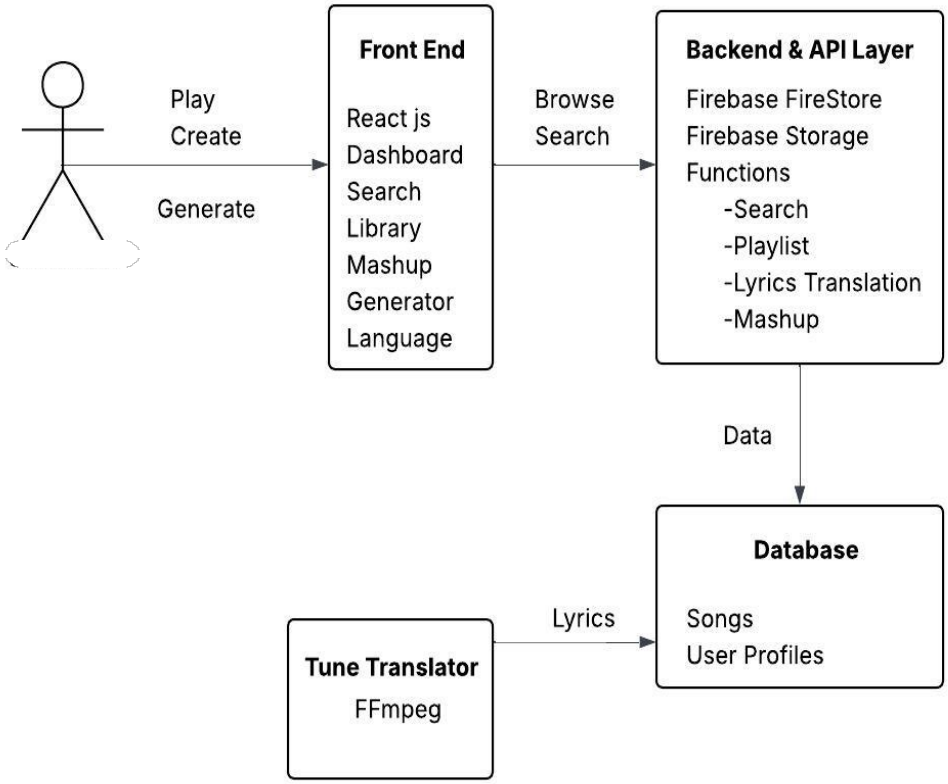


Fig 5: Class Diagram for TuneOra

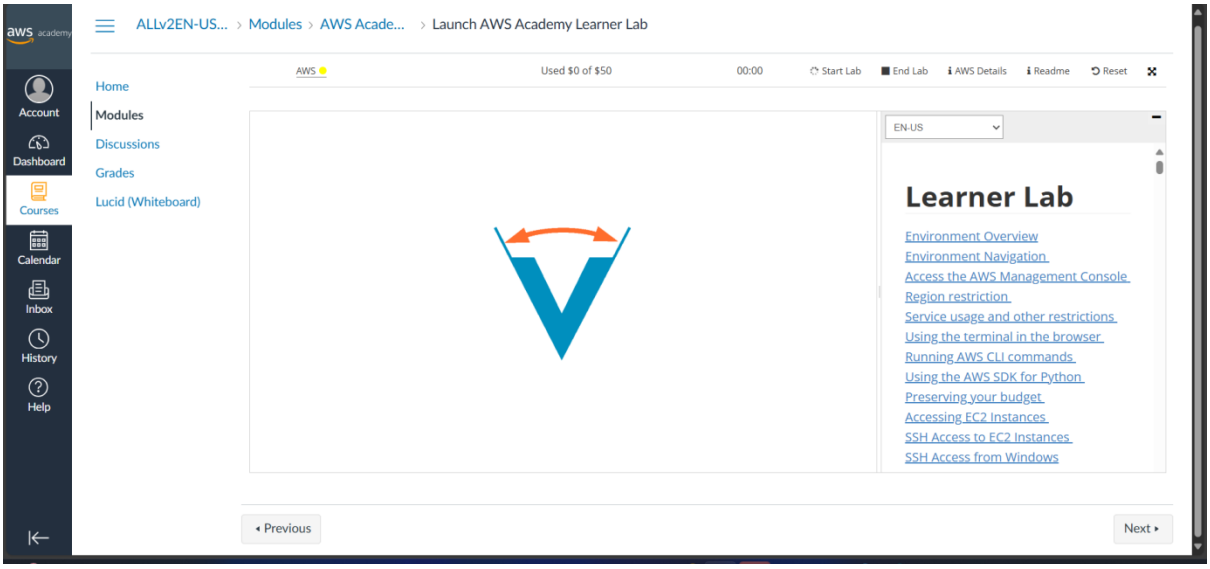
3. Component Diagram:
A component diagram illustrates how different software components are connected and interact to form a complete system.
It helps visualize the organization and dependencies among modules or subsystems.



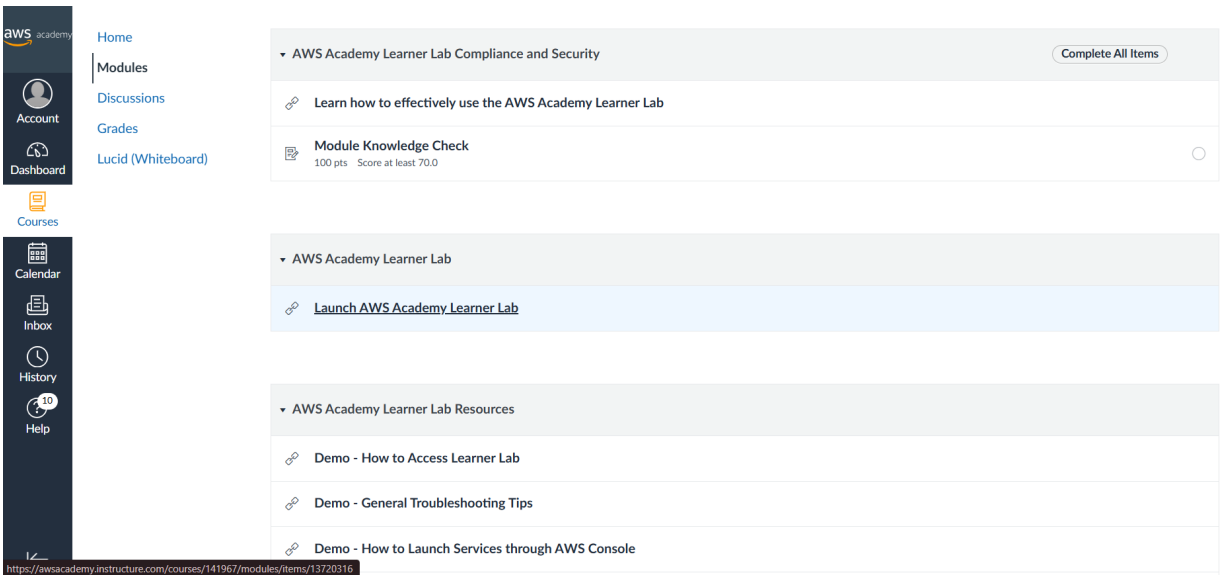
12.Creation of virtual machine for Ubuntu OS and Deploying the web application

