KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY
(AN AUTONOMOUS INSTITUTION)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH,
Narayanguda, Hyderabad, Telangana – 500029

DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)

LAB RECORD

SOFTWARE ENGINEERING LAB

B. Tech. III YEAR I SEM (KR23)

ACADEMIC YEAR
2025-26

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY
(AN AUTONOMOUS INSTITUTION)

Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH, Hyderabad
Narayanguda, Hyderabad, Telangana – 500029

Certificate

This is to certify that following is a Bonafide Record of the workbook task done by

_____bearing Roll No_____ of _____

Branch of _____ year B. Tech. Course in the_____

Subject during the Academic year _____ & _____ under our supervision.

Number of week tasks completed: _____

Signature of Staff Member Incharge

Signature of Head of the Dept.

Signature of Internal Examiner

Signature of External Examiner

KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY
(AN AUTONOMOUS INSTITUTION)
Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH, Hyderabad
Narayanguda, Hyderabad, Telangana – 500029

Daily Laboratory Assessment Sheet

Name of the Lab: Student Name:
Branch & Section: HT. No:

Sr. No.	Name of the Experiment	Date	Observation Marks (5M)	Record Marks (5M)	Viva Voice Marks(10M)	Total Marks (20M)	Signature of Faculty
	Total						

Faculty Incharge

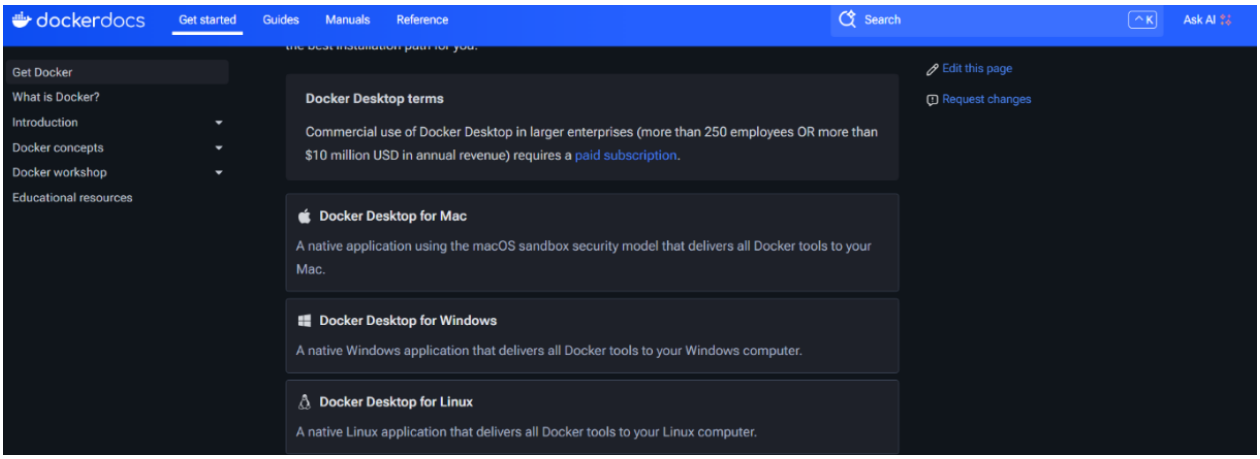
INDEX

	b. Gain skills in writing and interpreting docker-compose.yml files for service setup. c. Deploy the same setup across different machines without manual configuration. d. Configure container networking and persistent storage within Compose. e. Reduce setup time and enable faster iteration during application development.	
7.	**Creating a Multi-Module Maven Project** a. Build and package Java and Web applications using Maven. b. Add dependencies using **pom.xml**, compile and test using plugins. c. Resolve errors and conflicts arising from dependency mismatches. d. Work with parent and multi-module Maven projects. e. Generate executable JARs and deployable WARs using Maven	
8.	**Jenkins Automation** a. Hands-on practice on manual creation of Jenkins pipeline using Maven projects from Github b. Create the job and build the pipeline for maven-java and maven-web project.	
9.	**Pipeline Creation using script** a. Evaluation of Jenkins pipeline. b. WORKING ON BUILD TRIGGERS FOR LAST JENKINS PIPILINE c. Hands-on practice on creation of scripted Jenkins pipeline. d. Take the screenshots for above task	
10.	**Working with minikube and Nagios** a. Hands-on practice of creating, running and scaling pods in minikube. b. Running Nginx server on specified port number by explaining the Nginx monitoring tool c. Running Nagios server and Understanding the Monitoring tool using Docker. d. AWS-free Trier account Creation steps	
11.	**Jenkins-CI/CD** a. CI-Continuous Integration using Webhooks. b. Sending E-mail Notification on Build Failure or success	
12.	**Creation of virtual machine for Ubuntu OS and Deploying the web application** 1. Creation of virtual machine 2. Deploying the web application 3. Accessing it publicly	

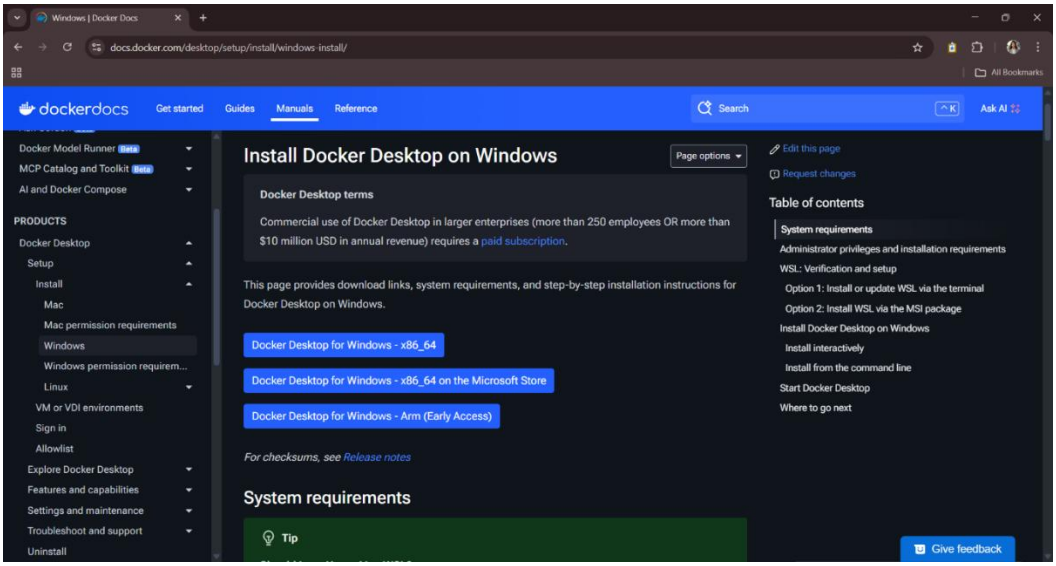
1.Software Installation & SRS Document:

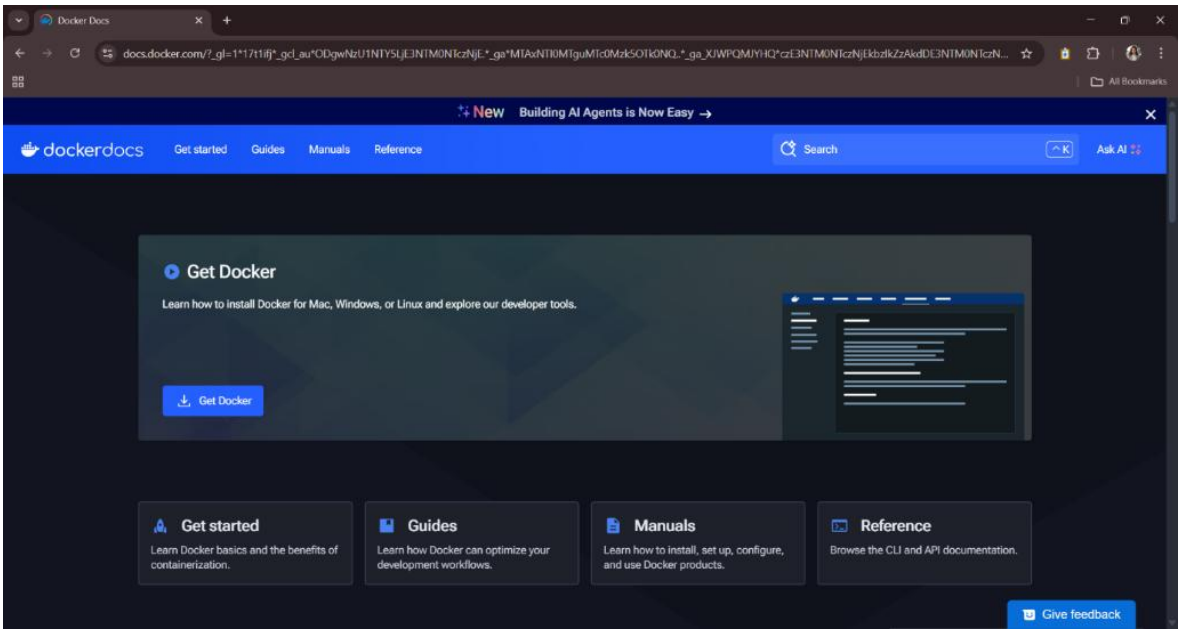
DOCKER- INSTALLATION

Step-1: Go to docker website

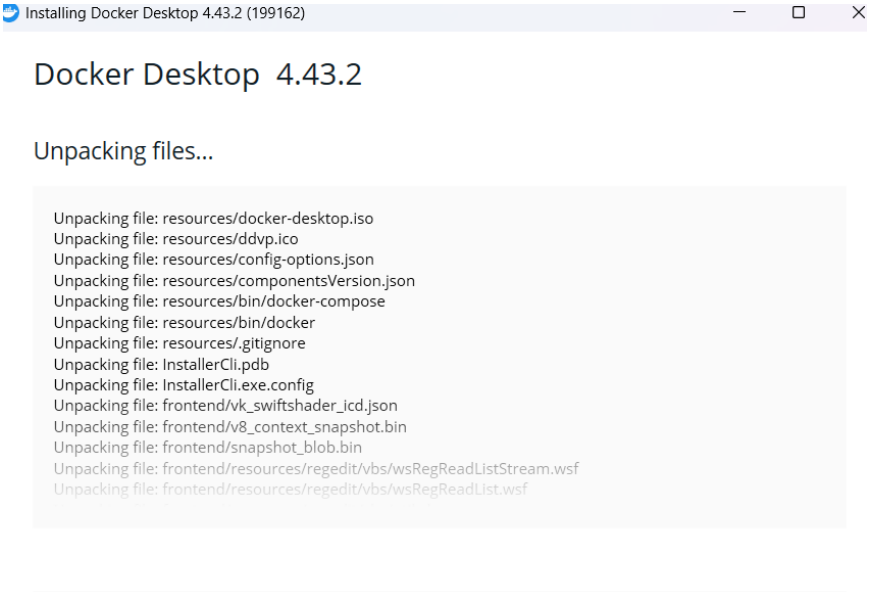


Step-2:Select the suitable one for your system

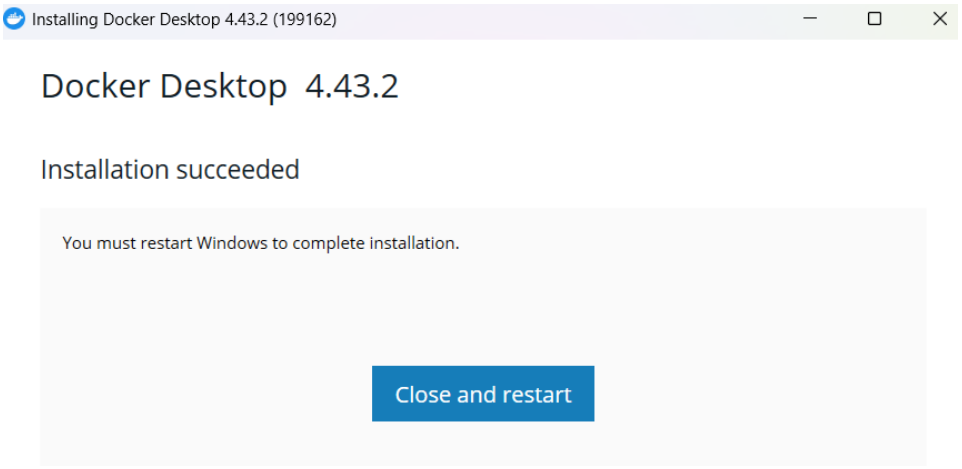




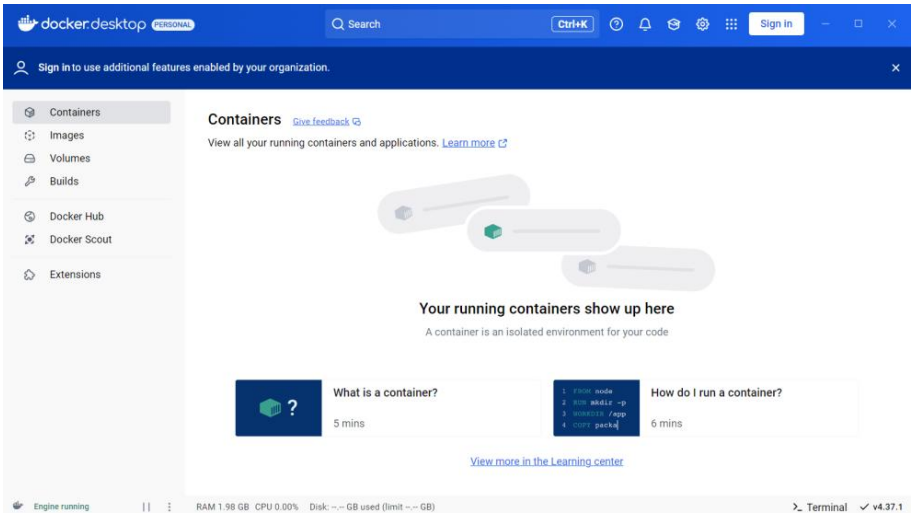
Step-3: After clicking on get docker it starts initializing



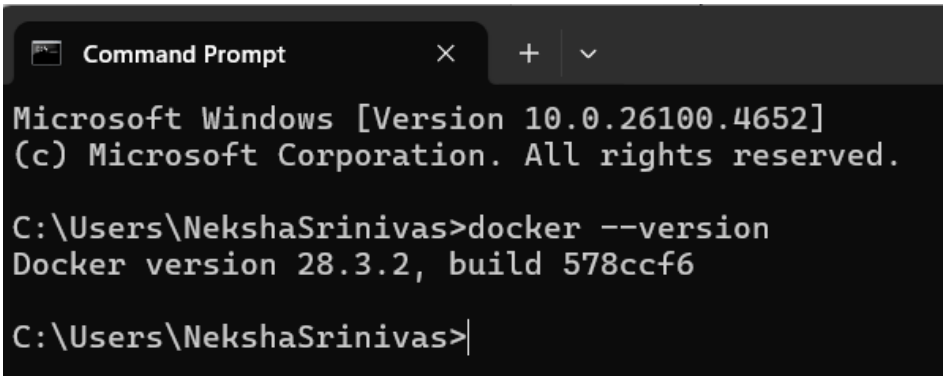
Step-4:Installation successful



Step-5: Docker interface



Step-6: docker version

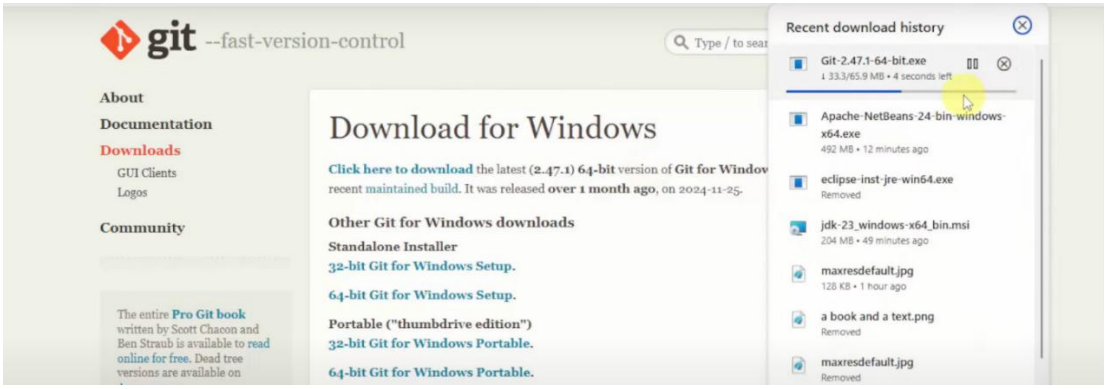


GIT – INSTALLATION:

Step-1: Go to Git website

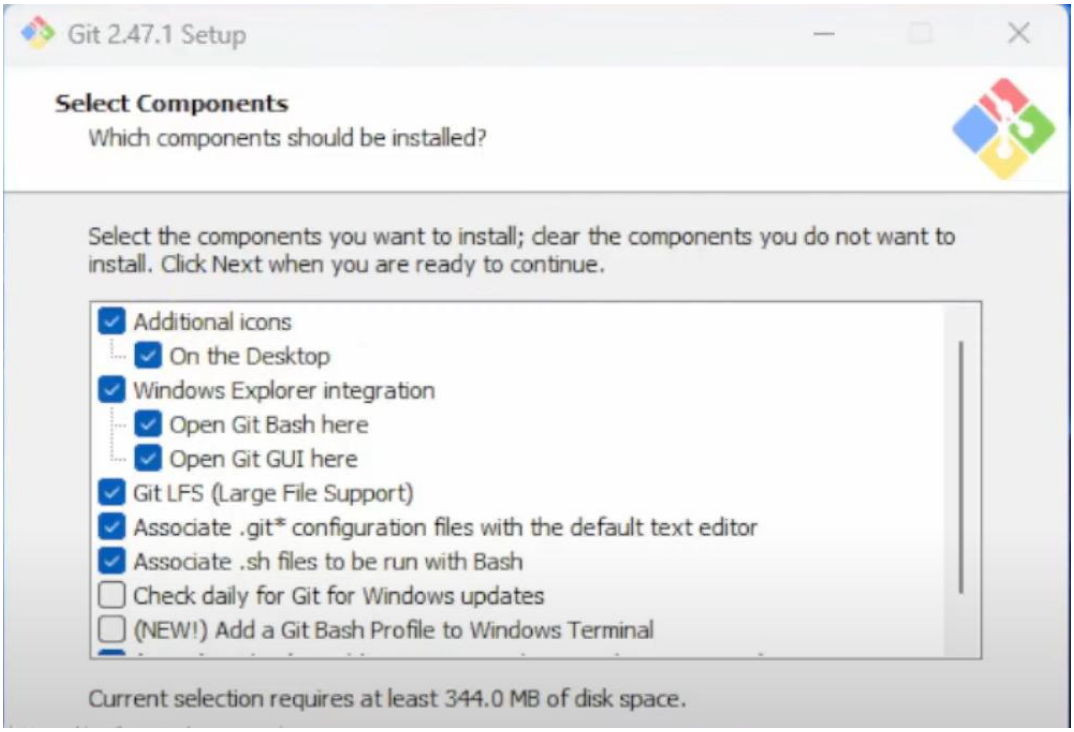
Step-2:click on downloads and options will be displayed

Step-3: Download for windows(suitable one for your system)



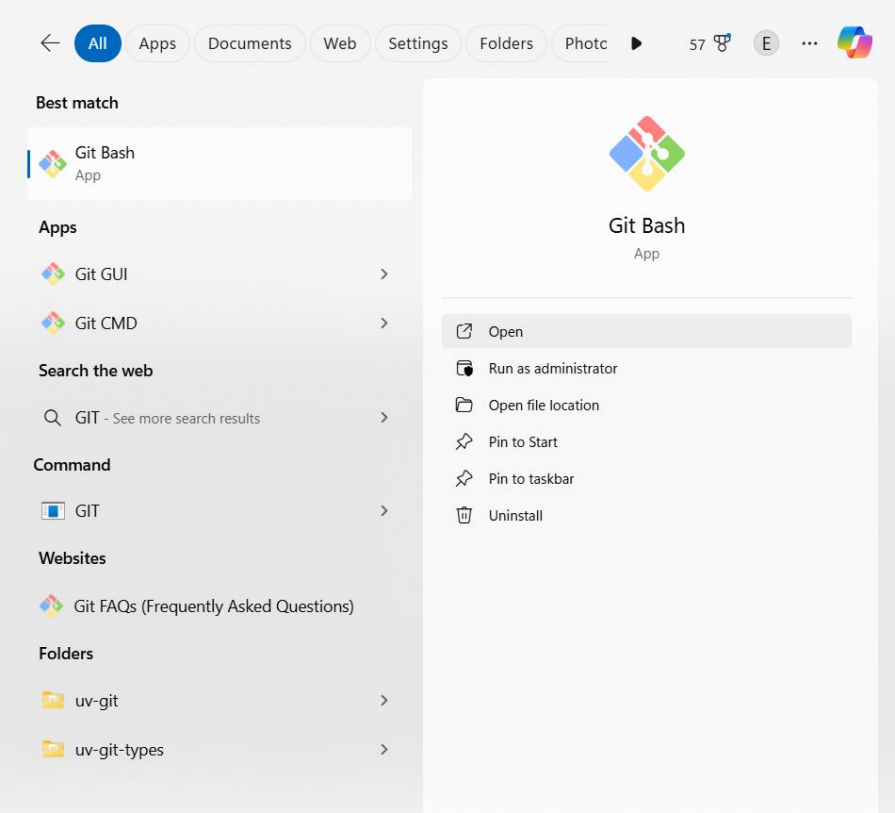
Step-4: License will be displayed click on next