DEPLOYMENT OF INDEX.HTML USING EC2 INSTANCE in AWS

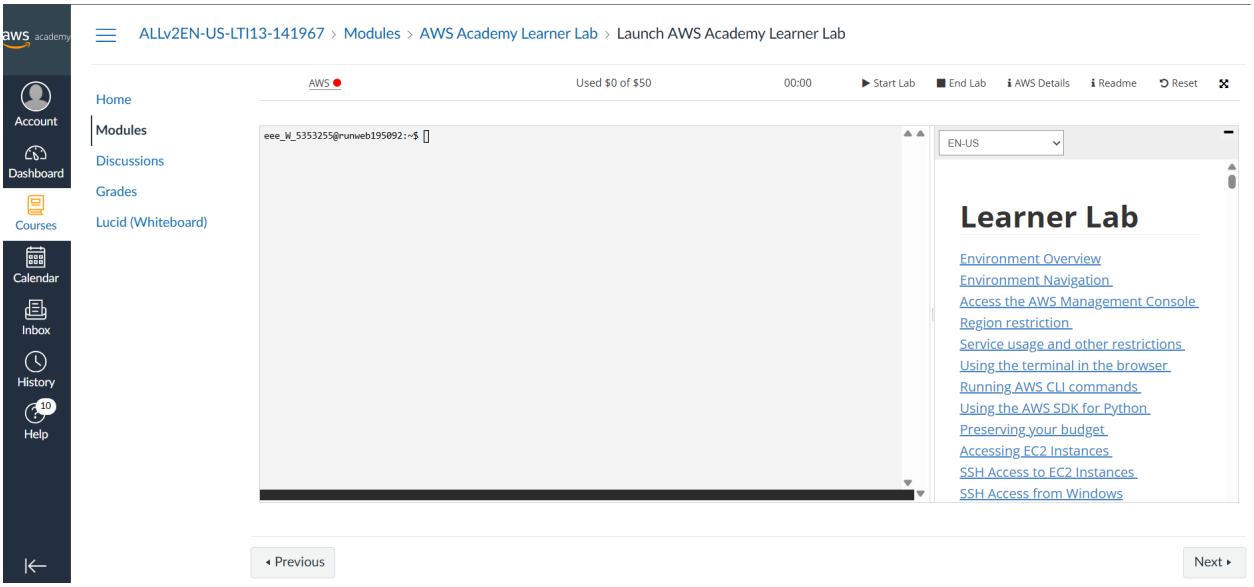
Step 1: Click on Modules.



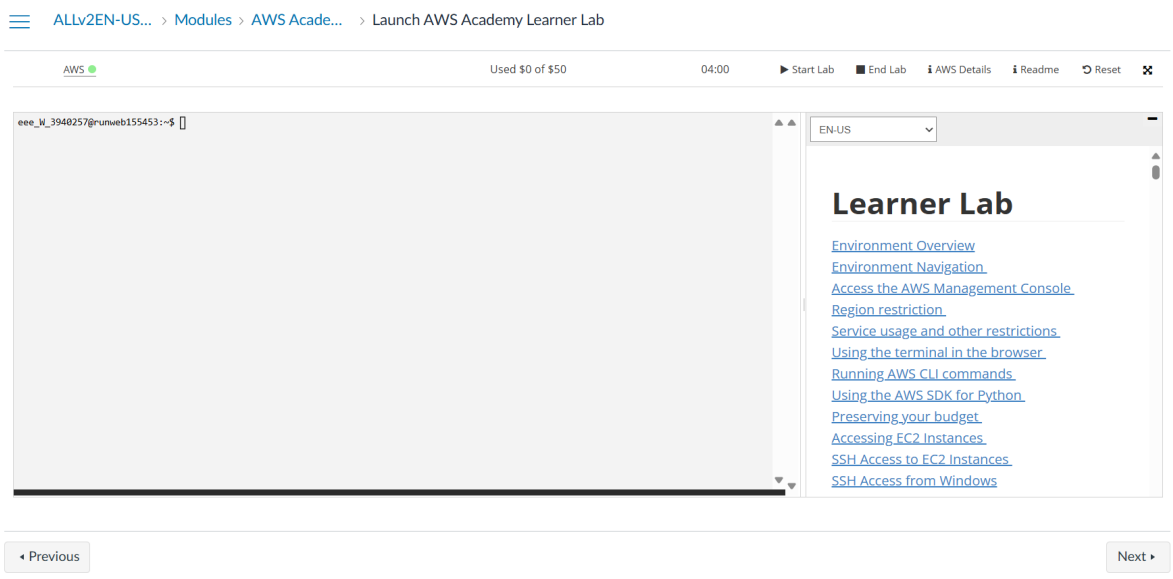
Step 2: Scroll down and select Lunch AWS Academy Lab



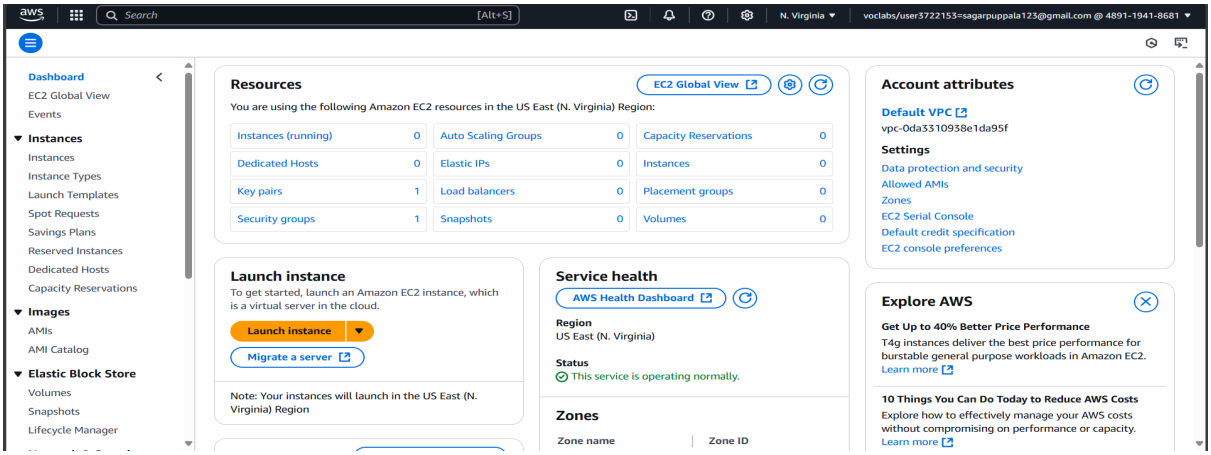
Step 3: click on start lab



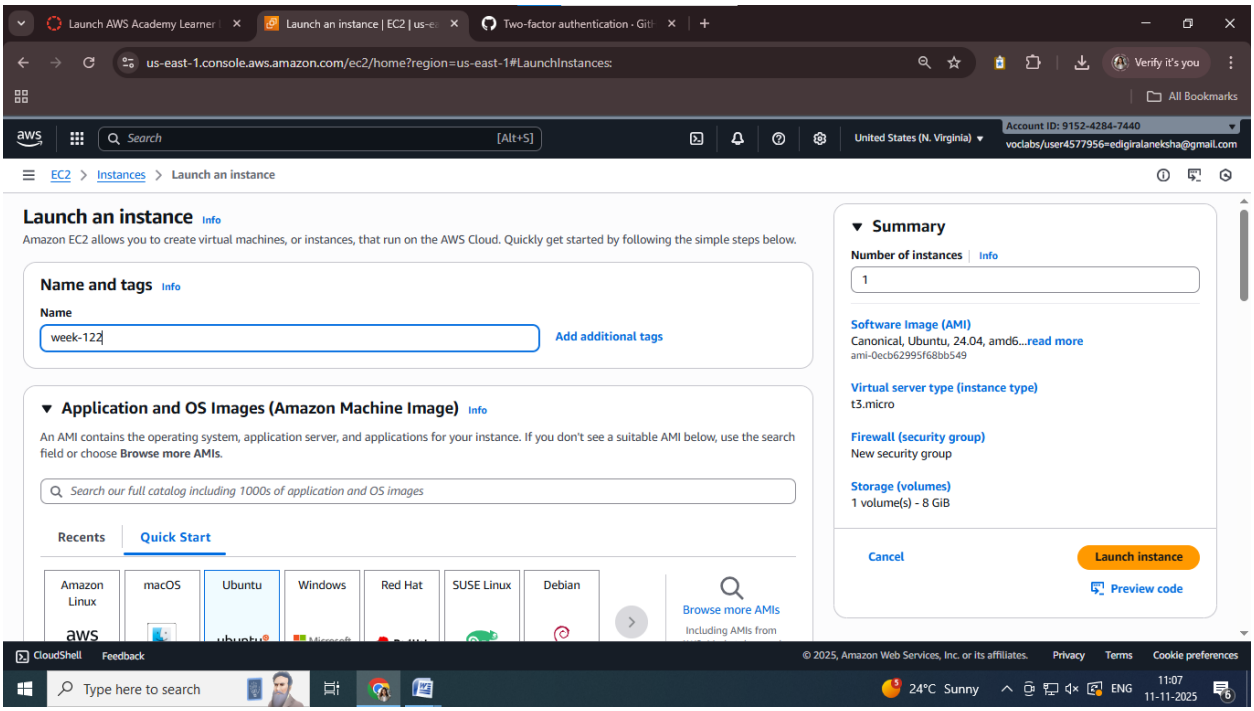
Step 4: click on AWS and in the services select EC2



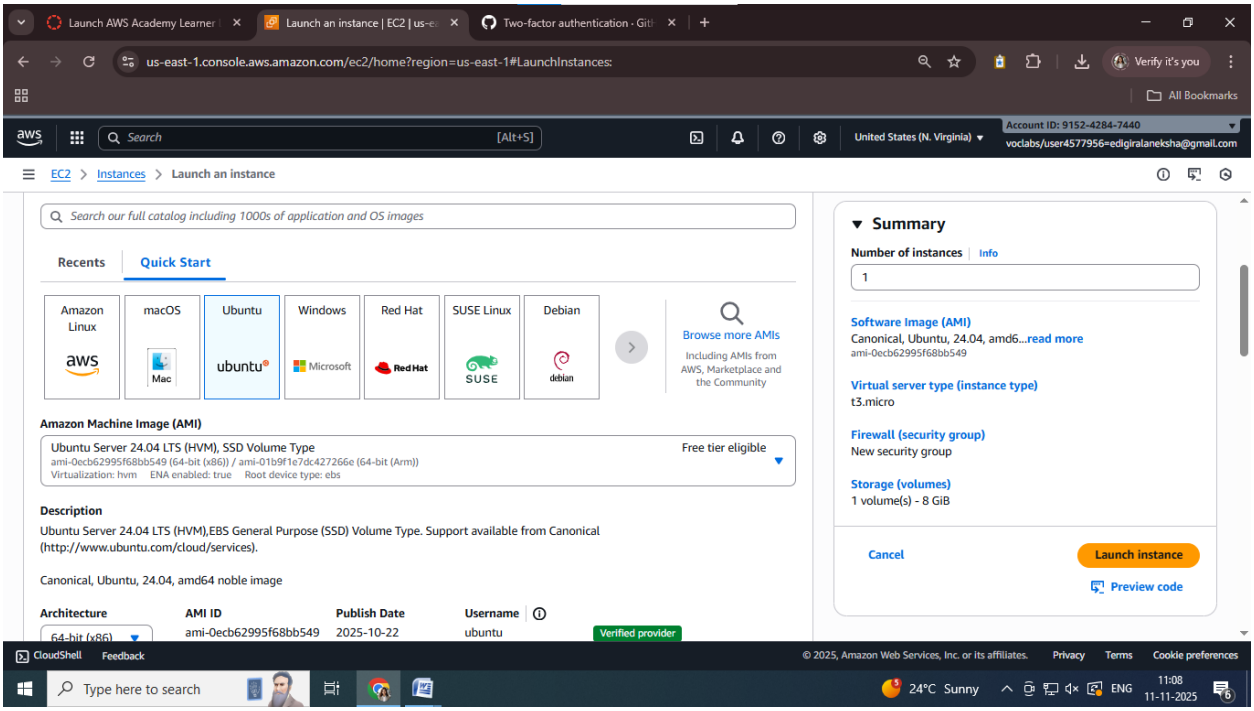
Step 5: select instances and select instance click on launch instance