Step-5: Select the componets and click next

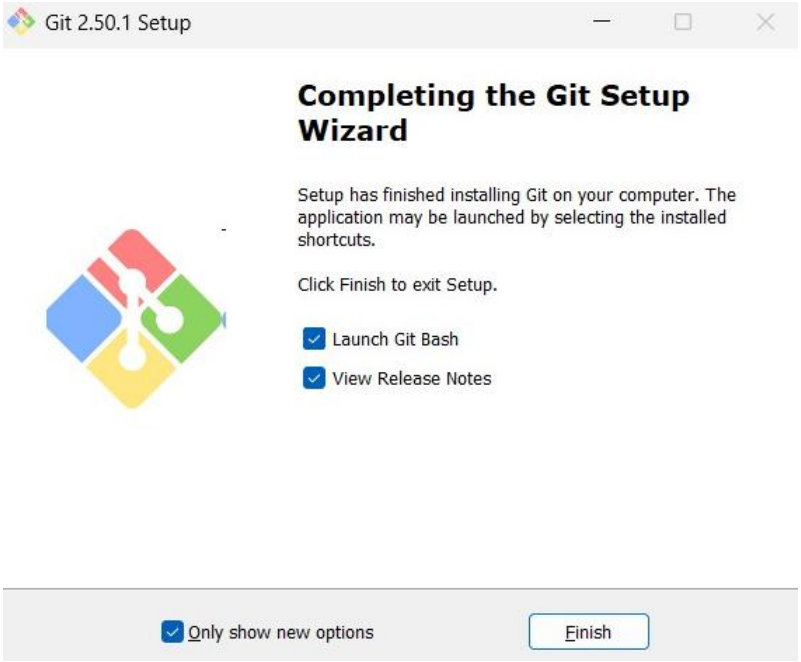


Git bash:

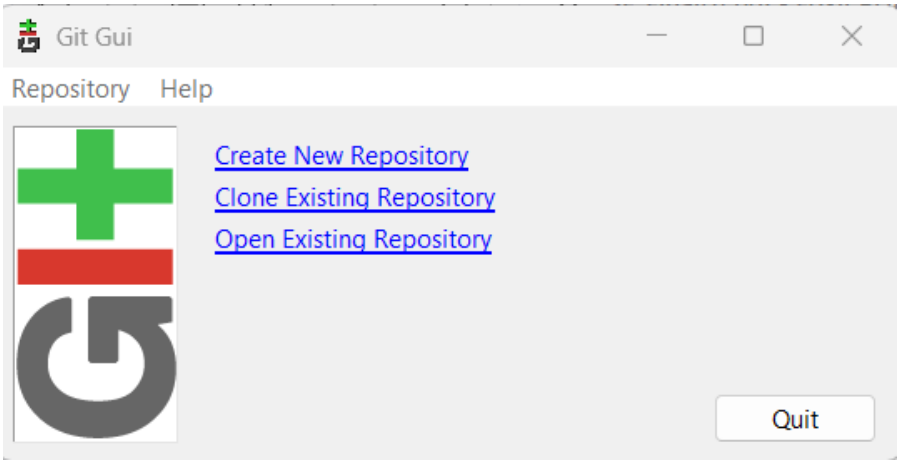
Step-1: Go to search bar and click git bash