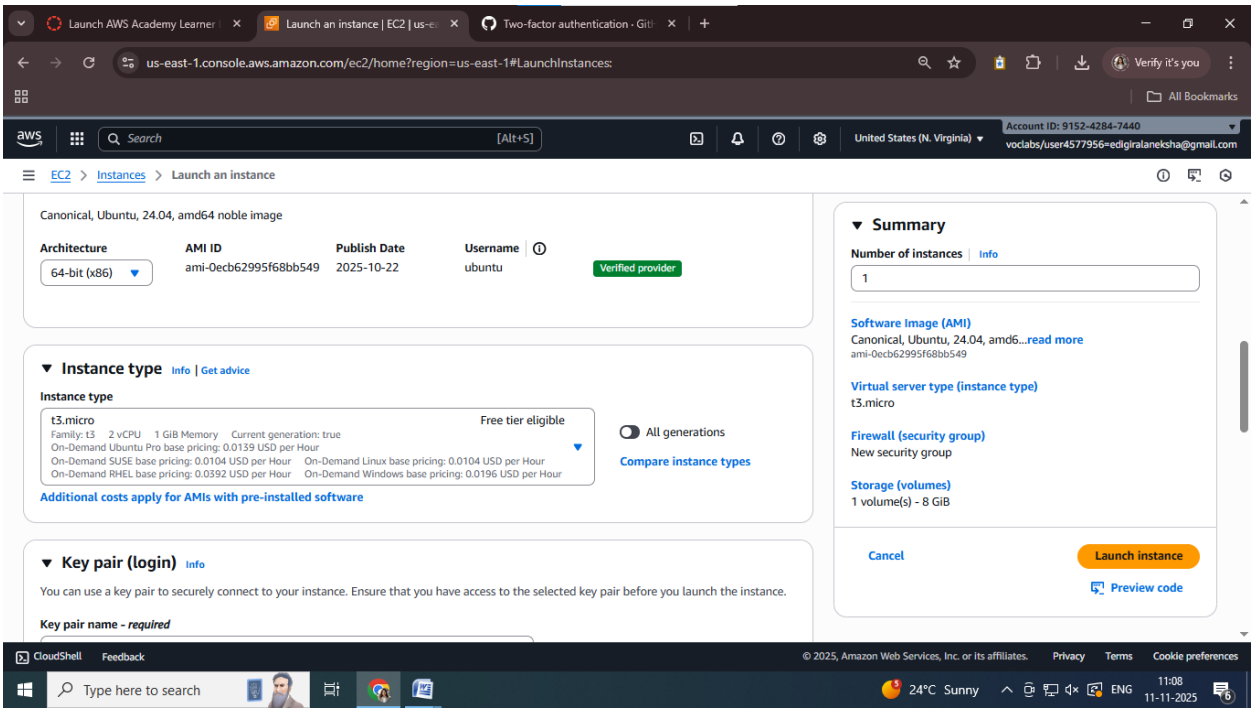
Step 6: Give the name of the machine “week-122”