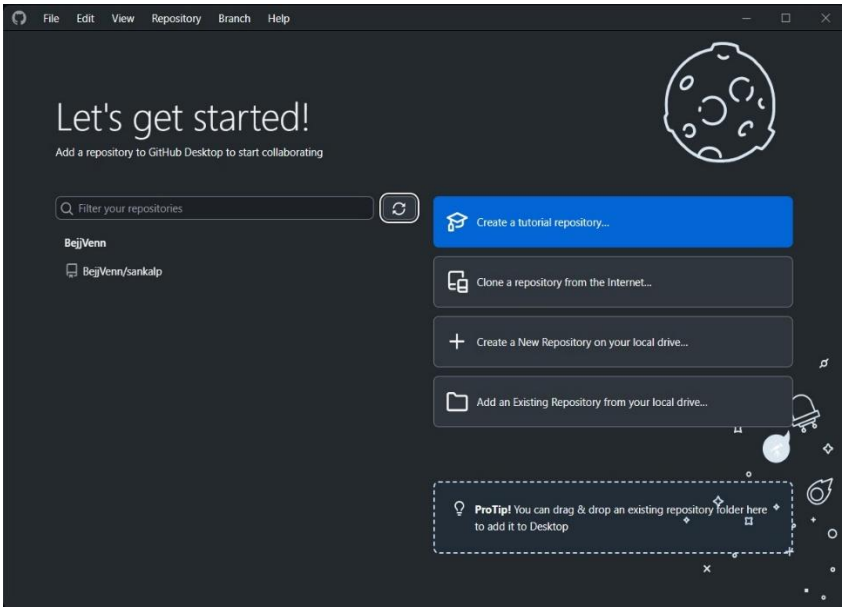
Step-2: Click on finish



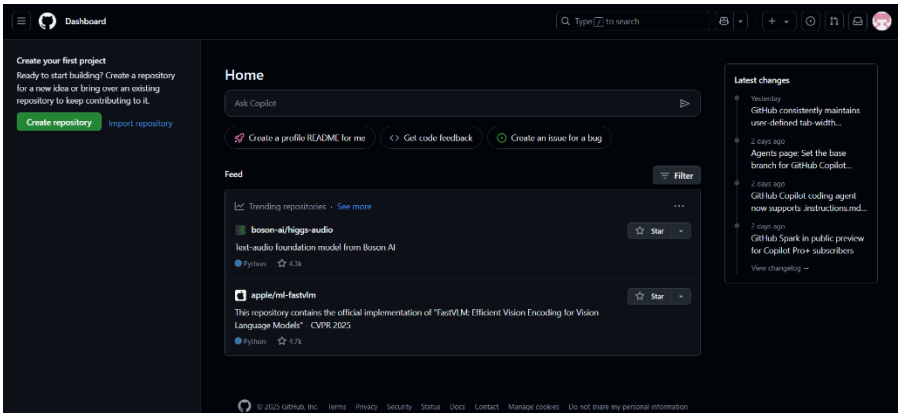
Step-3: git bash interface



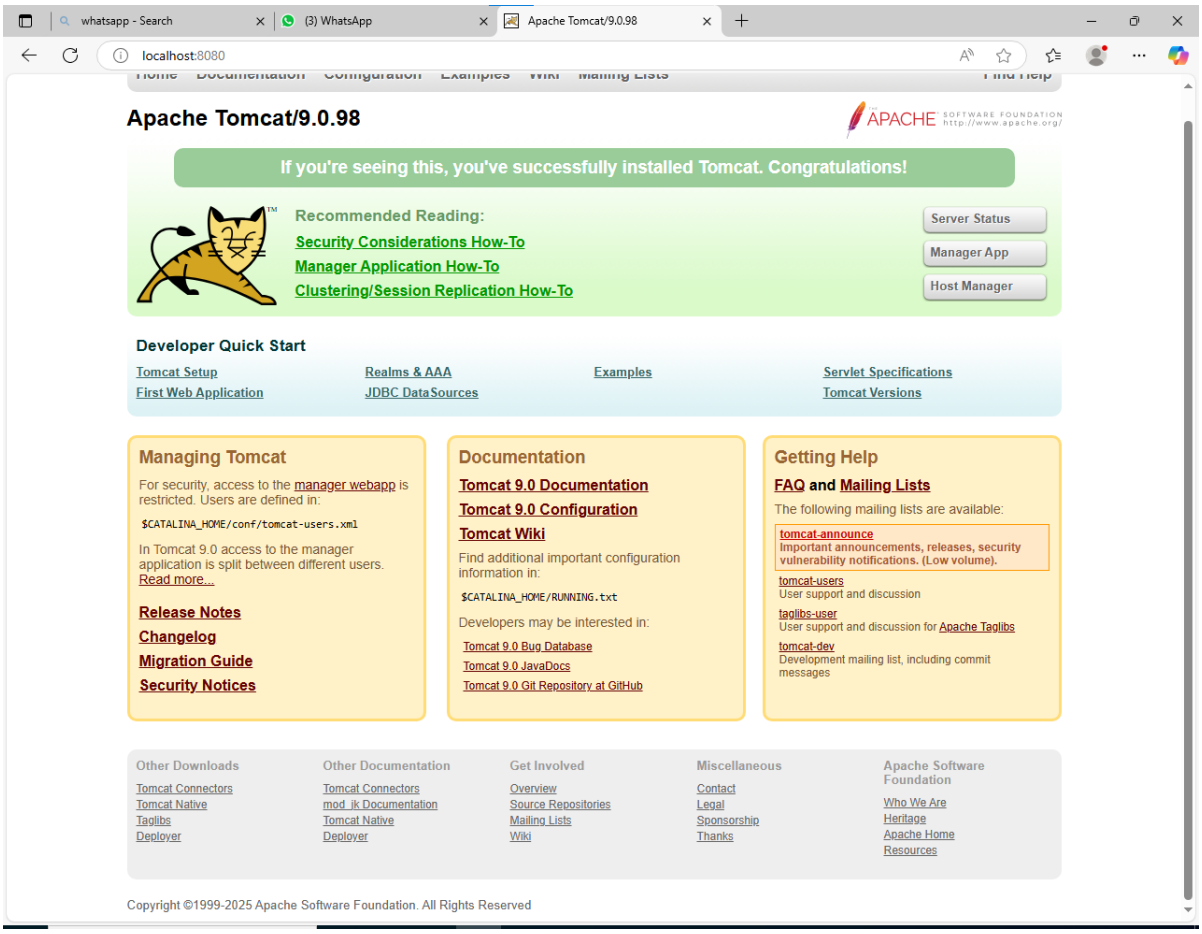
GIT-ACCOUNT



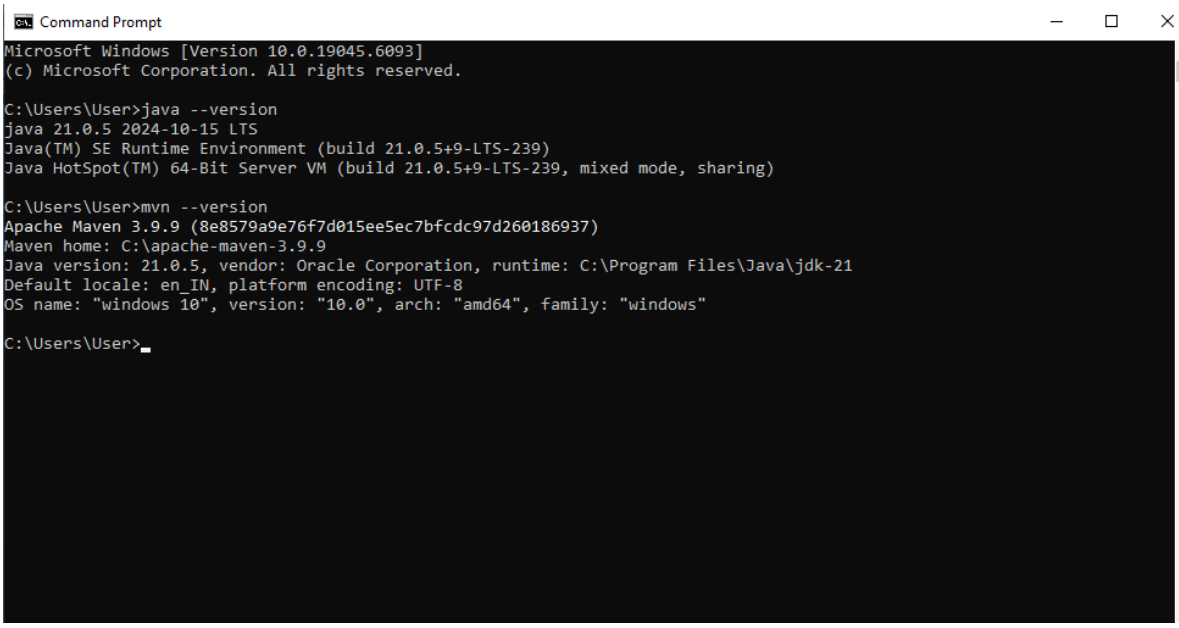
GIT-ACCOUNT



Tomcat

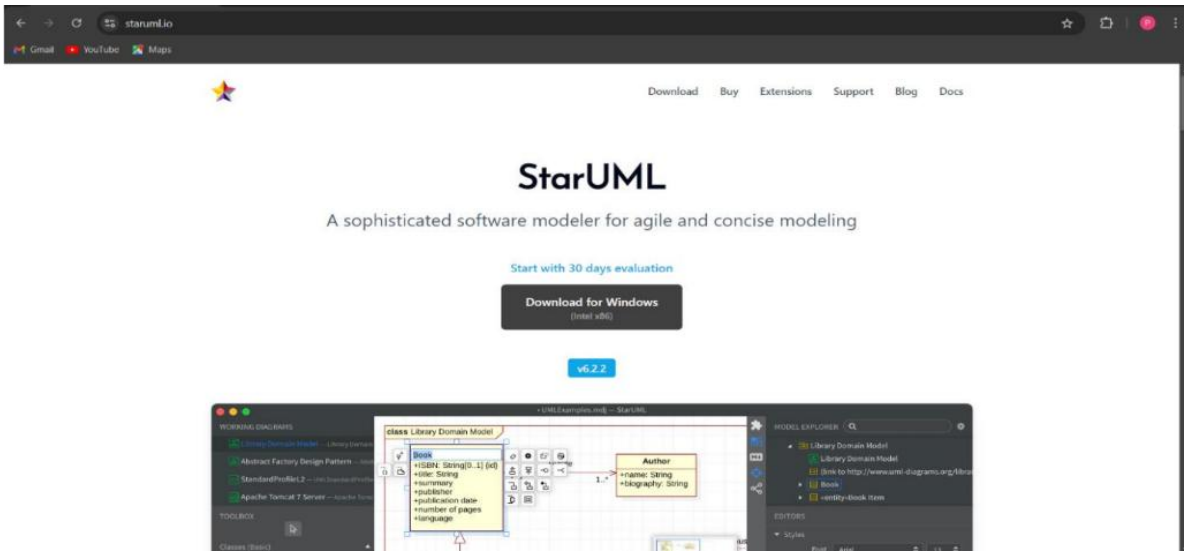


Java and maven versions

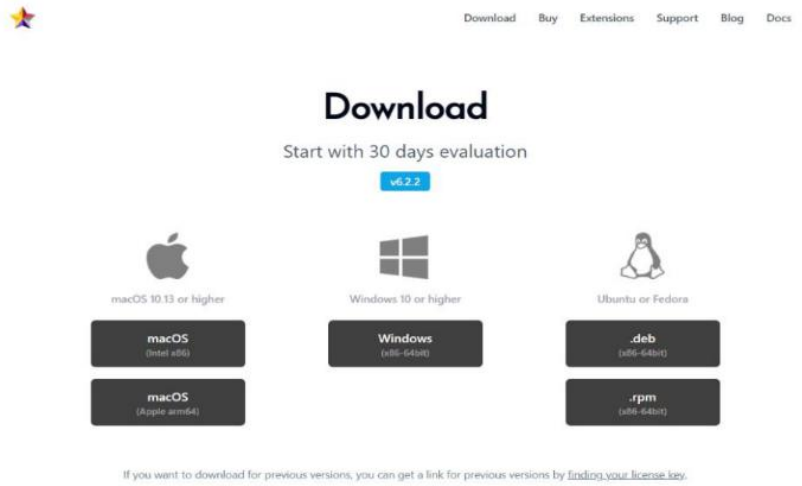


StarUML INSTALLATION

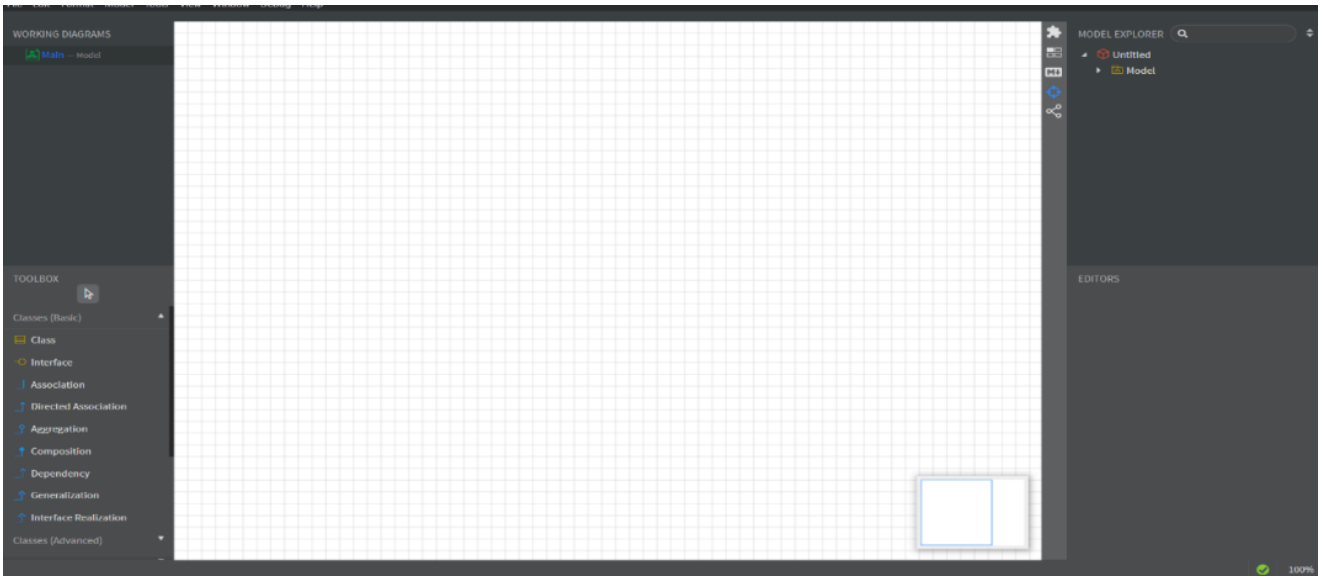
Step-1: Go to startuml website



Step-2: from the given options select the suitable one for your system



Step-3: Interface

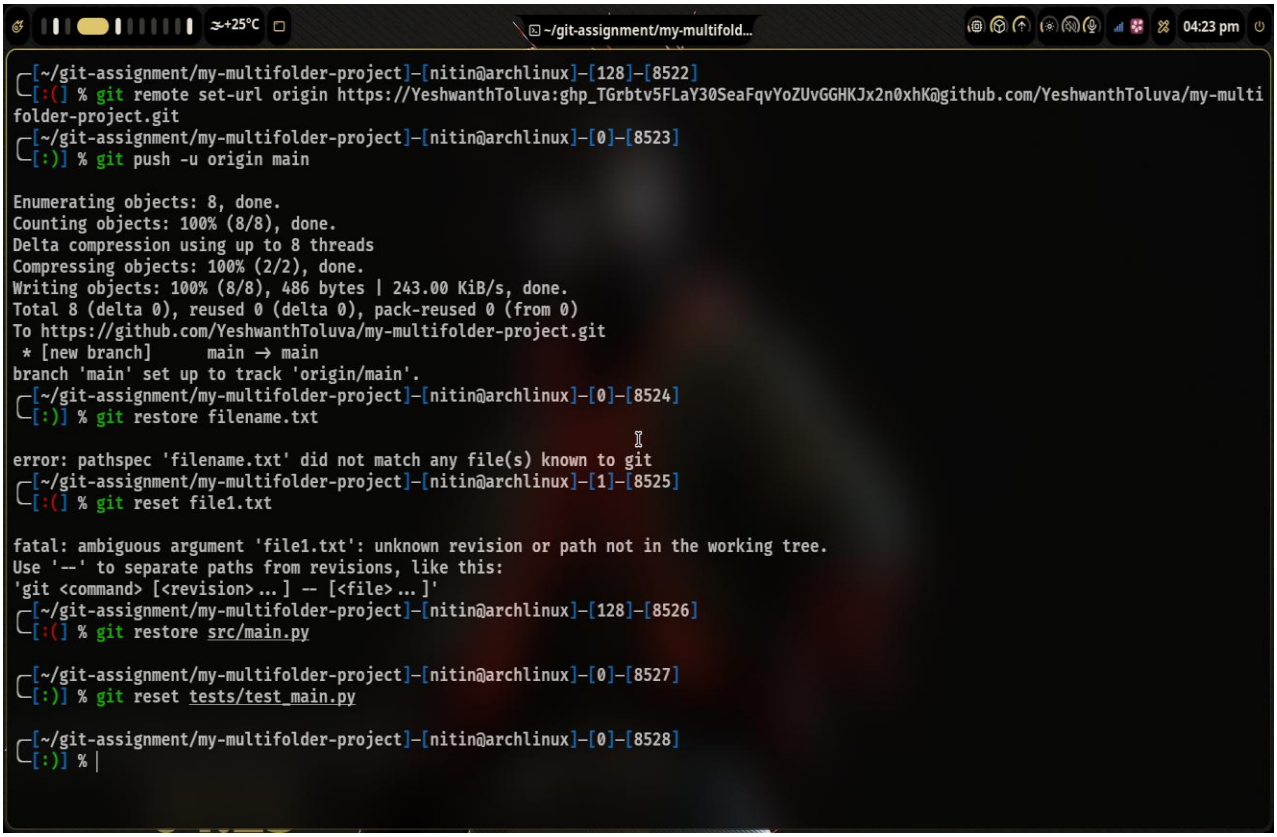


2. Exploring git local and remote commands on the multi-folder project

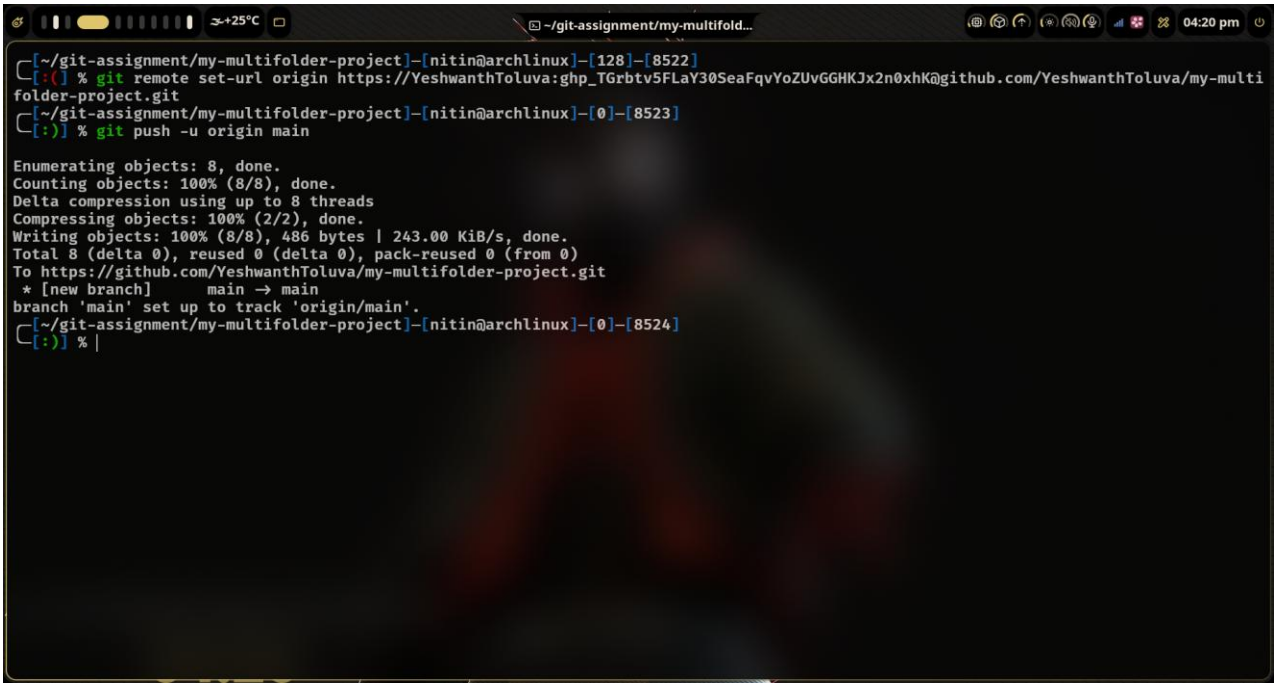
Github Global Configuration:

```
C:\Users\NekshaSrinivas>git config --global --list
core.editor="C:\Users\NekshaSrinivas\AppData\Local\Programs\Microsoft VS Code\bin\code" --wait
user.name=Edigirala-Neksha
user.email=edigiralaneksha@gmail.com

C:\Users\NekshaSrinivas>
```

Git Push to GitHub Public Repository with Remote Set



Scenario-Based Git Commands: Discarding and Unstaging Changes



1. You've cloned a repository and made some changes to a local branch. Now you want to push these changes to the remote repository, but you're getting an error saying "rejected - non-fast-forward." How would you resolve this?

This error occurs when the remote branch has changes that your local branch doesn’t. To resolve it:

git pull --rebase origin <branch-name>

This rebases your local changes on top of the latest remote changes. After resolving any conflicts, push your changes:

2. You’ve been working on a feature branch, and now you need to push it to the remote repository. However, the remote repository already has a main branch. How do you push your feature branch without affecting the main branch?

You can push your feature branch independently:

git push origin feature/feat-1

This creates a new remote branch and does not affect the main branch.

3. You cloned a remote repository, but after a while, the repository's structure changed and new branches were added. How would you keep your local repository updated with the latest changes from the remote repository?

Use the following commands:

git fetch origin

This updates your local copy with all branches and changes from the remote. You can then check out new branches using:

git checkout branch-name

4. A colleague has pushed some changes to the main branch, but you have local changes in the same branch. You want to pull their changes, but you want to avoid merge conflicts. What steps would you take?

Use rebase to integrate their changes on top of your work:

git stash # Temporarily store your changes

git pull --rebase origin main

git stash pop # Apply your changes on top

This reduces the chance of conflicts and keeps history clean.

5. You accidentally pushed a sensitive file (e.g., API keys) to the remote repository. How would you fix this situation?

Steps to remove the sensitive data:

Remove the file and commit:

git rm --cached path/to/file

git commit -m "Remove sensitive file"

git push origin main

If the secret is in history, use git filter-branch or BFG Repo-Cleaner to rewrite history:

**git filter-branch --force --index-filter **

**"git rm --cached --ignore-unmatch path/to/file" **

--prune-empty --tag-name-filter cat -- --all

Force push and rotate the secret.

6. You're working on a feature branch, and your manager requests that you integrate the latest changes from main into your feature branch. What steps would you take?

Use rebase or merge:

Rebase:

git checkout feature/your-feature

git fetch origin

git rebase origin/main

7. You cloned a remote repository, but later you find that you need to push your changes to a different remote repository. How do you configure your local repository to push to this new remote?

Then push your changes:

git push origin branch-name

8. After running git pull, you notice that your local branch is behind the remote branch. How would you proceed to bring your local branch up to date without losing your local changes?

Use stash or rebase:

git stash

git pull --rebase origin branch-name

git stash pop

This ensures a clean rebase and retains your changes.

9. You're working on a project with multiple collaborators, and you notice that your local changes conflict with changes that have been pushed by others. How would you resolve the conflicts?

Pull the latest changes:

git pull origin branch-name

Git will highlight conflicts. Open the files, manually resolve the <<<<<<<, =======, and >>>>>>> markers.

Mark as resolved and commit:

git add .

git commit

10. You've pushed a feature branch to a remote repository, but now you need to delete the branch from the remote. How would you do that?

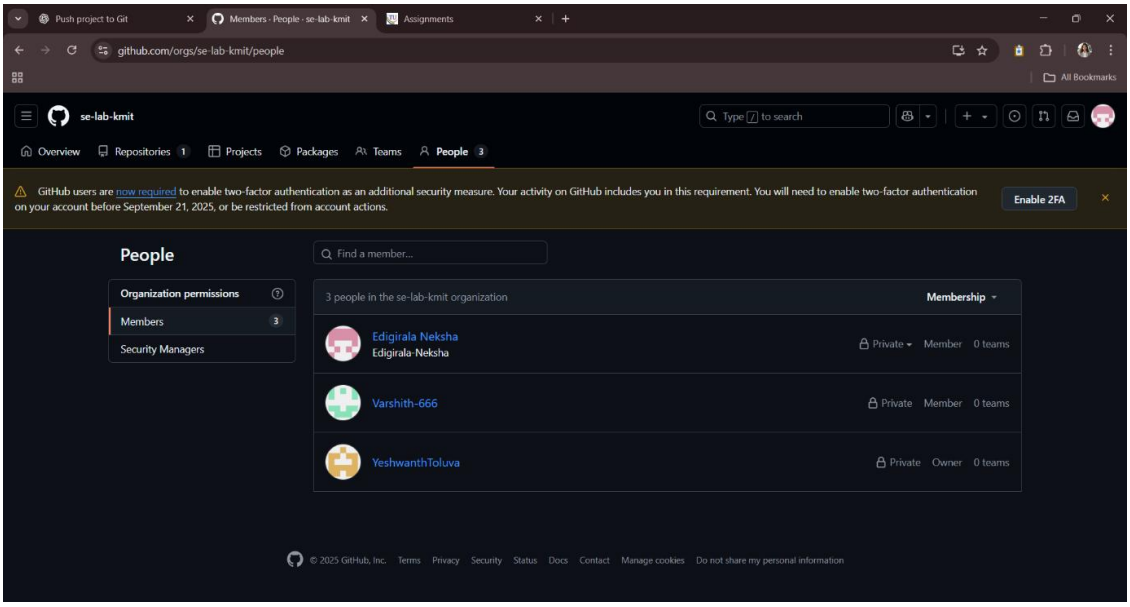
Use the following command:

git push origin --delete feature/branch-name

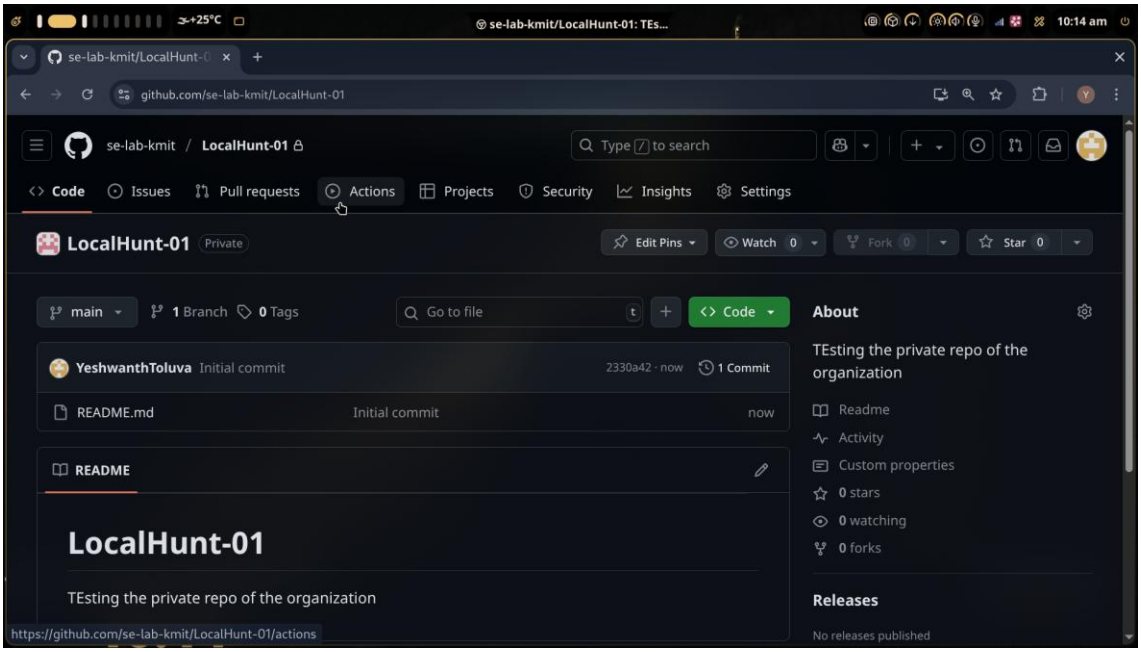
This will remove the branch from the remote repository.

3. Collaborative coding using git

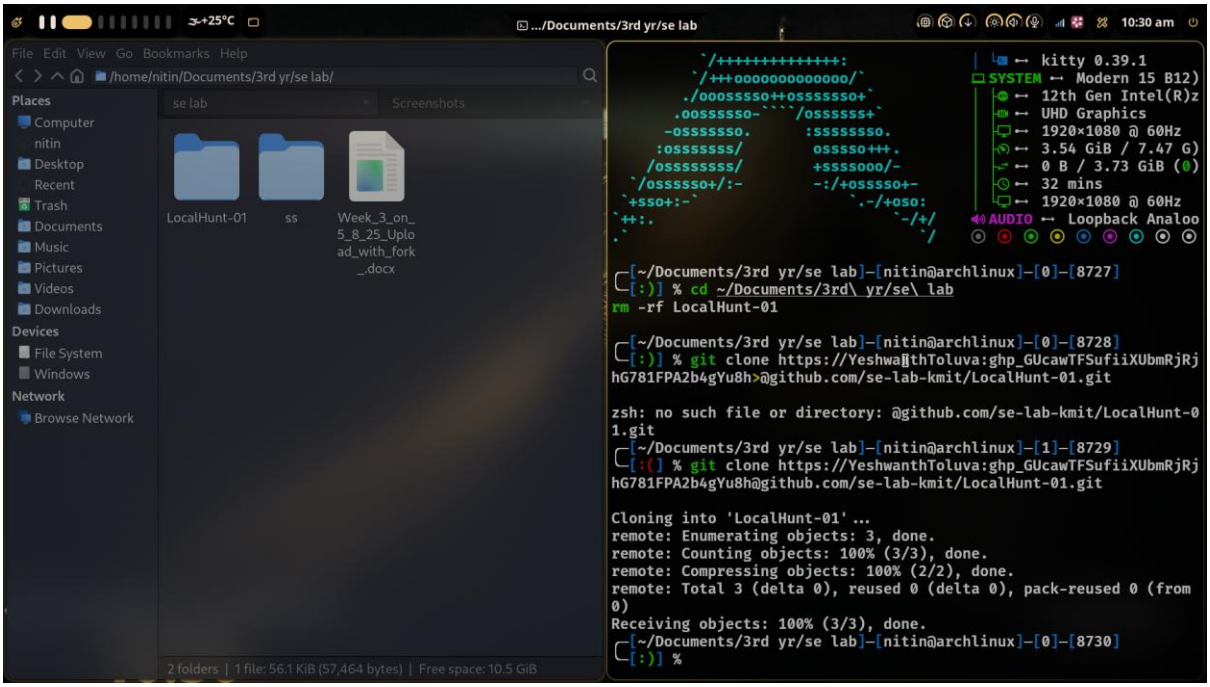
GitHub Organization Members Page - se-lab-kmit Team Overview



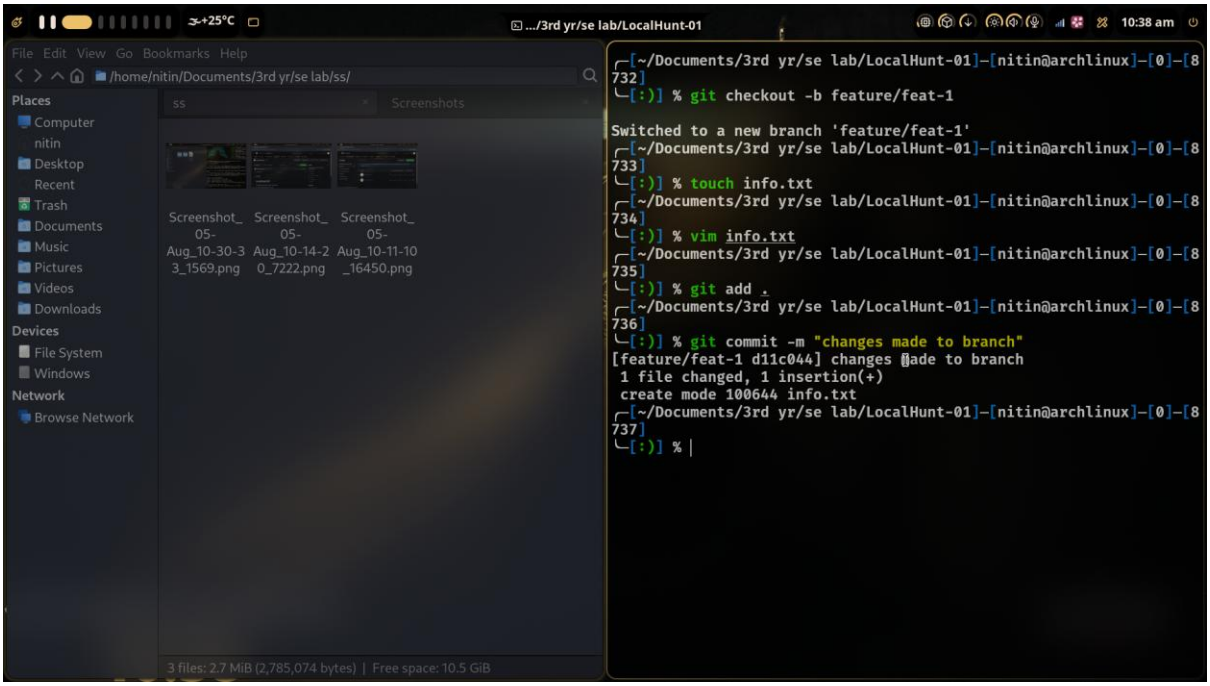
GitHub Repository Overview - LocalHunt-01 Private Repository



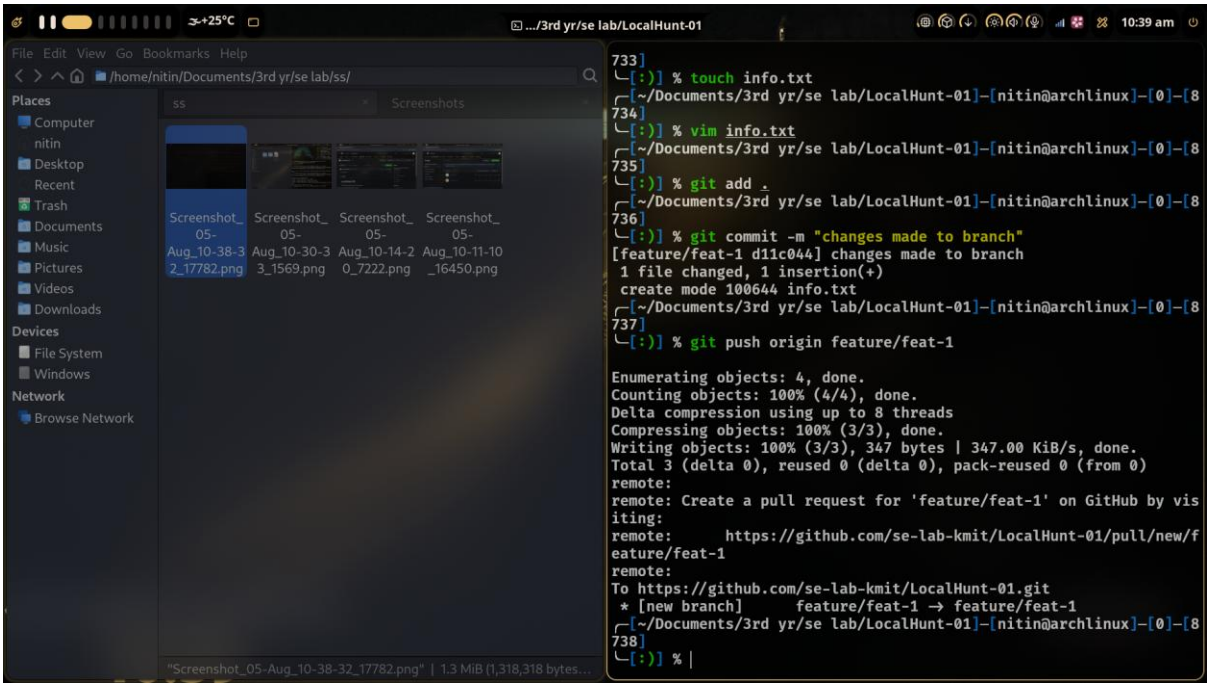
Terminal Git Clone Operations - LocalHunt-01 Repository Setup



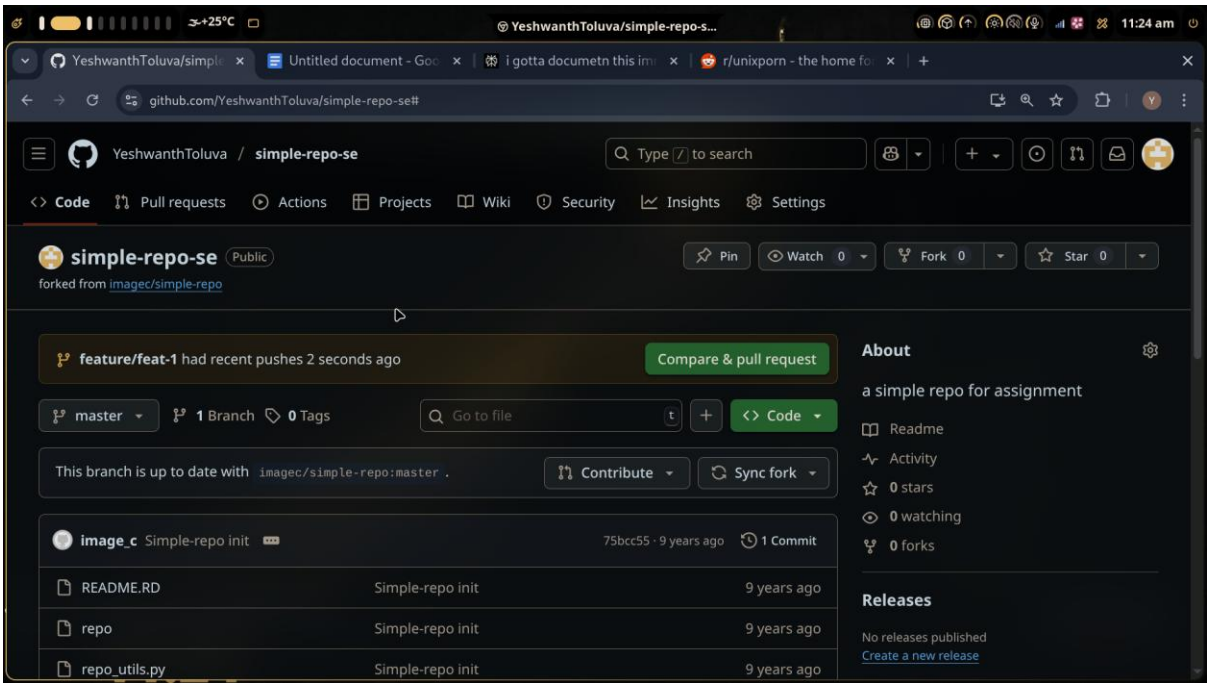
Git Branch Operations - Feature Branch Creation and File Management



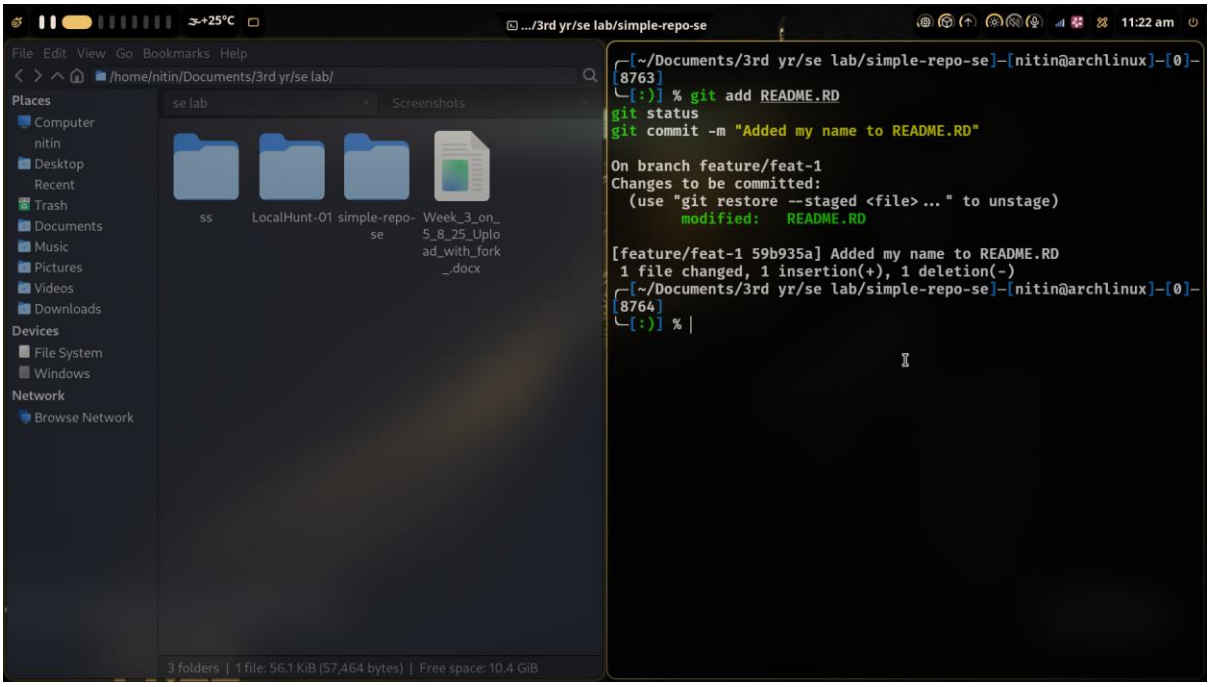
Git Push and Pull Request Creation - Feature Branch Workflow



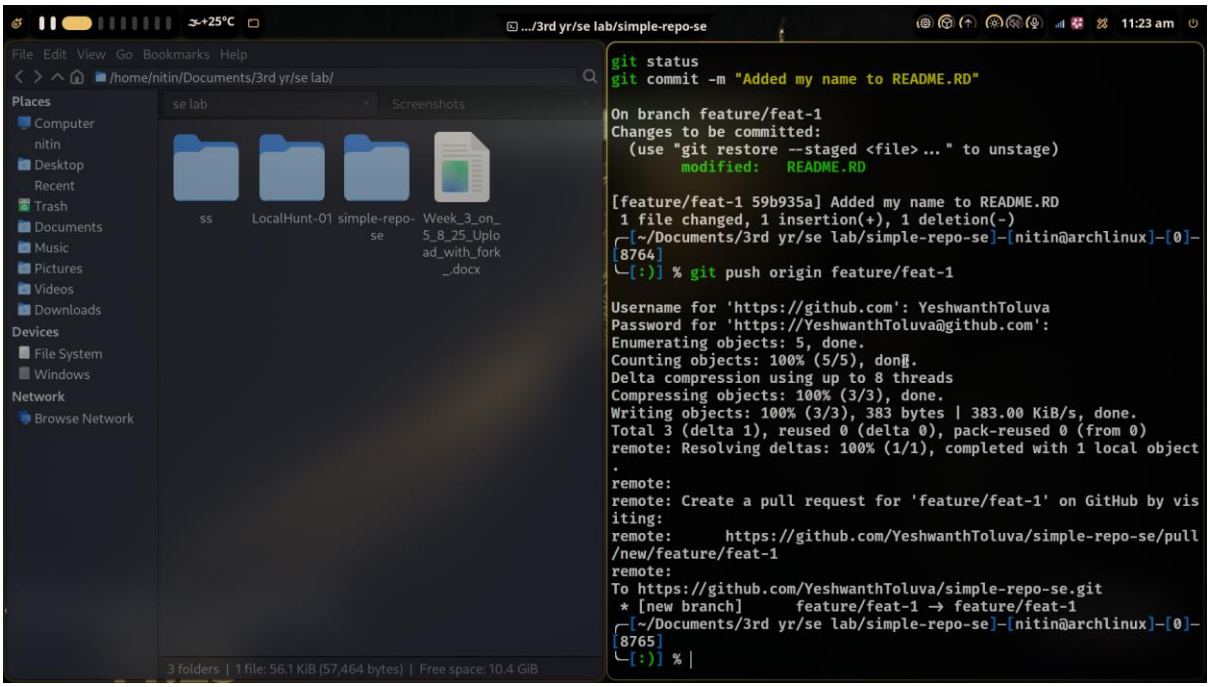
GitHub Repository Fork - simple-repo-se Overview and Setup



Git Commit and Status - README.RD File Modifications in Feature Branch

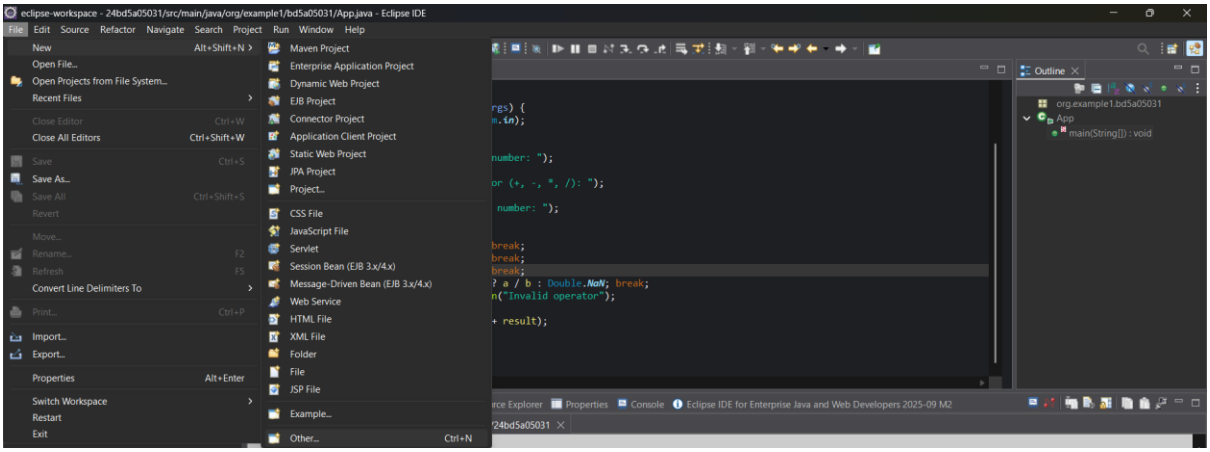


Git Push to Forked Repository - Feature Branch Upload and Pull Request Creation

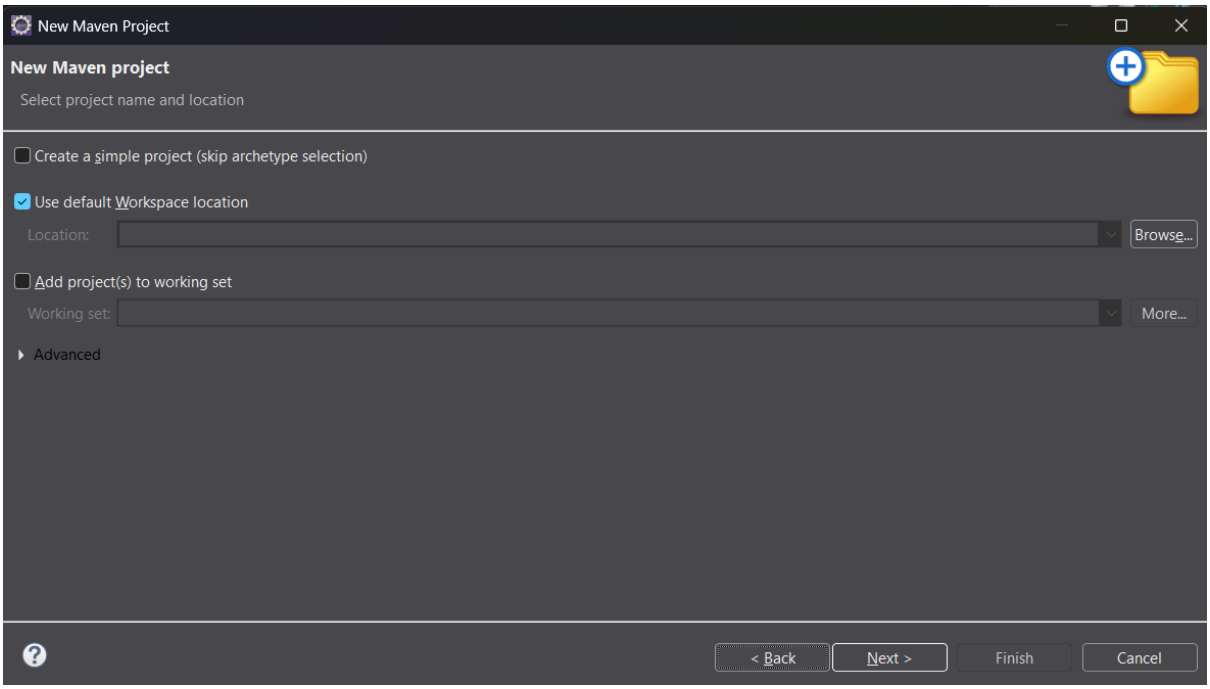


4. Build and package Java and Web applications using Maven

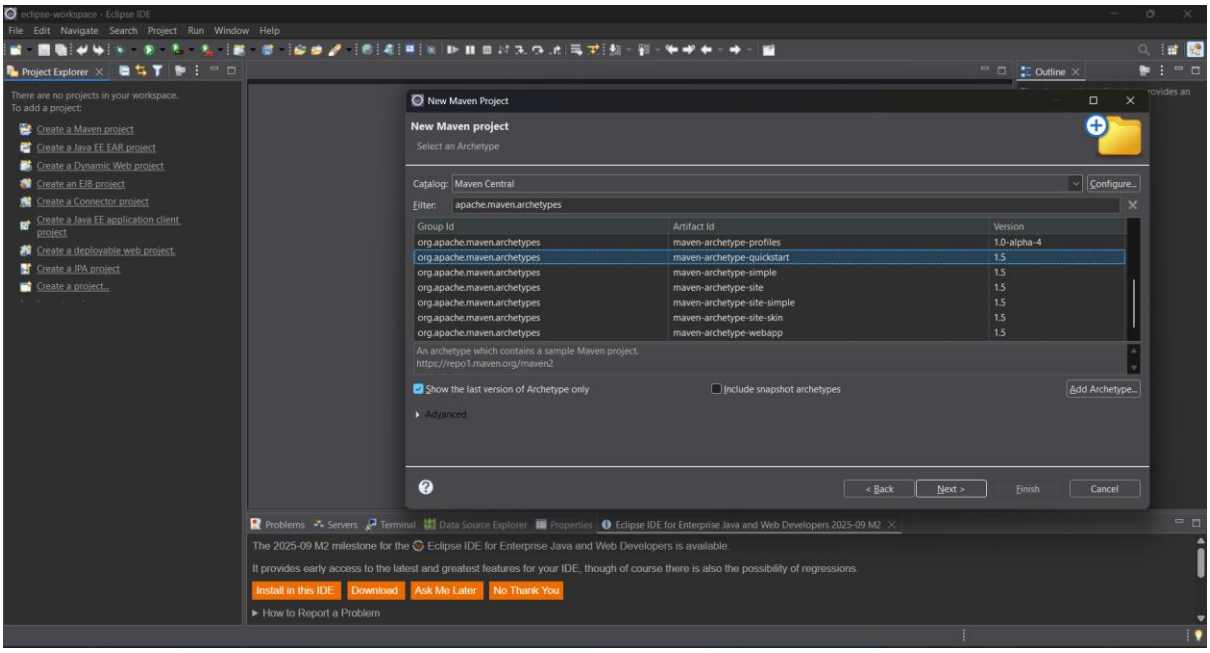
Step-1: Open the eclipse and click on file>new>Maven project



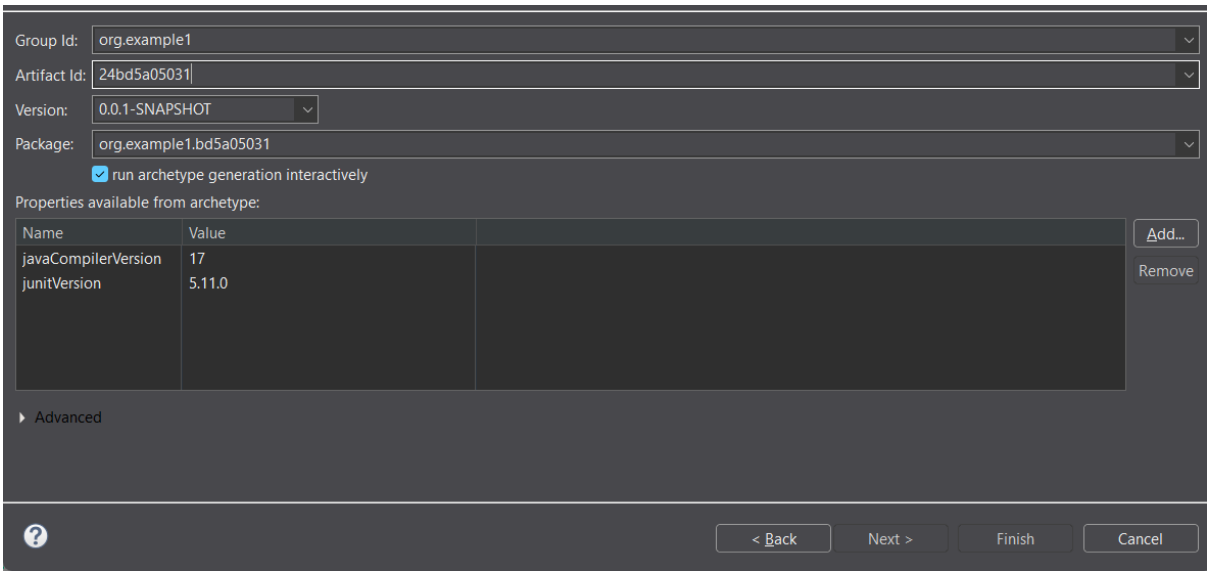
Step-2: select the default workspace and click on next



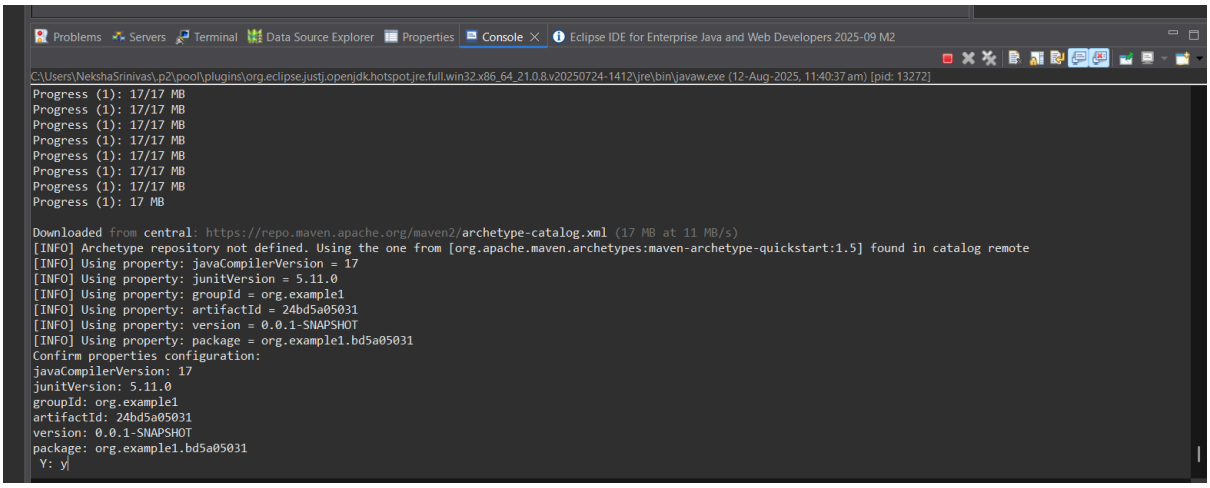
Step-3: in the filter option select the one maven-archetype-quickstart



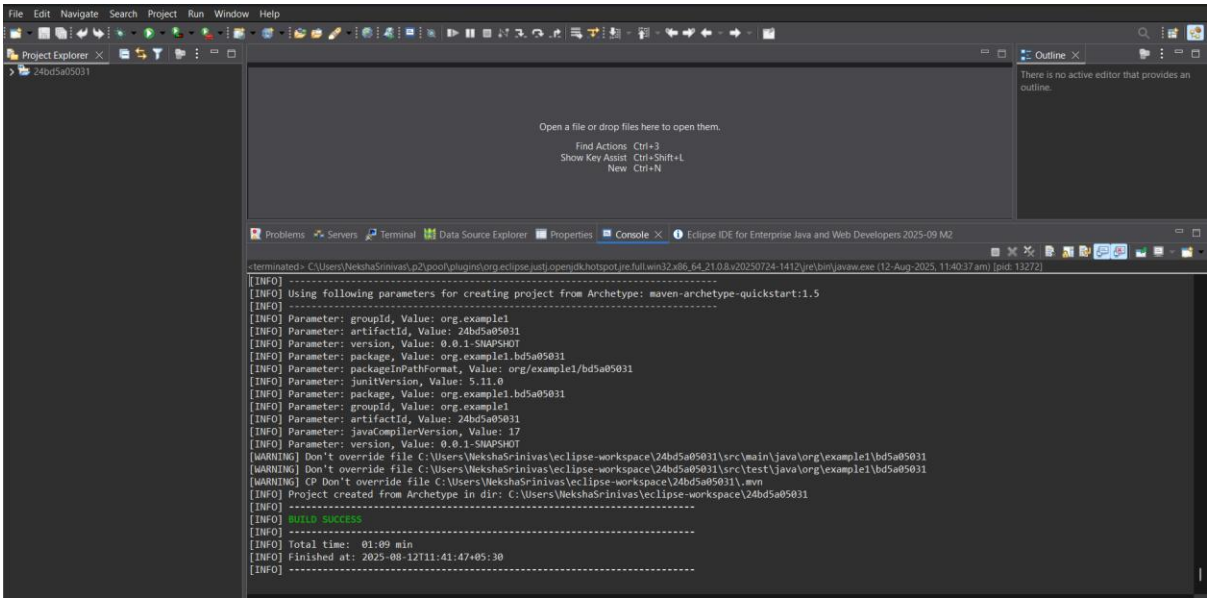
Step-4: give the Group Id and Artifact Id and click on next