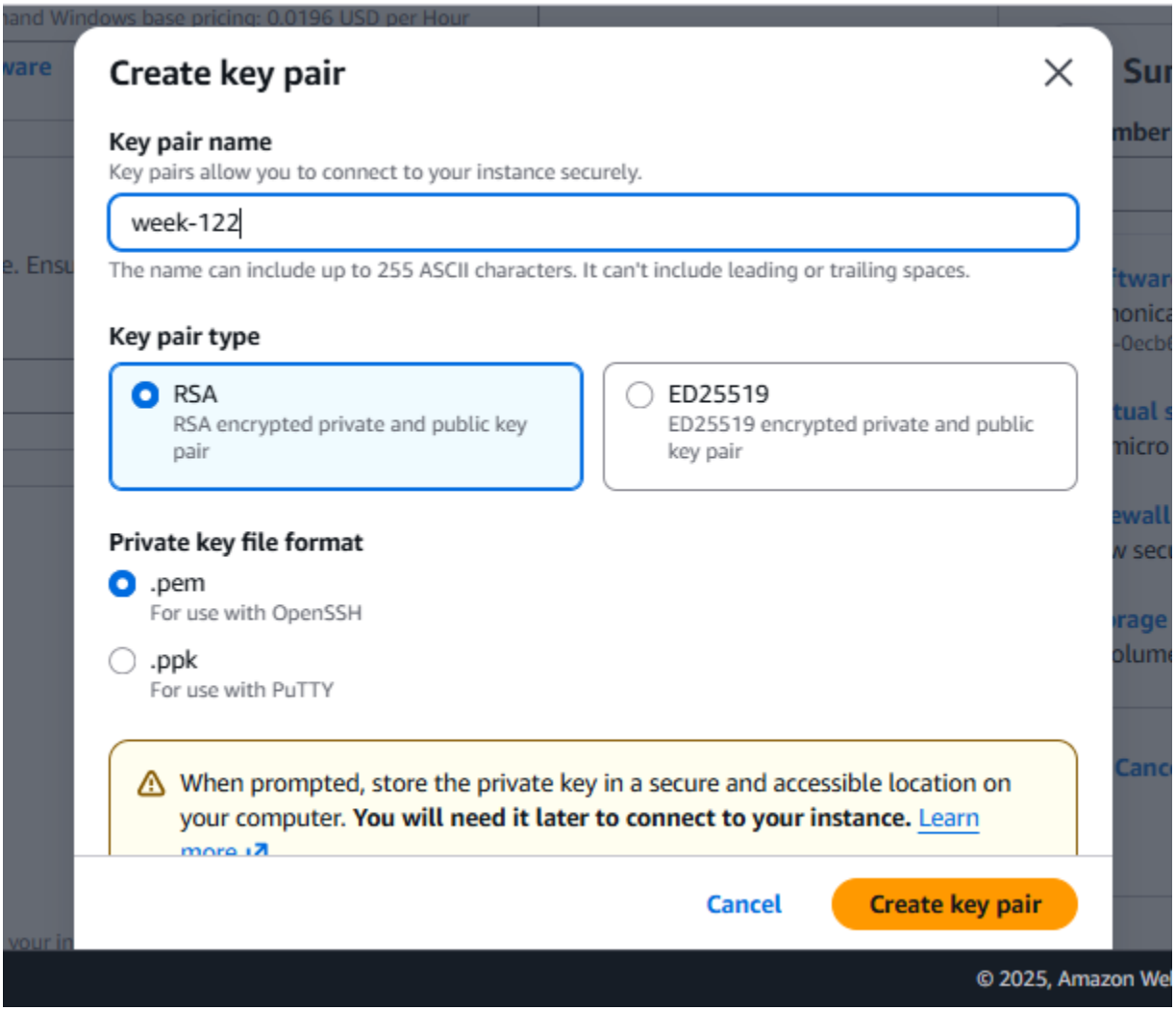
Step 6: Select the ubuntu server



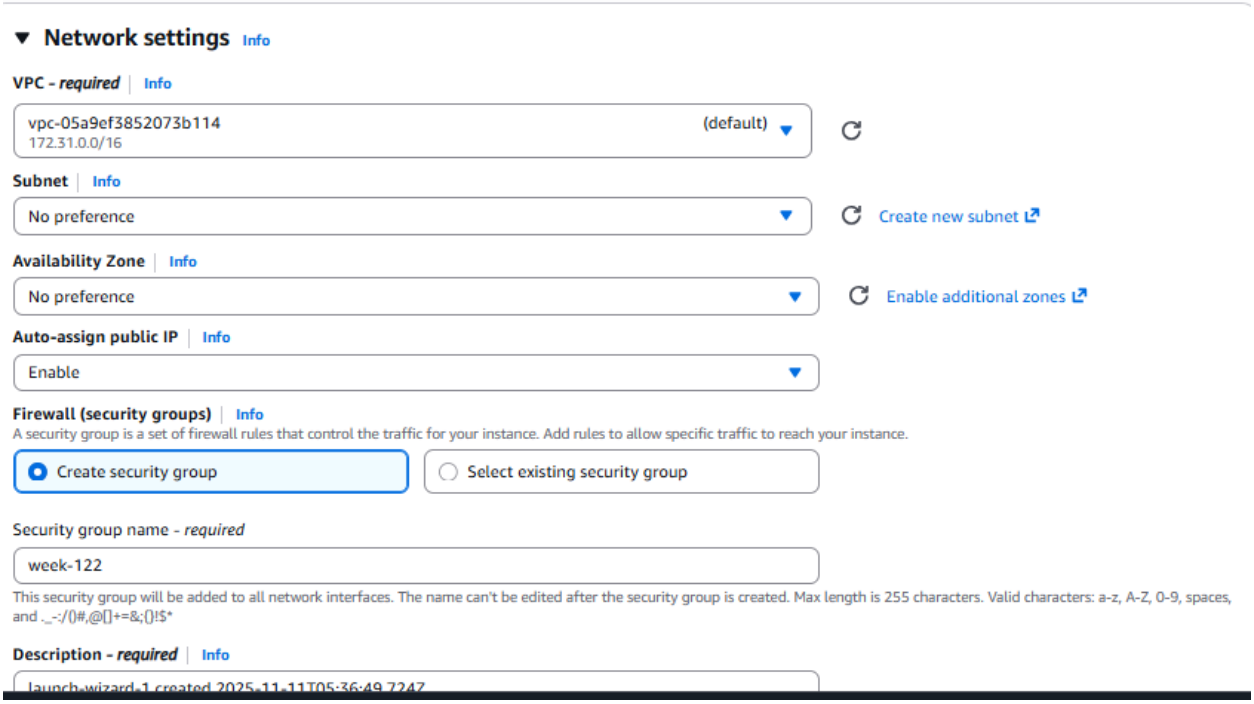
Step 7: select architecture as 64-bit and instance type as t3.micro(i.e., they are free)



Step 8: Create a new keypair and select type as RSA and .pem option and click on create key pair



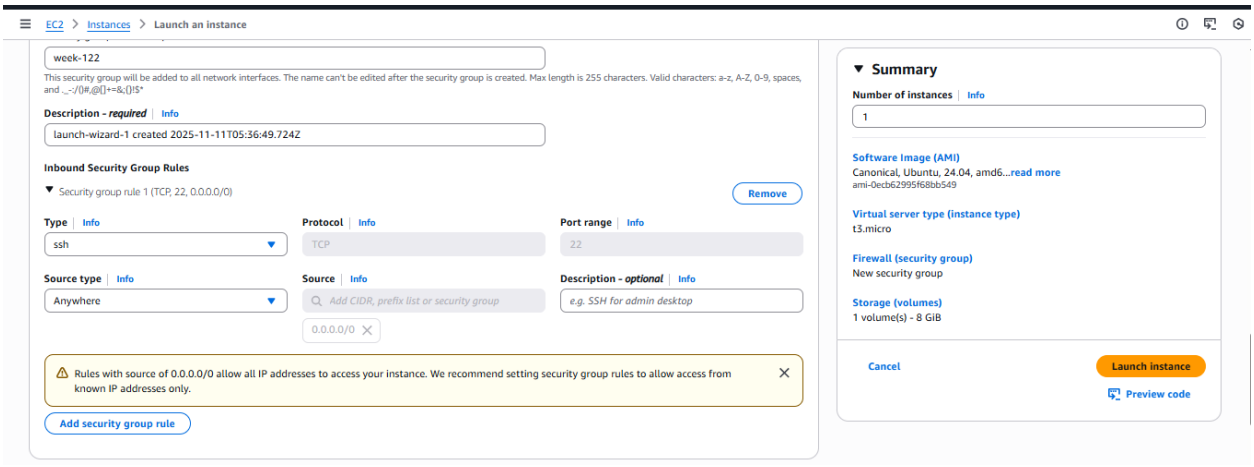
Step 9: In network settings select “create security group” and give the security group name