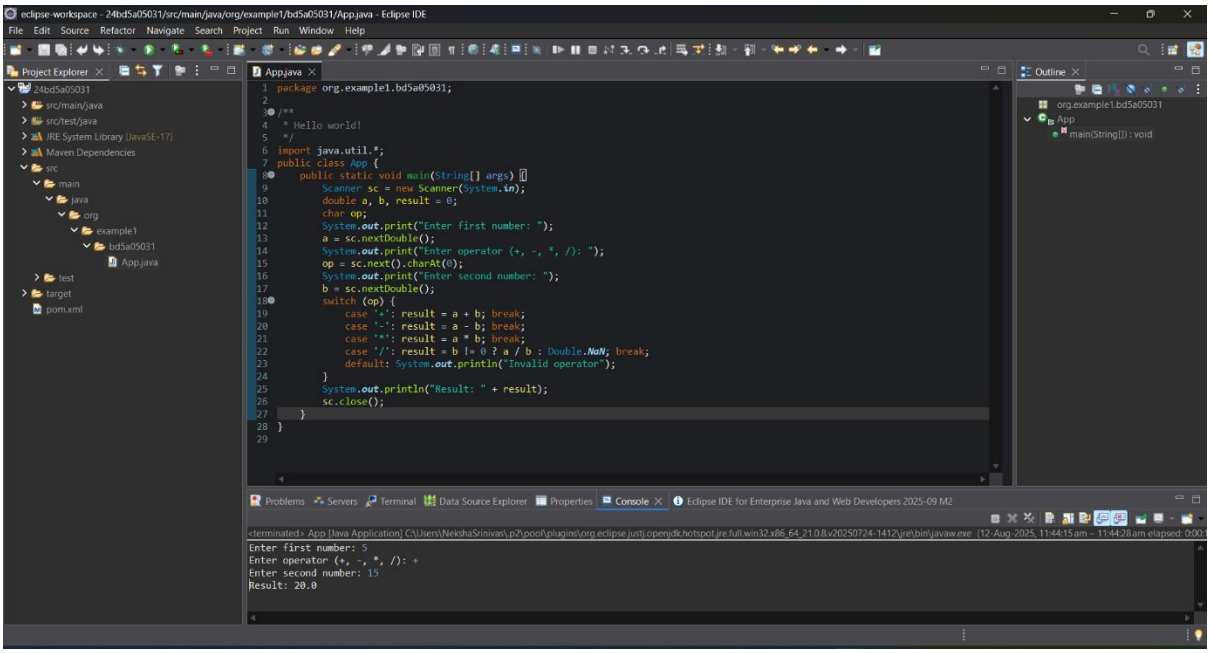
Step-5: In the console the progress will be showed type y (refers to yes) and press enter



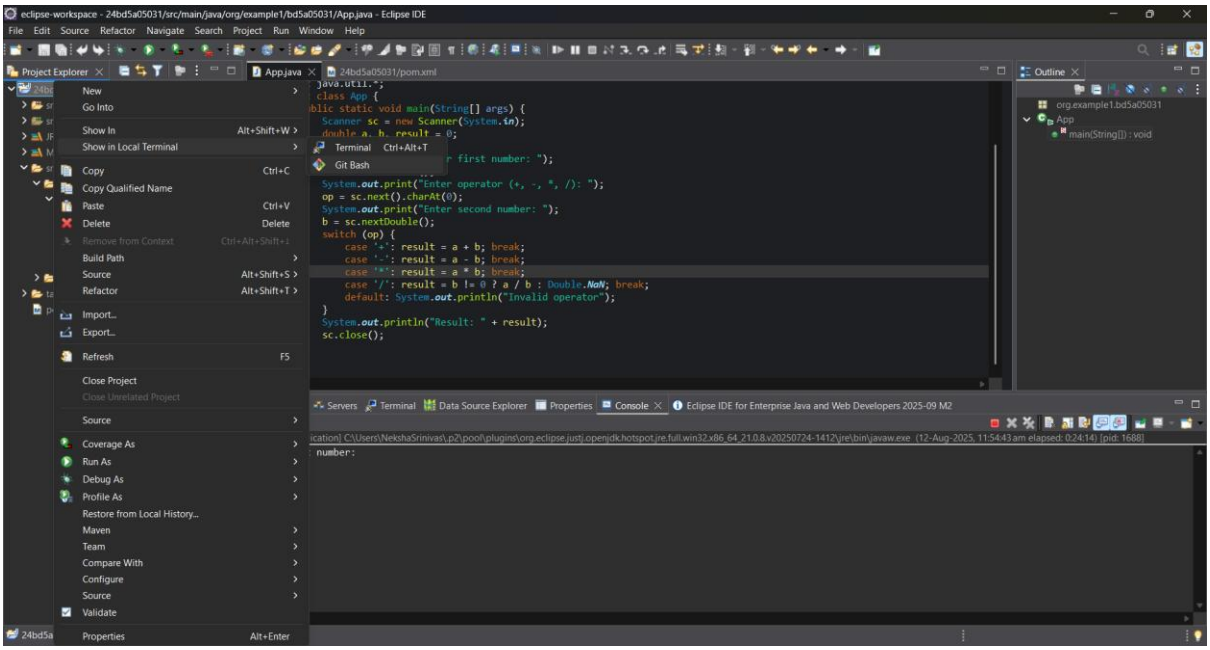
Step-6: BUILD SUCCESS will be shown



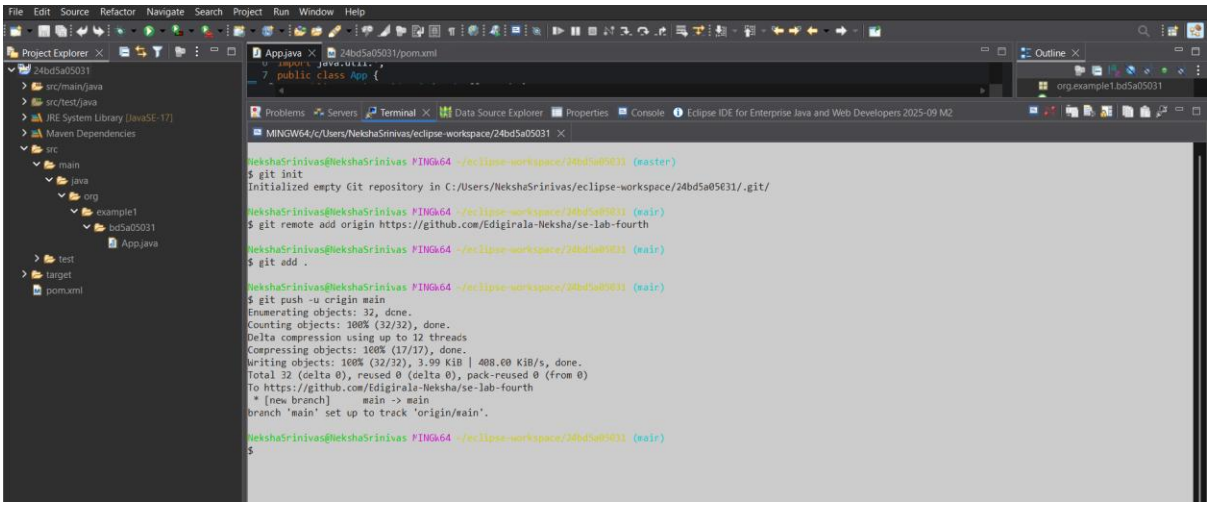
Step-6: write the code in the App.java file



Step-7: right click on the root folder and select show in git bash

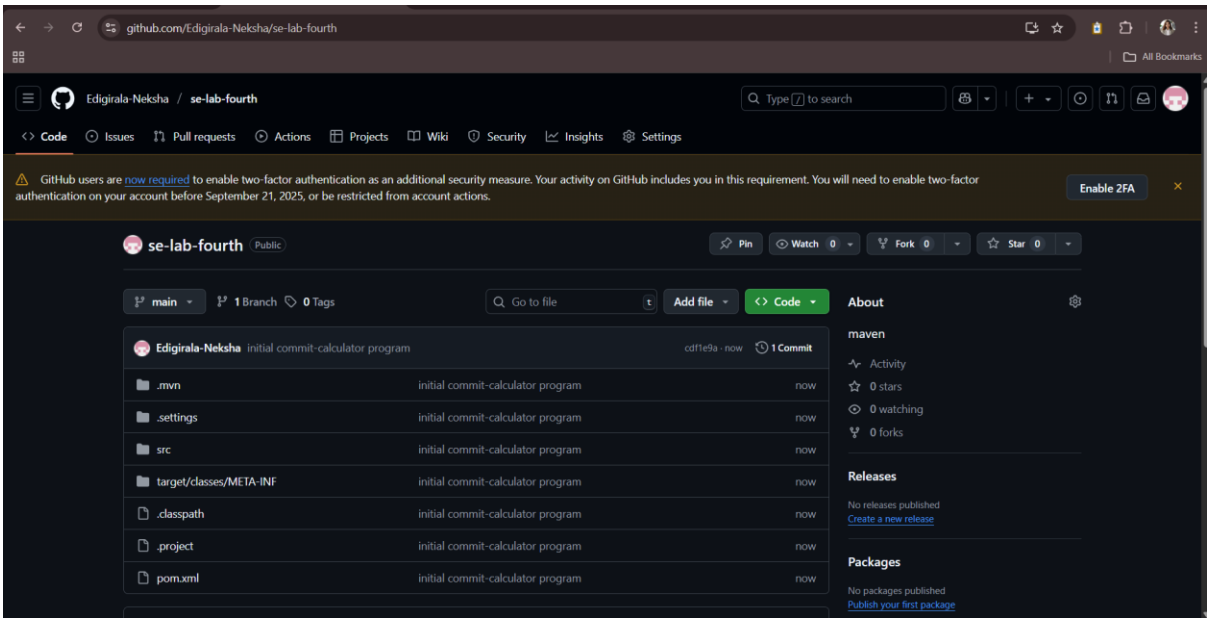


Step-8: push to the git repo



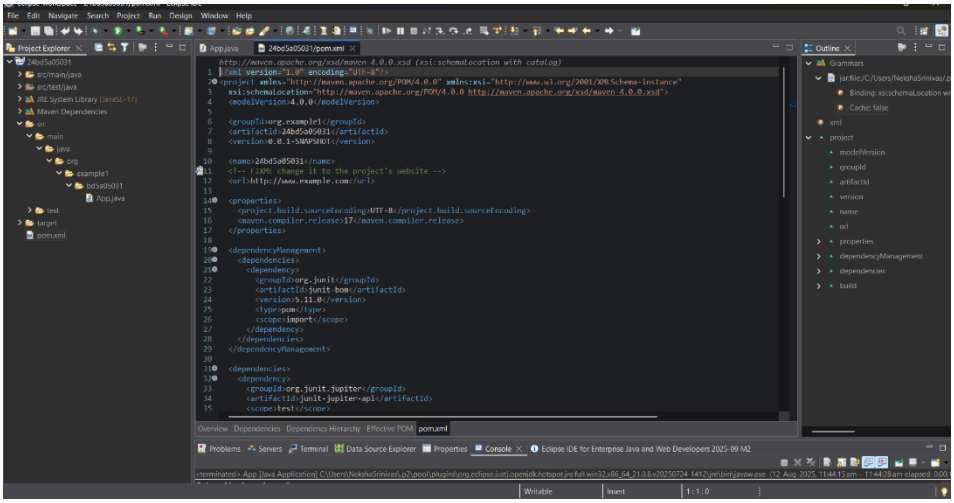
Git repo:

Git repo link: https://github.com/Edigirala-Neksha/se-lab-fourth



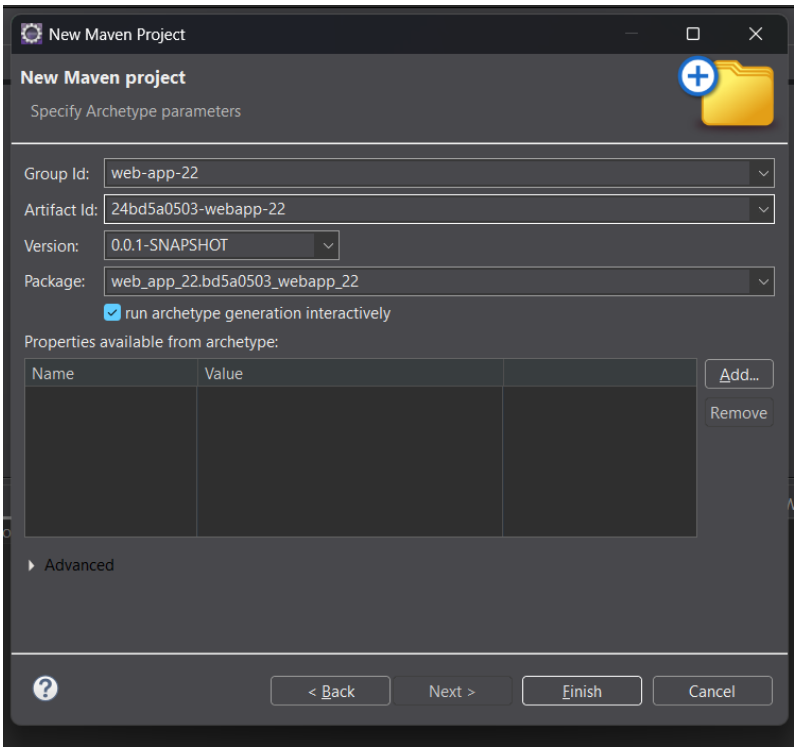
pom.xml file:

Shows the structure-



Creating maven-web project:

Step 1: Create a new maven project and give the details



Step 2: Click y to continue the creation of project

```
C:\Users\NekshaSrinivas\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.8.v20250724-1412\jre\bin\javaw.exe (02-Sept-2025, 7:19:56 pm) [pid: 13772]
Progress (1): 17/17 MB
Progress (1): 17/17 MB
Progress (1): 17/17 MB
Progress (1): 17/17 MB
Progress (1): 17/17 MB
Progress (1): 17 MB

Downloaded from central: https://repo.maven.apache.org/maven2/archetype-catalog.xml (17 MB at 9.1 MB/s)
[INFO] Archetype repository not defined. Using the one from [org.apache.maven.archetypes:maven-archetype-webapp:1.5] found in catalog remote
[INFO] Using property: groupId = web-app-22
[INFO] Using property: artifactId = 24bd5a0503-webapp-22
[INFO] Using property: version = 0.0.1-SNAPSHOT
[INFO] Using property: package = web_app_22.bd5a0503_webapp_22
Confirm properties configuration:
groupId: web-app-22
artifactId: 24bd5a0503-webapp-22
version: 0.0.1-SNAPSHOT
package: web_app_22.bd5a0503_webapp_22
 Y: y
```

Step 3: If the build is success it will show the message

```
package: web_app_22.bd5a0503_webapp_22
 Y: y
[INFO] ----------------------------------------------------------------------------
[INFO] Using following parameters for creating project from Old (1.x) Archetype: maven-archetype-webapp:1.0
[INFO] ----------------------------------------------------------------------------
[INFO] Parameter: basedir, Value: C:\Users\NekshaSrinivas\eclipse-workspace
[INFO] Parameter: package, Value: web_app_22.bd5a0503_webapp_22
[INFO] Parameter: groupId, Value: web-app-22
[INFO] Parameter: artifactId, Value: 24bd5a0503-webapp-22
[INFO] Parameter: packageName, Value: web_app_22.bd5a0503_webapp_22
[INFO] Parameter: version, Value: 0.0.1-SNAPSHOT
[INFO] project created from Old (1.x) Archetype in dir: C:\Users\NekshaSrinivas\eclipse-workspace\24bd5a0503-webapp-22
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  43.500 s
[INFO] Finished at: 2025-09-02T19:20:41+05:30
[INFO] ------------------------------------------------------------------------
```

Step 4: write the html code for the web page:

index.jsp

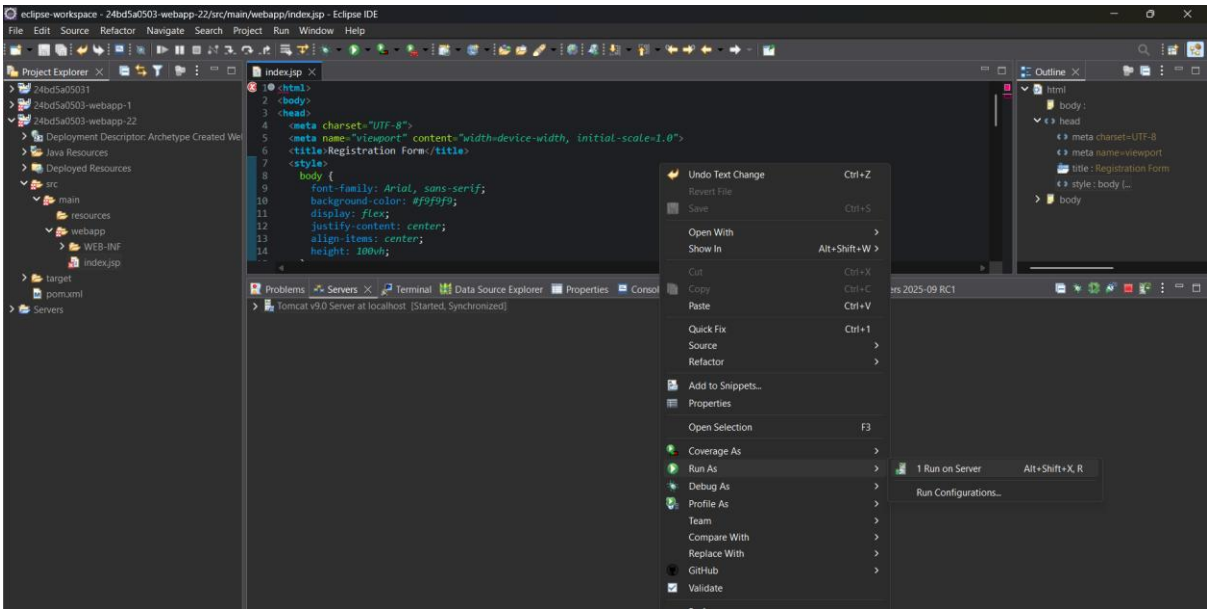
```
<html>
<body>
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Registration Form</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f9f9f9;
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
    }
    .form-container {
      background: #fff;
      padding: 20px 30px;
      border-radius: 10px;
      box-shadow: 0 4px 10px rgba(0,0,0,0.1);
      width: 300px;
    }
    .form-container h2 {
      text-align: center;
      margin-bottom: 20px;
    }
    .form-container input {
      width: 100%;
      padding: 10px;
      margin: 8px 0;
      border: 1px solid #ccc;
      border-radius: 5px;
    }
    .form-container button {
      width: 100%;
      padding: 10px;
      background: #4CAF50;
```

Web-page:

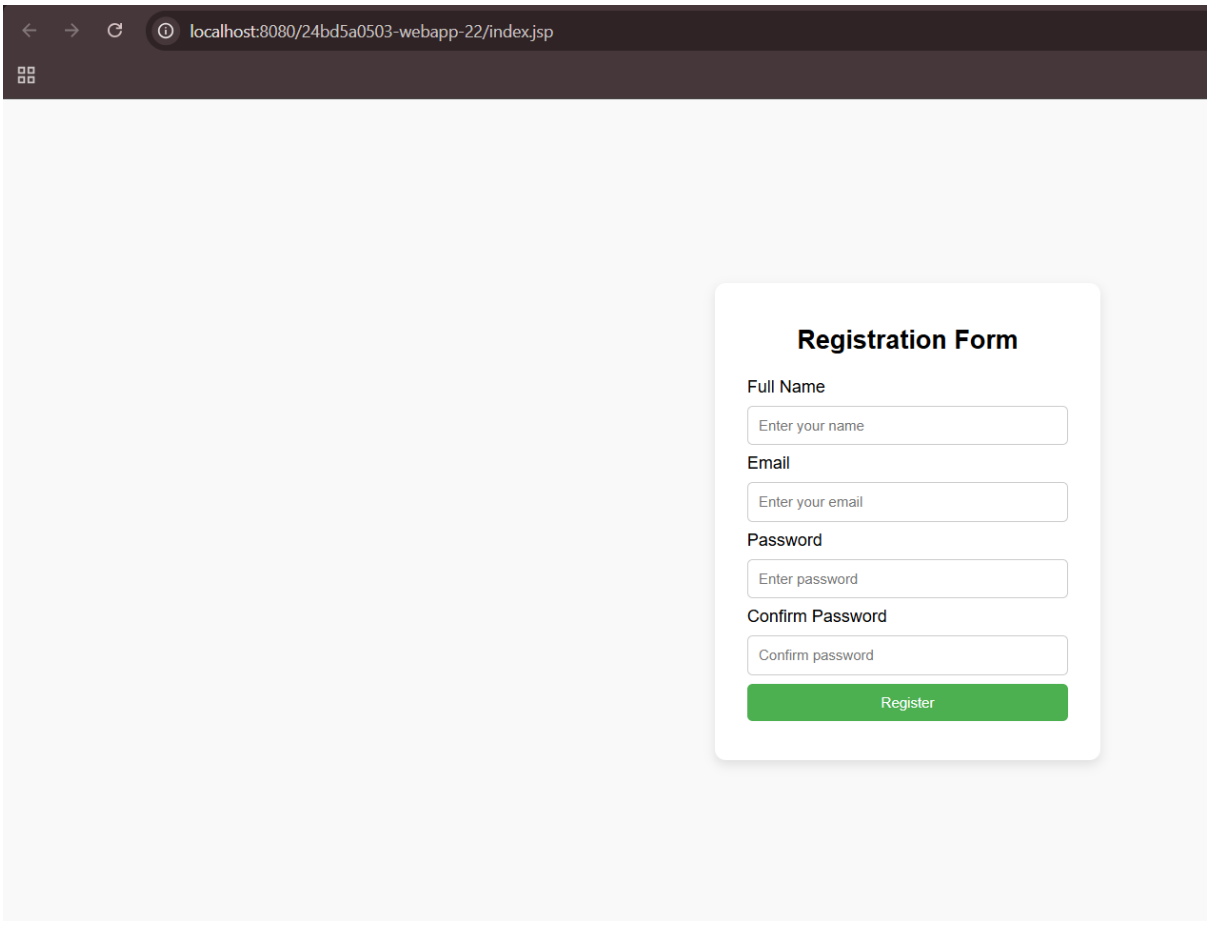
index.jsp

```
    }
    .form-container button {
      width: 100%;
      padding: 10px;
      background: #4CAF50;
      color: white;
      border: none;
      border-radius: 5px;
      cursor: pointer;
    }
    .form-container button:hover {
      background: #45a049;
    }
  </style>
</head>
<body>
  <div class="form-container">
    <h2>Registration Form</h2>
    <form action="#" method="post">
      <label for="fullname">Full Name</label>
      <input type="text" id="fullname" name="fullname" placeholder="Enter your name" required>

      <label for="email">Email</label>
      <input type="email" id="email" name="email" placeholder="Enter your email" required>

      <label for="password">Password</label>
      <input type="password" id="password" name="password" placeholder="Enter password" required>

      <label for="confirm">Confirm Password</label>
      <input type="password" id="confirm" name="confirm" placeholder="Confirm password" required>

      <button type="submit">Register</button>
    </form>
  </div>
</body>
</html>
```

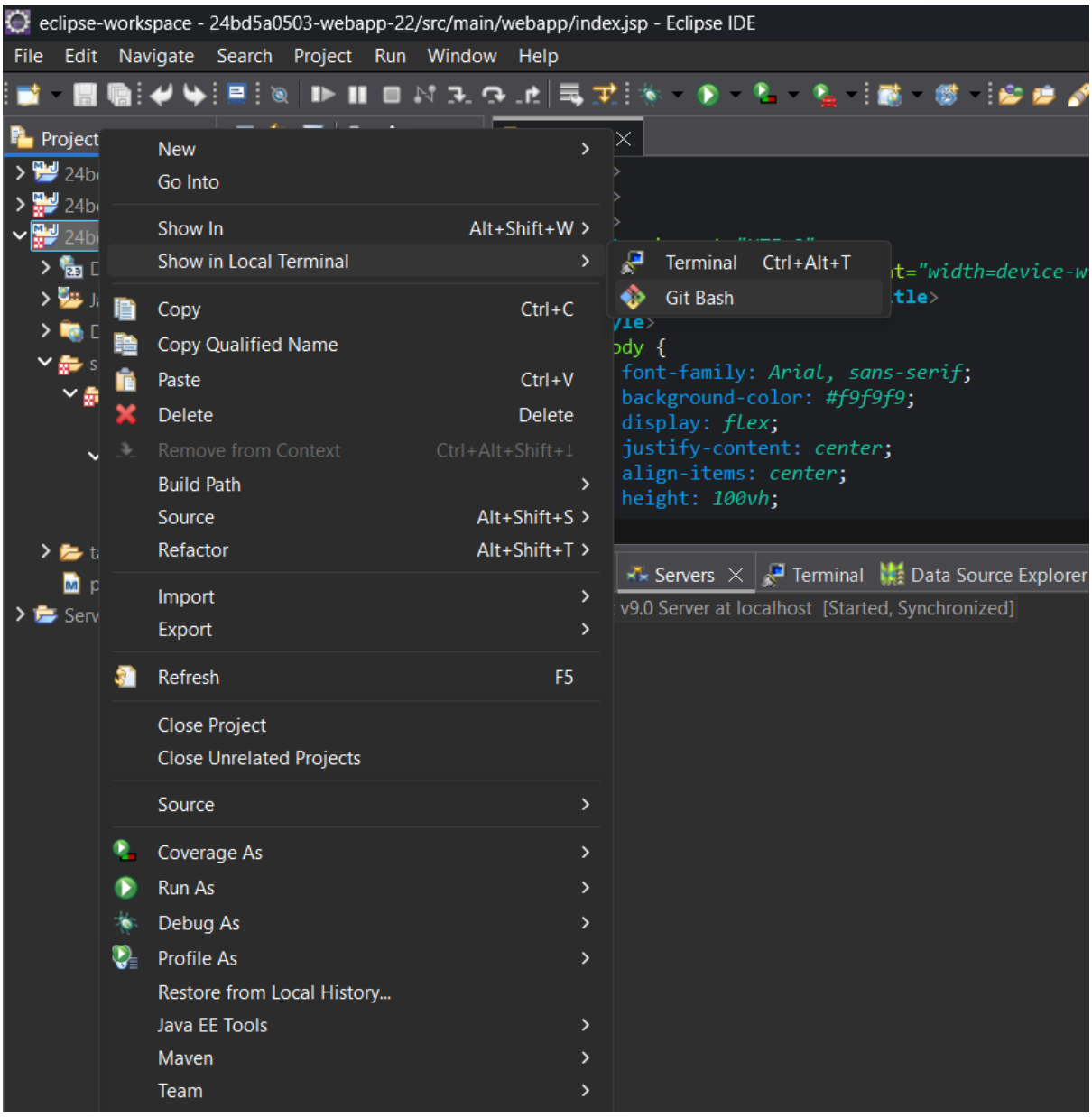
Step 5: Select run on server



Step 6: It will show the following output:



Step 7: To push it into git, select git bash from show in local terminal

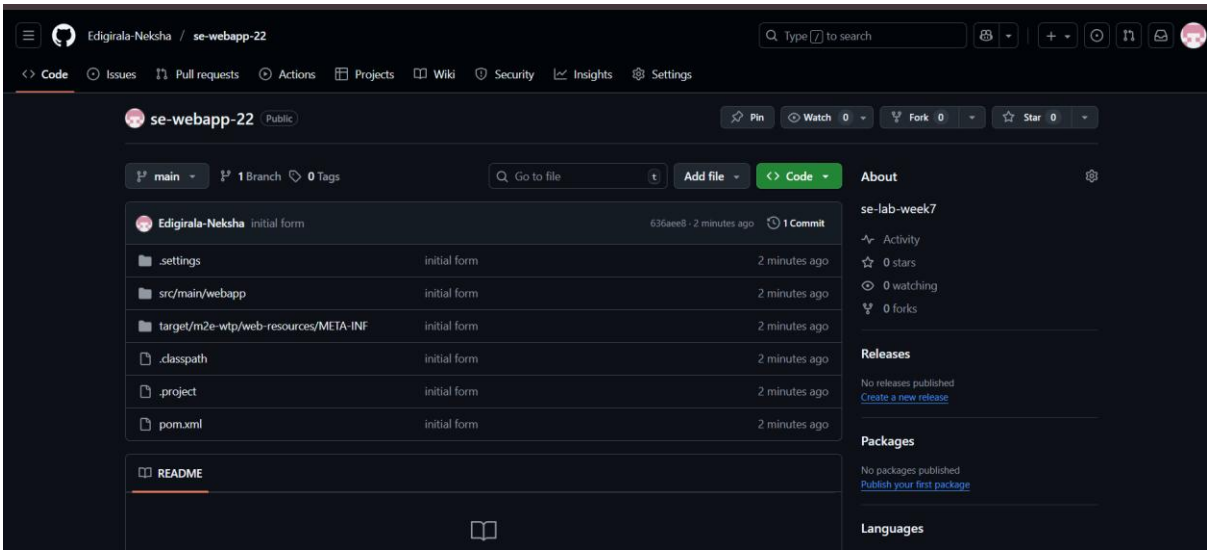


Step 8: use the command of git to push the maven web project

```
MINGW64:/c/Users/NekshaSrinivas/eclipse-workspace/24bd5a05...   MINGW64:/c/Users/NekshaSrinivas/eclipse-workspace/24bd5a05...   MINGW64:

NekshaSrinivas@NekshaSrinivas MINGW64 ~/eclipse-workspace/24bd5a0503-webapp-22 (master)
$ git init
Initialized empty Git repository in C:/Users/NekshaSrinivas/eclipse-workspace/24bd5a0503-webapp-22/.git/

NekshaSrinivas@NekshaSrinivas MINGW64 ~/eclipse-workspace/24bd5a0503-webapp-22 (main)
$ git add .

NekshaSrinivas@NekshaSrinivas MINGW64 ~/eclipse-workspace/24bd5a0503-webapp-22 (main)
$ git commit -m "initial form"
[main (root-commit) 636aee8] initial form
 16 files changed, 254 insertions(+)
 create mode 100644 .classpath
 create mode 100644 .project
 create mode 100644 .settings/.jsdtscope
 create mode 100644 .settings/org.eclipse.jdt.core.prefs
 create mode 100644 .settings/org.eclipse.m2e.core.prefs
 create mode 100644 .settings/org.eclipse.wst.common.component
 create mode 100644 .settings/org.eclipse.wst.common.project.facet.core.xml
 create mode 100644 .settings/org.eclipse.wst.jsdt.ui.superType.container
 create mode 100644 .settings/org.eclipse.wst.jsdt.ui.superType.name
 create mode 100644 .settings/org.eclipse.wst.validation.prefs
 create mode 100644 pom.xml
 create mode 100644 src/main/webapp/WEB-INF/web.xml
```

```
NekshaSrinivas@NekshaSrinivas MINGW64 ~/eclipse-workspace/24bd5a0503-webapp-22 (main)
$ git branch
* main

NekshaSrinivas@NekshaSrinivas MINGW64 ~/eclipse-workspace/24bd5a0503-webapp-22 (main)
$ git push origin main
Enumerating objects: 29, done.
Counting objects: 100% (29/29), done.
Delta compression using up to 12 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (29/29), 4.43 KiB | 283.00 KiB/s, done.
Total 29 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/Edigirala-Neksha/se-webapp-22.git
 * [new branch]      main -> main

NekshaSrinivas@NekshaSrinivas MINGW64 ~/eclipse-workspace/24bd5a0503-webapp-22 (main)
$
```

Step 9: verify the repo in git hub



5. Docker CLI commands

Installing Docker and Setting up Nginx

Introduction

Docker is a platform that allows us to run applications inside lightweight containers. Containers are isolated environments that include everything needed to run an application. This makes it easy to set up and deploy software without worrying about dependencies or configurations on the host system.

In this task, we used Docker to run an **Ubuntu container**, install **nginx** inside it, and serve a customized homepage

Step 1: Pulling the Ubuntu Image

First, we pulled the latest Ubuntu image from Docker Hub.

```
PS C:\Users\NekshaSrinivas> docker --version
Docker version 28.3.2, build 578ccf6
PS C:\Users\NekshaSrinivas> cd SE-1
PS C:\Users\NekshaSrinivas\SE-1> docker --version
Docker version 28.3.2, build 578ccf6
PS C:\Users\NekshaSrinivas\SE-1> docker pull ubuntu:latest
latest: Pulling from library/ubuntu
b71466b94f26: Pull complete
Digest: sha256:7c06e91f61fa88c08cc74f7e1b7c69ae24910d745357e0dfe1d2c0322aaf2
0f9
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
```

Step 2: Running the Container

We created and started a new container named **myubuntu**, mapping port **3000** on the host to port **80** inside the container.

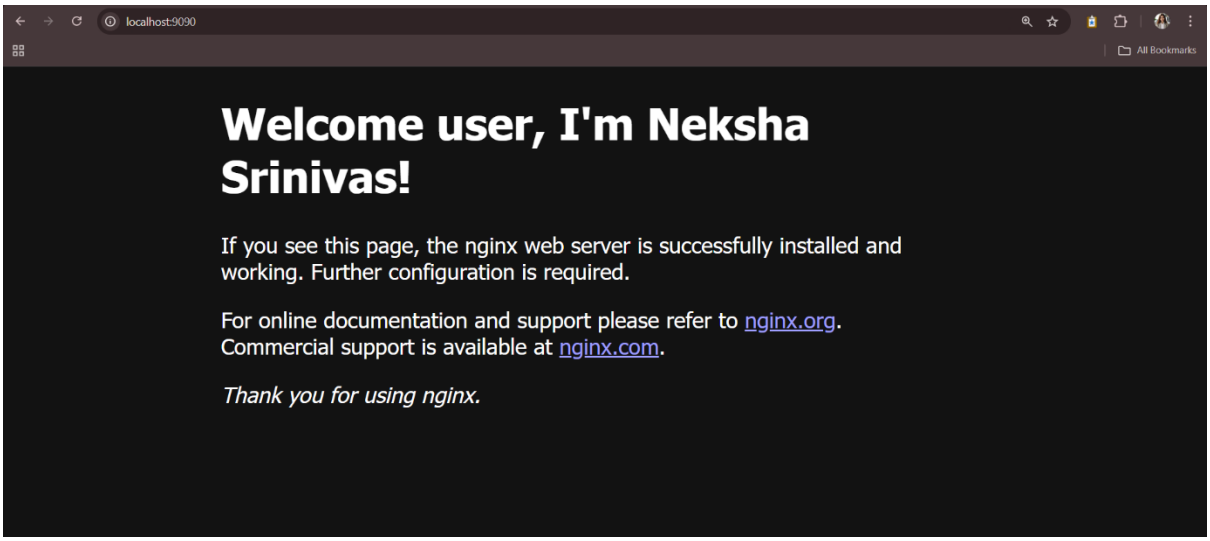
```
PS C:\Users\NekshaSrinivas\SE-1> docker run -it -p 9090:80 --name myubuntu1
ubuntu:latest
root@773ac2012994:/# apt update
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble InRelease [256 kB]
Get:3 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packag
es [1135 kB]
Get:4 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [
1355 kB]
Get:6 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Pack
ages [2047 kB]
Get:8 http://archive.ubuntu.com/ubuntu noble/main amd64 Packages [1808 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Pack
ages [23.0 kB]
Get:10 http://archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [331
 kB]
```

Step 3: Installing Nginx and redirecting to index.html page to edit the content

```
Processing triggers for libc-bin (2.39-0ubuntu8.5) ...
root@773ac2012994:/# ls
bin                dev   lib    mnt   root  sbin.usr-is-merged  tmp
bin.usr-is-merged  etc   lib64  opt   run   srv                 usr
boot               home  media  proc  sbin  sys                 var
root@773ac2012994:/# cd usr
root@773ac2012994:/usr# ls
bin  games  include  lib  lib64  libexec  local  sbin  share  src
root@773ac2012994:/usr# cd share
root@773ac2012994:/usr/share# ls
apport               gcc          pam
base-files           gdb          pam-configs
base-passwd          info         perl5
bash-completion      info.dir     pixmaps
bug                  keyrings     polkit-1
common-licenses      libc-bin     profile
debconf              libgcrypt20  profile.md5sums
debianutils          lintian      sensible-utils
dict                 locale       staff-group-for-usr-local
doc                  man          tabset
doc-base             menu         terminfo
dot.bashrc           misc         util-linux
dot.profile          motd         vim
dot.profile.md5sums  networks
dpkg                 nginx
root@773ac2012994:/usr/share# cd nginx
root@773ac2012994:/usr/share/nginx# ls
html  modules
root@773ac2012994:/usr/share/nginx# cd html
root@773ac2012994:/usr/share/nginx/html# ls
index.html
root@773ac2012994:/usr/share/nginx/html# nano index.html
bash: nano: command not found
root@773ac2012994:/usr/share/nginx/html# apt install nano
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  hunspell
The following NEW packages will be installed:
  nano
```

Step 4: navigate to index.html using command –“nano index.html” Changed the content of h1 tag

Step 5: Viewing the page from local host



6. Docker

DOCKER IMAGE CREATION:

Image can be created in two ways:

1. Using Docker commit
2. Using docker file

Step 1: Created a new container of ubuntu so image can be created on that container

```
PS C:\Users\NekshaSrinivas\SE-1> docker run -it -p 9090:80 --name ubuntu-con
t-1 ubuntu:latest
root@74098c332e58:/# apt update
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble InRelease [256 kB]
Get:3 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packag
es [1137 kB]
Get:4 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Pack
ages [2066 kB]
Get:6 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [
1363 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Pack
ages [23.0 kB]
Get:9 http://archive.ubuntu.com/ubuntu noble/restricted amd64 Packages [117
kB]
Get:10 http://archive.ubuntu.com/ubuntu noble/universe amd64 Packages [19.3
MB]
Get:11 http://archive.ubuntu.com/ubuntu noble/main amd64 Packages [1808 kB]
Get:12 http://archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [331
 kB]
Get:13 http://archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packa
```

Step 2: Using commit the image is being created:

```
PS C:\Users\NekshaSrinivas\SE-1> docker commit ubuntu-cont-1 img-commit-1
sha256:153126502820131f25f36cc59f7c4557275621bcdc7a54b48c8ffd4409685efd
PS C:\Users\NekshaSrinivas\SE-1> docker images
REPOSITORY     TAG       IMAGE ID       CREATED         SIZE
img-commit-1   latest    153126502820   7 seconds ago   326MB
mynginx        latest    de77ca8d52cb   30 hours ago    279MB
mypythonapp    latest    8a39b6d82115   30 hours ago    1.63GB
nginx          latest    33e0bbc7ca9e   12 days ago     279MB
ubuntu         latest    7c06e91f61fa   3 weeks ago     117MB
PS C:\Users\NekshaSrinivas\SE-1> docker run -it img-commit-1
root@909ab066a51f:/# git --version
git version 2.43.0
root@909ab066a51f:/# docker tag img-commit-1 nekshasrinivas/img-commit-1
bash: docker: command not found
root@909ab066a51f:/# exit
exit
PS C:\Users\NekshaSrinivas\SE-1> docker tag img-commit-1 nekshasrinivas/img-
commmit-1
PS C:\Users\NekshaSrinivas\SE-1> docker push nekshasrinivas/img-commmit-1
Using default tag: latest
The push refers to repository [docker.io/nekshasrinivas/img-commmit-1]
4024494ad21b: Pushed
b71466b94f26: Mounted from library/ubuntu
latest: digest: sha256:153126502820131f25f36cc59f7c4557275621bcdc7a54b48c8ff
d4409685efd size: 751
```

Step 3: Image creation using docker file

```
PS C:\Users\NekshaSrinivas\SE-1> mkdir image-creation


    Directory: C:\Users\NekshaSrinivas\SE-1


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        26-08-2025     18:26                image-creation


PS C:\Users\NekshaSrinivas\SE-1> ls


    Directory: C:\Users\NekshaSrinivas\SE-1


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        26-08-2025     18:26                image-creation
d-----        25-08-2025     12:25                static_site
-a----        25-08-2025     12:15             36 app.py
-a----        25-08-2025     12:15            100 Dockerfile
-a----        23-08-2025     16:28       29739008 myapi.tar
-a----        05-08-2025     16:33             38 README.md


PS C:\Users\NekshaSrinivas\SE-1> cd image-creation
PS C:\Users\NekshaSrinivas\SE-1\image-creation> notepad Dockerfile
PS C:\Users\NekshaSrinivas\SE-1\image-creation> ls


    Directory: C:\Users\NekshaSrinivas\SE-1\image-creation


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        26-08-2025     18:27             59 Dockerfile.txt


PS C:\Users\NekshaSrinivas\SE-1\image-creation> ren Dockerfile.txt Dockerfile
```

Step 4: after writing the content in docker file use the command docker build

```
PS C:\Users\NekshaSrinivas\SE-1\image-creation> docker build -t img-dockerfile-1 .
[+] Building 44.0s (7/7) FINISHED                          docker:desktop-linux
 => [internal] load build definition from Dockerfile                       0.1s
 => => transferring dockerfile: 96B                                        0.0s
 => [internal] load metadata for docker.io/library/ubuntu:latest           0.1s
 => [internal] load .dockerignore                                          0.1s
 => => transferring context: 2B                                            0.0s
 => [1/3] FROM docker.io/library/ubuntu:latest@sha256:7c06e91f61fa88c      0.1s
 => => resolve docker.io/library/ubuntu:latest@sha256:7c06e91f61fa88c      0.0s
 => [2/3] RUN apt-get update                                              12.5s
 => [3/3] RUN apt-get install git -y                                      24.4s
 => exporting to image                                                     6.5s
 => => exporting layers                                                    4.5s
 => => exporting manifest sha256:99d816a6b717e709d838937a995f24d0121e      0.0s
 => => exporting config sha256:f021a40f65d4b684b65cd403292af90ec68210      0.0s
 => => exporting attestation manifest sha256:04dc38eee96b84e155b083e5      0.1s
 => => exporting manifest list sha256:9868ecb2df510b52e539c55076bf63c      0.0s
 => => naming to docker.io/library/img-dockerfile-1:latest                 0.0s
 => => unpacking to docker.io/library/img-dockerfile-1:latest              1.8s
PS C:\Users\NekshaSrinivas\SE-1\image-creation> docker run -it img-dockerfile-1
root@adfe97a50685:/# docker --version
bash: docker: command not found
root@adfe97a50685:/# git --version
git version 2.43.0
root@adfe97a50685:/# exit
exit
PS C:\Users\NekshaSrinivas\SE-1\image-creation> docker tag img-dockerfile-1 nekshasrinivas/img-dockerfile-1
PS C:\Users\NekshaSrinivas\SE-1\image-creation> docker push nekshasrinivas/img-dockerfile-1
Using default tag: latest
The push refers to repository [docker.io/nekshasrinivas/img-dockerfile-1]
6a5ccfd4b031: Pushed
eddb67216c21: Pushed
b71466b94f26: Mounted from nekshasrinivas/img-commmit-1
004a734bd8b1: Pushed
latest: digest: sha256:9868ecb2df510b52e539c55076bf63ccae47b54ab67e29de352ddbc3cb33b109 size: 855
```

Step 5: checking the images

```
PS C:\Users\NekshaSrinivas\SE-1> docker images
REPOSITORY                          TAG       IMAGE ID       CREATED          SIZE
img-dockerfile-1                    latest    9868ecb2df51   5 minutes ago    326MB
nekshasrinivas/img-dockerfile-1     latest    9868ecb2df51   5 minutes ago    326MB
img-commit-1                        latest    153126502820   12 minutes ago   326MB
nekshasrinivas/img-commmit-1        latest    153126502820   12 minutes ago   326MB
mynginx                             latest    de77ca8d52cb   30 hours ago     279MB
mypythonapp                         latest    8a39b6d82115   30 hours ago     1.63GB
nginx                               latest    33e0bbc7ca9e   12 days ago      279MB
ubuntu                              latest    7c06e91f61fa   3 weeks ago      117MB
PS C:\Users\NekshaSrinivas\SE-1>
```

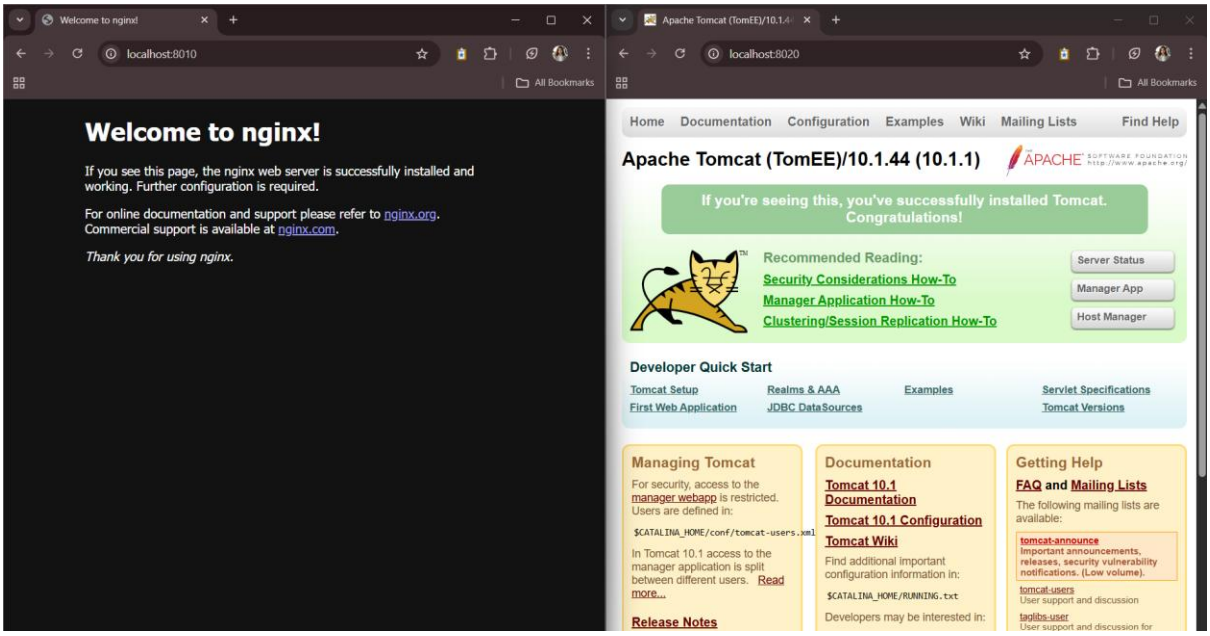
DOCKER COMPOSE FILE:

Docker Compose is a tool used to define and run multi-container Docker applications. It allows you to define services, networks, and volumes that your application needs, all in a single file. This makes it easier to manage complex applications that require multiple containers (e.g., a web server and a database).

Step 1: Running two servers at the same time on different ports

```
PS C:\Users\NekshaSrinivas\SE-1> docker run -d -p 8010:80 nginx
2ea4a201f197b93276310a7d23f2a46060ba9c7387f869e8a2a804931b66b2d9
PS C:\Users\NekshaSrinivas\SE-1> docker run -d -p 8020:8080 tomee
3a524036f6b212843be468585f80fb029aed07715a8e33a38e4eb306044765a2
PS C:\Users\NekshaSrinivas\SE-1>
```

Step 2: Open the local host to view the pages



Step 3: Using docker file to run two servers parallelly

docker-compose.yml

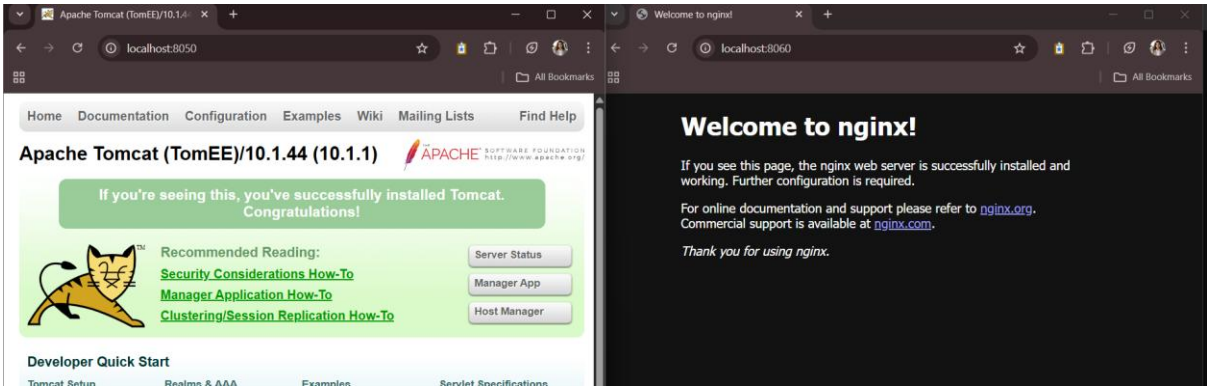
C: > Users > NekshaSrinivas > SE-1 > comp-1-server > docker-compose.yml

```yaml
services:
  web:
    image: nginx
    ports:
      - "8060:80"
  db:
    image: tomee
    ports:
      - "8050:8080"

```

Step 4: Use the docker-compose up -d command to execute the docker file

```
PS C:\Users\NekshaSrinivas\SE-1\comp-1-server> ren Dockerfile docker-compose
.yml
PS C:\Users\NekshaSrinivas\SE-1\comp-1-server> docker-compose up -d
[+] Running 3/3
 ✔Network comp-1-server_default  C...                              0.1s
 ✔Container comp-1-server-db-1   St...                             0.6s
 ✔Container comp-1-server-web-1  S...                              0.7s
PS C:\Users\NekshaSrinivas\SE-1\comp-1-server>
```

Step 5: Open the localhost to view the servers



WORD-PRESS:

Step 1: Create a docker-compose file and write the content for wordpress and mysql

```
PS C:\Users\NekshaSrinivas\SE-1> cd mysql
PS C:\Users\NekshaSrinivas\SE-1\mysql> notepad docker-compose
PS C:\Users\NekshaSrinivas\SE-1\mysql> ls


    Directory: C:\Users\NekshaSrinivas\SE-1\mysql


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        26-08-2025     18:48            672 docker-compose.txt


PS C:\Users\NekshaSrinivas\SE-1\mysql>
PS C:\Users\NekshaSrinivas\SE-1\mysql> ren docker-compose.txt docker-compose
.yml
PS C:\Users\NekshaSrinivas\SE-1\mysql> ls


    Directory: C:\Users\NekshaSrinivas\SE-1\mysql


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        26-08-2025     18:48            672 docker-compose.yml
```

Step 2: docker-compose.yml file:

docker-compose.yml

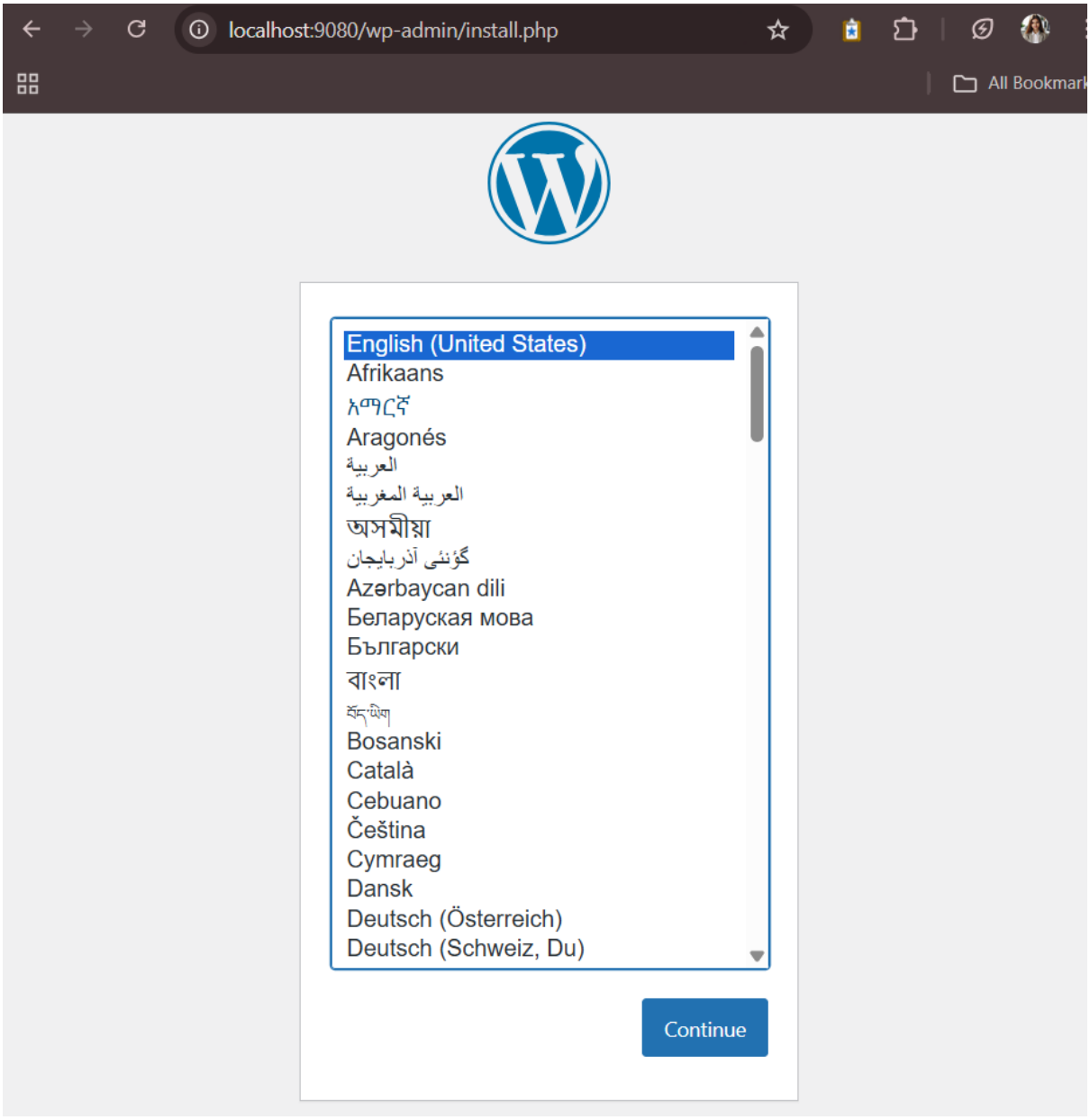
C: > Users > NekshaSrinivas > SE-1 > mysql > docker-compose.yml

```yaml
services:
  wordpress:  # WordPress service
    image: wordpress:latest
    ports:
      - "9080:80"  # Map port 80 of the container to port 8080 of the host
    environment:
      WORDPRESS_DB_HOST: db:3306  # Database host
      WORDPRESS_DB_USER: wordpress
      WORDPRESS_DB_PASSWORD: wordpress
      WORDPRESS_DB_NAME: wordpress
    depends_on:
      - db  # Ensures the db service starts first

  db:  # MySQL service
    image: mysql:5.7
    environment:
      MYSQL_ROOT_PASSWORD: rootpassword
      MYSQL_DATABASE: wordpress
      MYSQL_USER: wordpress
      MYSQL_PASSWORD: wordpress
```

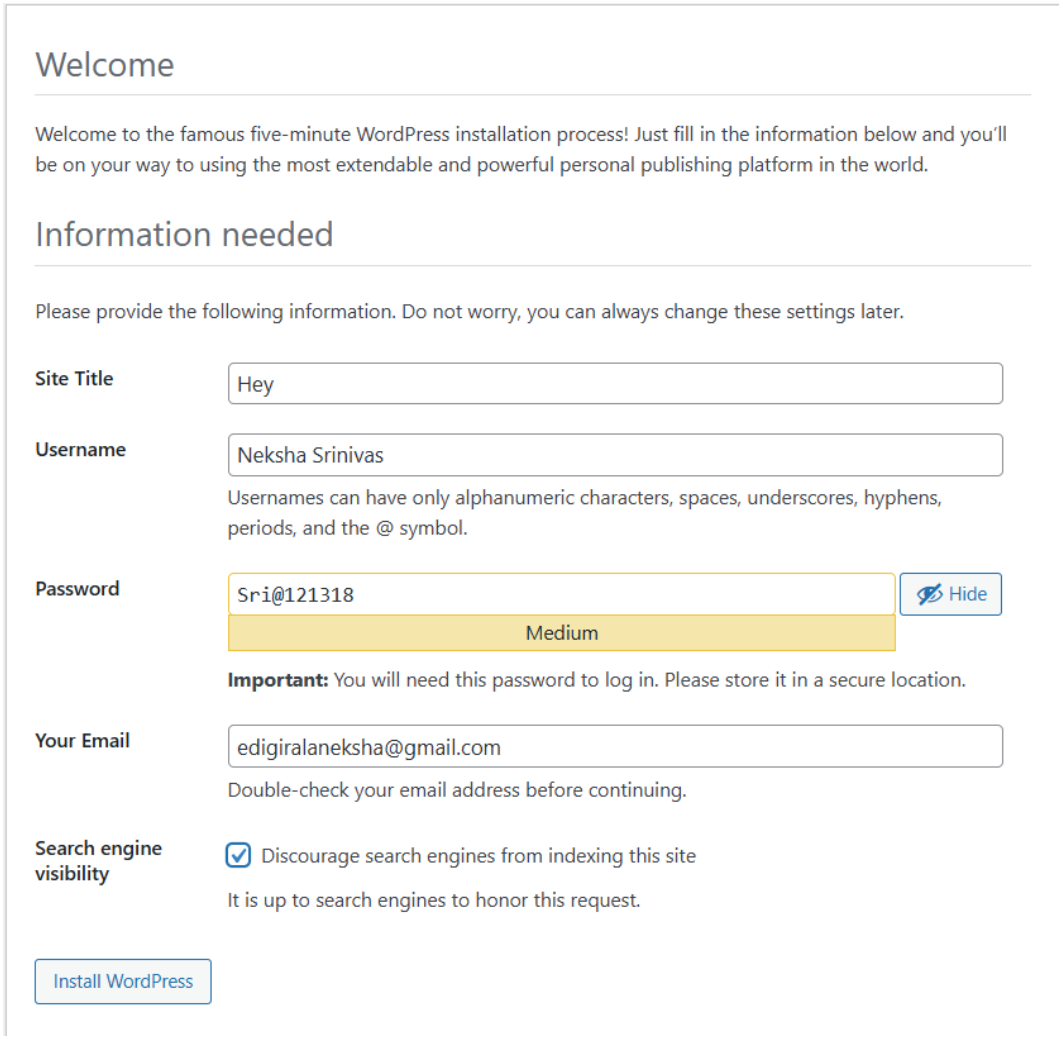
Step 3: Use the docker-compose up -d command to start the compose

```
PS C:\Users\NekshaSrinivas\SE-1\mysql> docker-compose up -d
[+] Running 3/3
 ✔Network mysql_default         Created                    0.1s
 ✔Container mysql-db-1          Started                    0.8s
 ✔Container mysql-wordpress-1   Start...                   1.0s
PS C:\Users\NekshaSrinivas\SE-1\mysql>
```

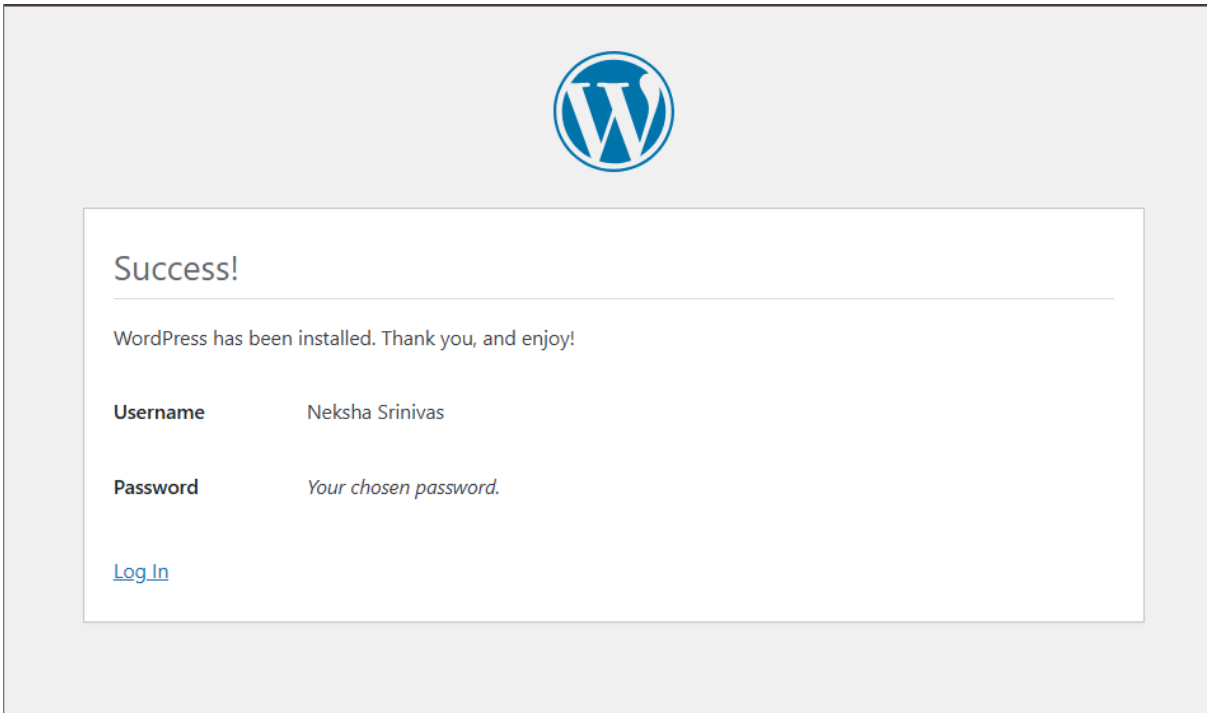
Step 4: Open in the local host and select the language