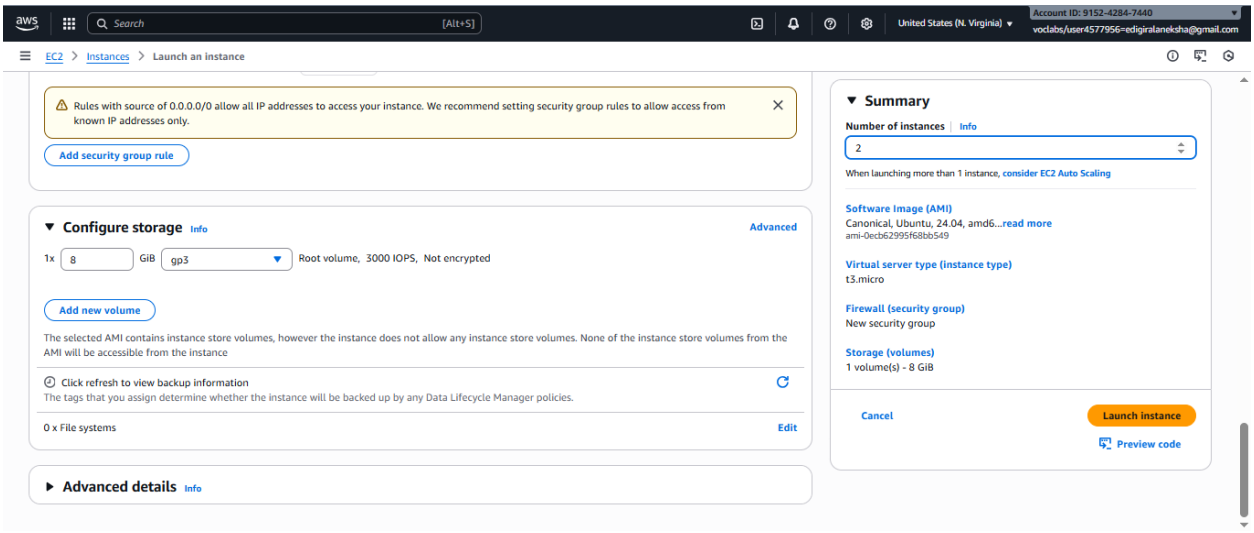
Step 10: Click on edit button on the top right corner and select

Type: ssh

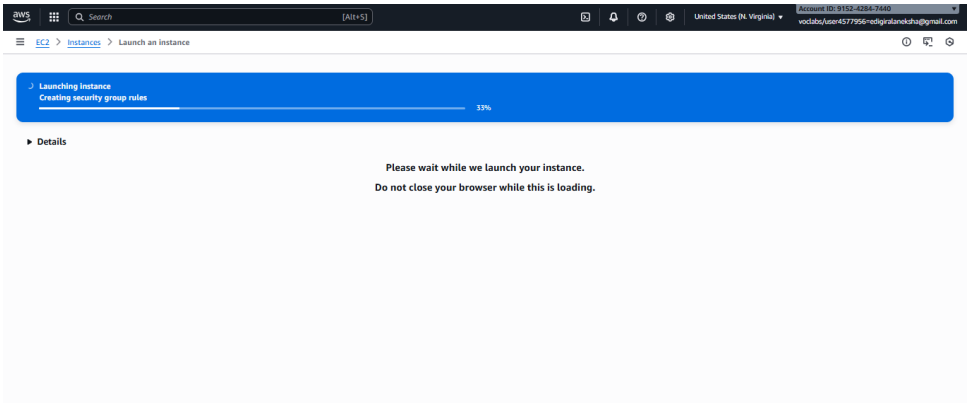
Source: anywhere

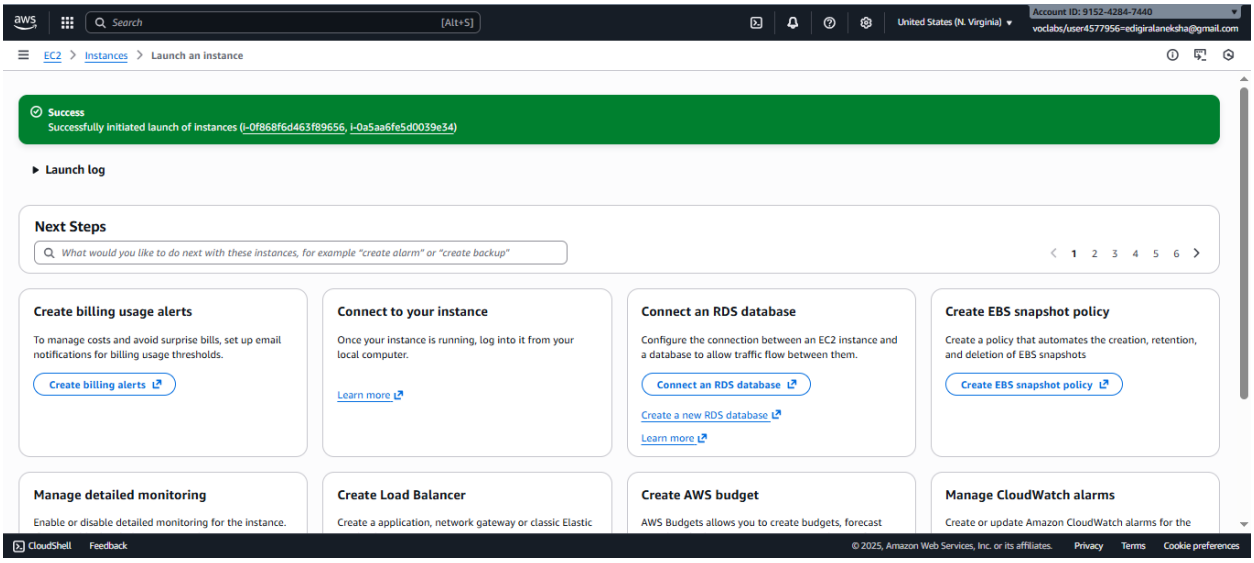


Step 11: in configure storage give 8GB and give number of instances as 2 and click on launch instance

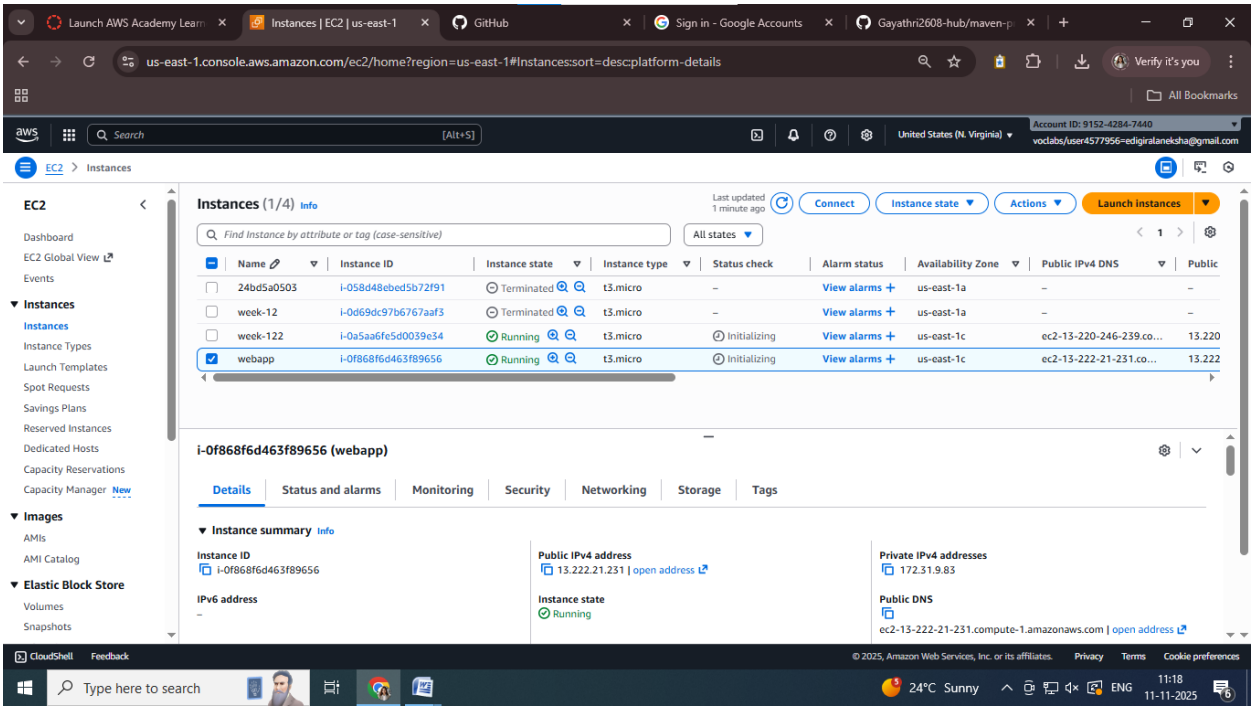


Step 12: The launching of instance will start and successful message will be shown

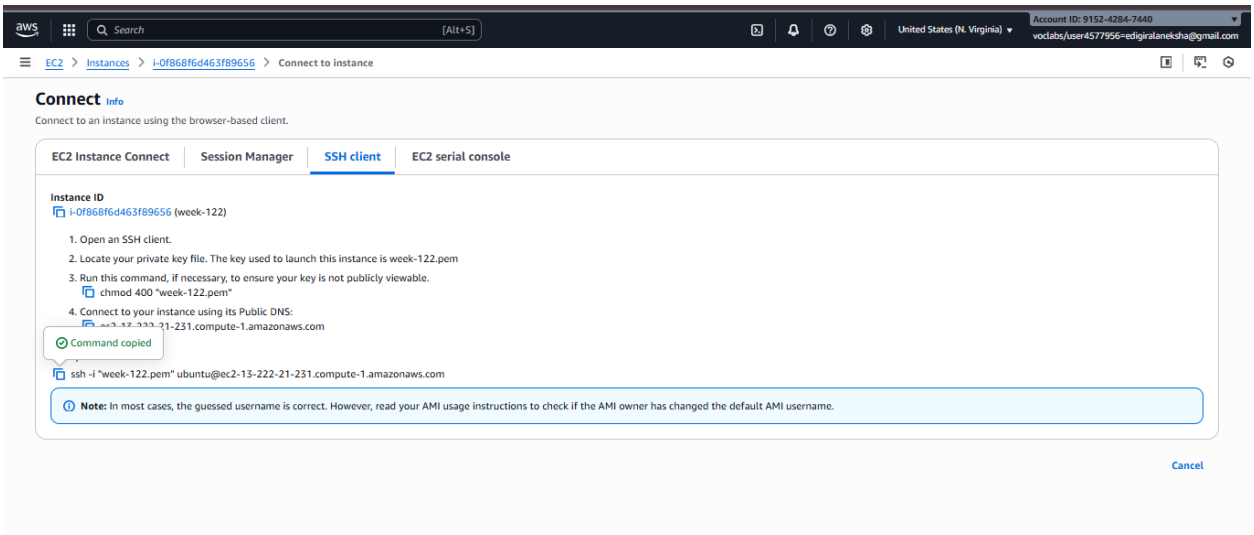




Step 13: In the instances the created ones will be shown, you can also rename the instance , changed week-1222 to “webapp”



Step 14: click on connect and select “SSH Client” and copy the ssh command



Step 15: Navigate to the path where the file with .pem extension is present(week-122.pem) and paste the command

```
PS C:\Users\User\downloads> ssh -i "week-122.pem" ubuntu@ec2-13-222-21-231.compute-1.amazonaws.com
The authenticity of host 'ec2-13-222-21-231.compute-1.amazonaws.com (13.222.21.231)' can't be established.
ED25519 key fingerprint is SHA256:NEGegcHQjt8om/1AVLSqfmafnMphv5Ad4AlMwo8qECo.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-13-222-21-231.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1015-aws x86_64)

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

 System information as of Tue Nov 11 05:50:06 UTC 2025

  System load:  0.08              Temperature:           -273.1 C
  Usage of /:   25.9% of 6.71GB   Processes:             118
  Memory usage: 24%               Users logged in:       0
  Swap usage:   0%                IPv4 address for ens5: 172.31.9.83

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

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


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-9-83:~$
```

Step 16: check the docker and git version

If they are not present, then go to administrative terminal using command

“sudo su”

Then update using the command “sudo apt-get update”

```
ubuntu@ip-172-31-9-83:~$ docker --version
Command 'docker' not found, but can be installed with:
sudo snap install docker          # version 28.4.0, or
sudo snap install docker          # version 28.1.1+1
sudo apt  install docker.io       # version 28.2.2-0ubuntu1~24.04.1
sudo apt  install podman-docker   # version 4.9.3+ds1-1ubuntu0.2
See 'snap info <snapname>' for additional versions.
ubuntu@ip-172-31-9-83:~$ git --version
git version 2.43.0
ubuntu@ip-172-31-9-83:~$ sudo su
root@ip-172-31-9-83:/home/ubuntu# sudo apt-get update
```

Step 17: use command “sudo apt-get install docker.io” to install docker

```
Get:50 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:51 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [27.4 kB]
Get:52 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [5708 B]
Get:53 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:54 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [384 B]
Fetched 38.6 MB in 6s (6197 kB/s)
Reading package lists... Done
root@ip-172-31-9-83:/home/ubuntu# sudo apt-get install docker.ic
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-buildx docker-compose-v2 docker-doc rinse
  zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz runc ubuntu-fan
0 upgraded, 8 newly installed, 0 to remove and 10 not upgraded.
Need to get 76.0 MB of archives.
After this operation, 288 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 bridge-utils amd64 1.7.1-1ubuntu2 [33.9 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.3.3-0ubuntu1~24.04.2 [8815 kB
]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.28-0ubuntu1~24.04.1 [
38.4 MB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 dns-root-data all 2024071801~ubuntu0.24.04
.1 [5918 B]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 dnsmasq-base amd64 2.90-2ubuntu0.1 [376 kB
]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 docker.io amd64 28.2.2-0ubuntu1~24.04.
1 [28.3 MB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 ubuntu-fan all 0.12.16+24.04.1 [34.2 k
B]
Fetched 76.0 MB in 1s (81.3 MB/s)
Preconfiguring packages ...
Selecting previously unselected package pigz.
(Reading database ... 71735 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.8-1_amd64.deb ...
Unpacking pigz (2.8-1) ...
Selecting previously unselected package bridge-utils.
Preparing to unpack .../1-bridge-utils_1.7.1-1ubuntu2_amd64.deb ...
Unpacking bridge-utils (1.7.1-1ubuntu2) ...
Selecting previously unselected package runc.
Preparing to unpack .../2-runc_1.3.3-0ubuntu1~24.04.2_amd64.deb ...
```

Step 18: Clone the git repo that has maven project and change to that directory

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-9-83:/home/ubuntu# git clone https://github.com/Gayathri2608-hub/maven-practice.git
Cloning into 'maven-practice'...
remote: Enumerating objects: 60, done.
remote: Counting objects: 100% (60/60), done.
remote: Compressing objects: 100% (40/40), done.
remote: Total 60 (delta 11), reused 34 (delta 2), pack-reused 0 (from 0)
Receiving objects: 100% (60/60), 13.39 KiB | 3.35 MiB/s, done.
Resolving deltas: 100% (11/11), done.
root@ip-172-31-9-83:/home/ubuntu# ls
maven-practice
root@ip-172-31-9-83:/home/ubuntu# cd maven-practice
root@ip-172-31-9-83:/home/ubuntu/maven-practice# ls
Dockerfile  demo  pom.xml  readme  src  target
root@ip-172-31-9-83:/home/ubuntu/maven-practice#
```

Step 19: change to the project directory and check for Dockerfile, if not present create the Dockerfile – ‘‘nano Dockerfile” and then build the image

“sudo docker build -t image_name .” name of image:img1

```
root@ip-172-31-9-83:/home/ubuntu/maven-practice# ls
Dockerfile  demo  pom.xml  readme  src  target
root@ip-172-31-9-83:/home/ubuntu/maven-practice# sudo docker build -t dep1 .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  120.8kB
Step 1/4 : FROM tomcat:9.0
9.0: Pulling from library/tomcat
4b3ffd8ccb52: Pulling fs layer
b48f960b380d: Pulling fs layer
58424d8c3e86: Pulling fs layer
4f4fb700ef54: Pulling fs layer
37b617836889: Pulling fs layer
891b6ad931b7: Pulling fs layer
ac0beccecf50: Pulling fs layer
4f4fb700ef54: Waiting
37b617836889: Waiting
891b6ad931b7: Waiting
ac0beccecf50: Waiting
b48f960b380d: Verifying Checksum
b48f960b380d: Download complete
4b3ffd8ccb52: Verifying Checksum
4b3ffd8ccb52: Download complete
4f4fb700ef54: Verifying Checksum
4f4fb700ef54: Download complete
37b617836889: Verifying Checksum
37b617836889: Download complete
891b6ad931b7: Verifying Checksum
891b6ad931b7: Download complete
ac0beccecf50: Verifying Checksum
ac0beccecf50: Download complete
58424d8c3e86: Verifying Checksum
58424d8c3e86: Download complete
```

Step 20: Run the image “sudo docker run -d -p 8081:8080 img1”

```
root@ip-172-31-9-83:/home/ubuntu/ar/maven-practice# sudo docker run -d -p 8081:8080 img1
c5fd91cf9a9b4f0625d4d2c0d896406e8da76929ed75a3f9ccc1699fbbb08535
root@ip-172-31-9-83:/home/ubuntu/ar/maven-practice#
```

Step 21: Chcek the images and the containers

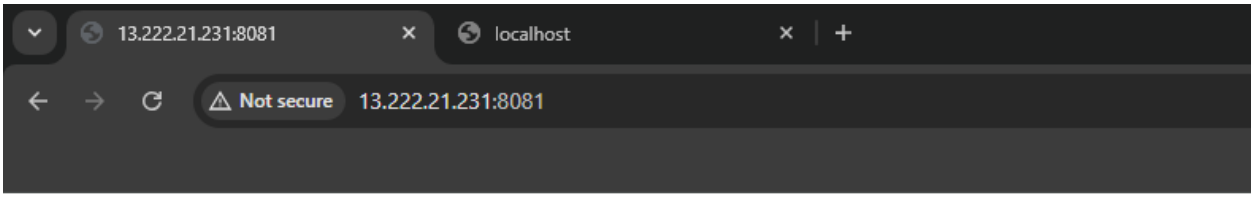
```
root@ip-172-31-9-83:/home/ubuntu/ar/maven-practice# sudo docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
img1         latest    a67a112ce8ac   2 minutes ago    413MB
dep1         latest    28efbe56e633   29 minutes ago   413MB
tomcat       9.0       2e4887a16e43   12 hours ago     413MB
root@ip-172-31-9-83:/home/ubuntu/ar/maven-practice# docker ps
CONTAINER ID   IMAGE     COMMAND             CREATED              STATUS              PORTS
          NAMES
c5fd91cf9a9b   img1      "catalina.sh run"   About a minute ago   Up About a minute   0.0.0.0:8081->8080/tcp, [::]:8081-
>8080/tcp   charming_banzai
84e7f9ce5ec2   dep1      "catalina.sh run"   9 minutes ago        Up 9 minutes        0.0.0.0:8080->8080/tcp, [::]:8080-
>8080/tcp   angry_shaw
b62aedc8bb3b   dep1      "catalina.sh run"   27 minutes ago       Up 27 minutes       0.0.0.0:7070->8080/tcp, [::]:7070-
>8080/tcp   sweet_archimedes
root@ip-172-31-9-83:/home/ubuntu/ar/maven-practice#
```

Step 22: Take the public IP address from the instances in AWS and open it in chrome along with the port number mapped.

Public IP- 13.222.21.231

Port used: 8081

Use: 13.222.21.231:8081, you will find your application that is deployed



Hello World! practice