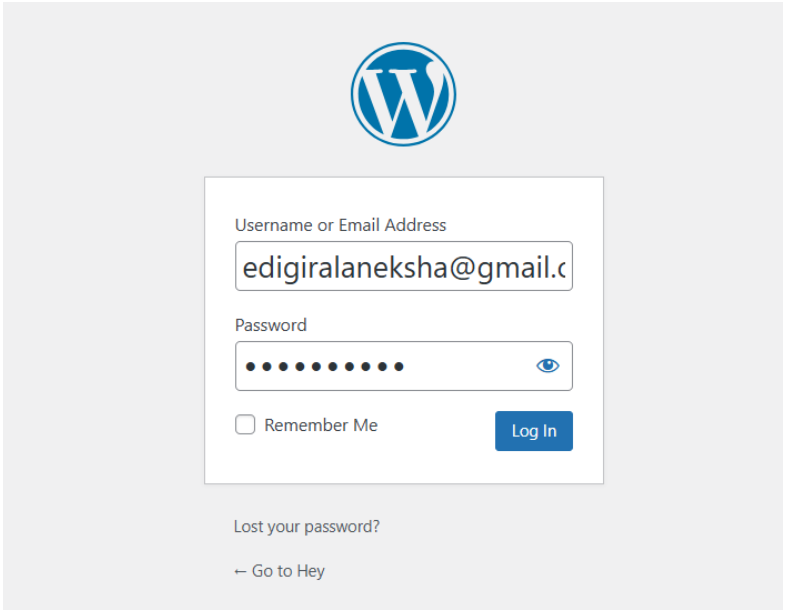
Step 5: Fill the details in the welcome page



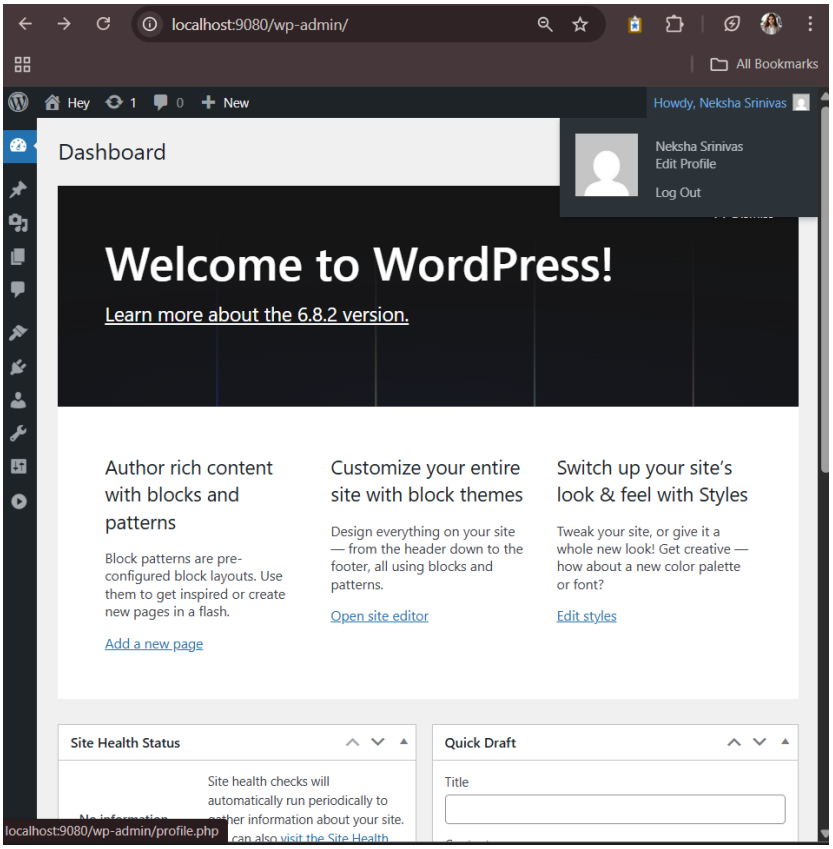
Step 6: Success message will be shown



Step 7: Use your credentials to log in



Step 7: The following page will be shown after login



Task:

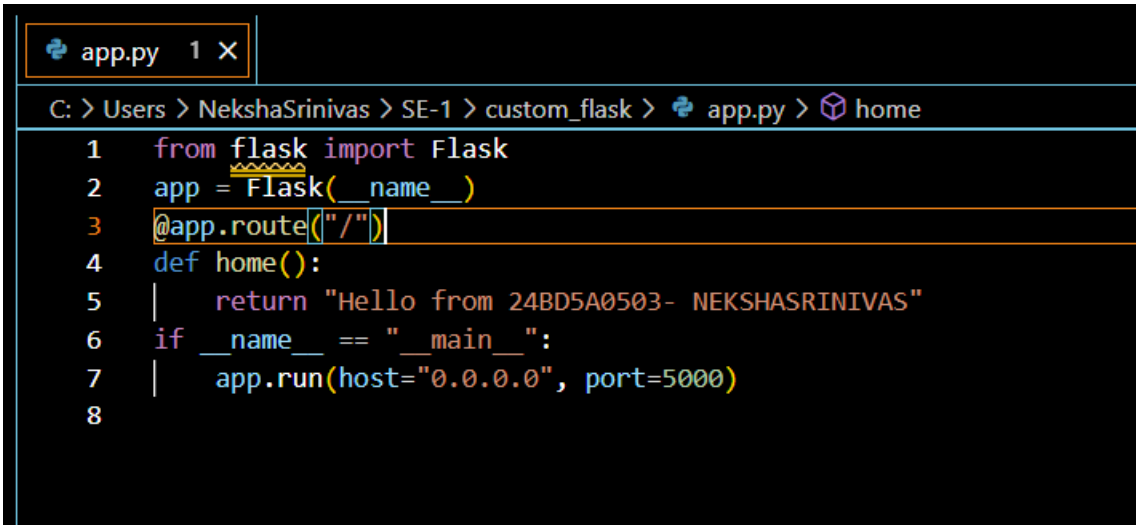
Create a simple Flask app in app.py:

Step 1: create a separate folder

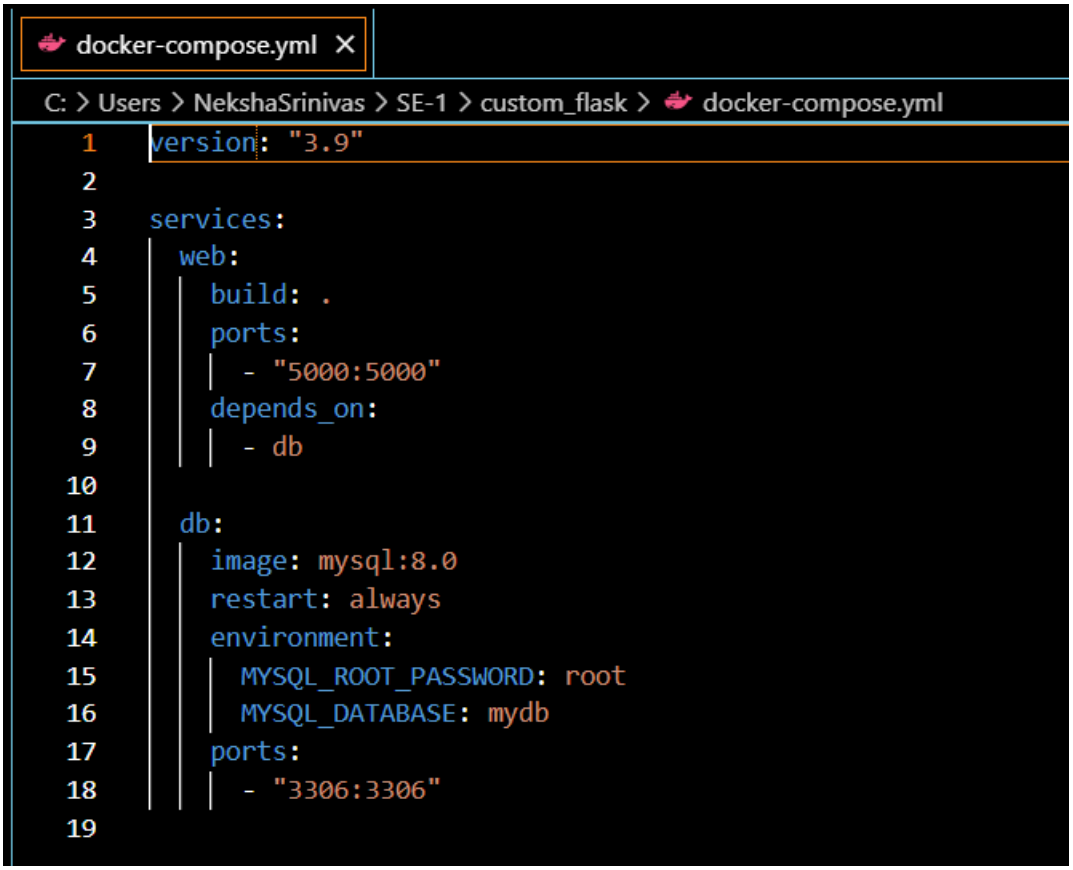
```
PS C:\Users\NekshaSrinivas\SE-1> mkdir custom_flask


    Directory: C:\Users\NekshaSrinivas\SE-1


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        28-08-2025     10:01                custom_flask


PS C:\Users\NekshaSrinivas\SE-1> cd custom_flask
PS C:\Users\NekshaSrinivas\SE-1\custom_flask> notepad app.py
PS C:\Users\NekshaSrinivas\SE-1\custom_flask> notepad Dockerfile
PS C:\Users\NekshaSrinivas\SE-1\custom_flask> ren Dockerfile.txt Dockerfile
PS C:\Users\NekshaSrinivas\SE-1\custom_flask> ls


    Directory: C:\Users\NekshaSrinivas\SE-1\custom_flask


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        28-08-2025     10:02            187 app.py
-a----        28-08-2025     10:02            105 Dockerfile


PS C:\Users\NekshaSrinivas\SE-1\custom_flask> notepad docker-compose.yml
```

Step 2: write the content of app.py , docker-compose.yml & Dockerfile

app.py:



docker-compose.yml:



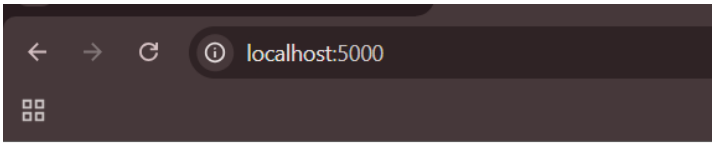
Dockerfile:

```
FROM python:3.10-slim
WORKDIR /app
COPY app.py /app/
RUN pip install flask
CMD ["python", "app.py"]
```

Step 3: run the compose using the command docker compose up –build:

```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        28-08-2025     10:02            187 app.py
-a----        28-08-2025     10:03             82 docker-compose.yml
-a----        28-08-2025     10:02            105 Dockerfile


PS C:\Users\NekshaSrinivas\SE-1\custom_flask> docker compose up --build
validating C:\Users\NekshaSrinivas\SE-1\custom_flask\docker-compose.yml:  ad
ditional properties 'web' not allowed
PS C:\Users\NekshaSrinivas\SE-1\custom_flask> docker compose up --build
time="2025-08-28T10:24:45+05:30" level=warning msg="C:\\Users\\NekshaSriniva
s\\SE-1\\custom_flask\\docker-compose.yml: the attribute `version` is obsole
te, it will be ignored, please remove it to avoid potential confusion"
[+] Running 12/12
 ✔db Pulled                                                          75.7s
   ✔04fa42a56901 Pull complete                                        1.8s
   ✔500d7b2546c4 Pull complete                                       38.2s
   ✔ecc6cc933381 Pull complete                                       38.5s
   ✔5cd63fb67c17 Pull complete                                        1.7s
   ✔4d3eacc36b14 Pull complete                                        1.7s
   ✔9476b8faedba Pull complete                                        3.5s
   ✔789fa151603e Pull complete                                        1.8s
   ✔1756a372d796 Pull complete                                        1.9s
   ✔bc0f5543b464 Pull complete                                       67.6s
   ✔131412d69359 Pull complete                                       42.5s
   ✔03ca01bc78d4 Pull complete                                        1.8s
#1 [internal] load local bake definitions
#1 reading from stdin 542B done
#1 DONE 0.0s

#2 [internal] load build definition from Dockerfile
#2 transferring dockerfile: 142B 0.0s done
#2 DONE 0.1s

#3 [internal] load metadata for docker.io/library/python:3.10-slim
```

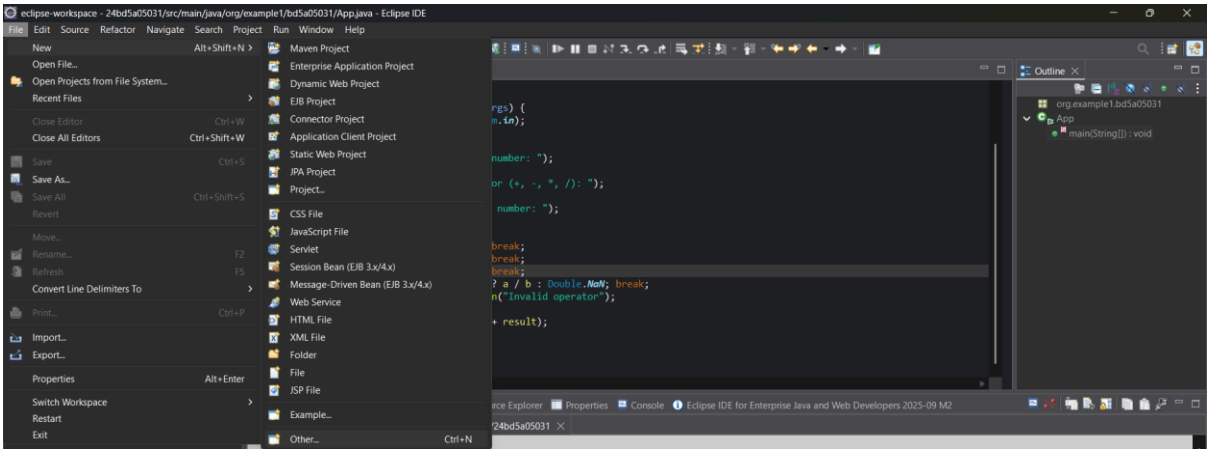
Step 4: Open the local host to view the custom page:



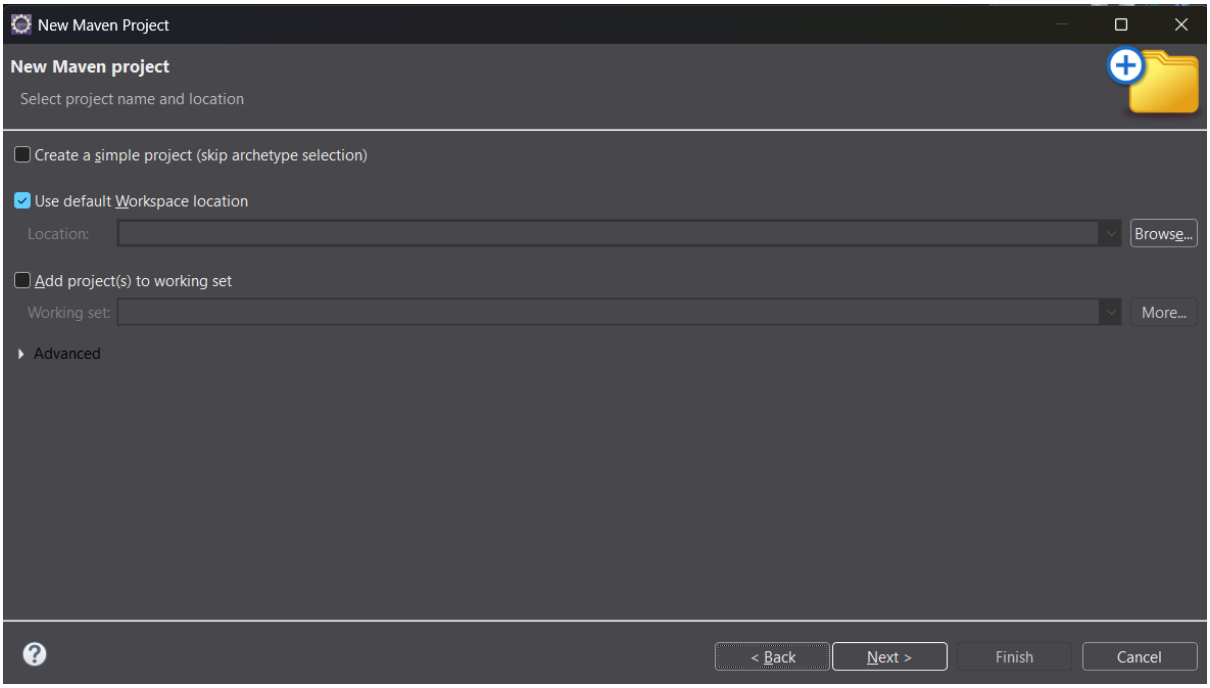
Hello from 24BD5A0503- NEKSHASRINIVAS

7. Creating a Multi-Module Maven Project

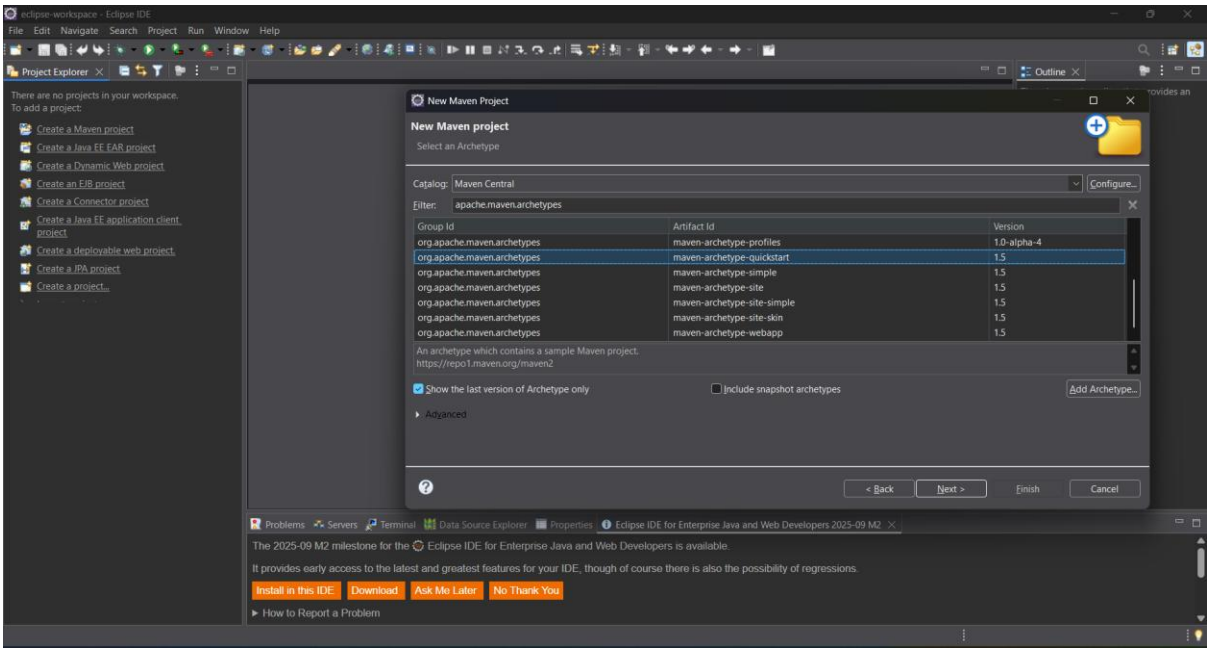
Step-1: Open the eclipse and click on file>new>Maven project



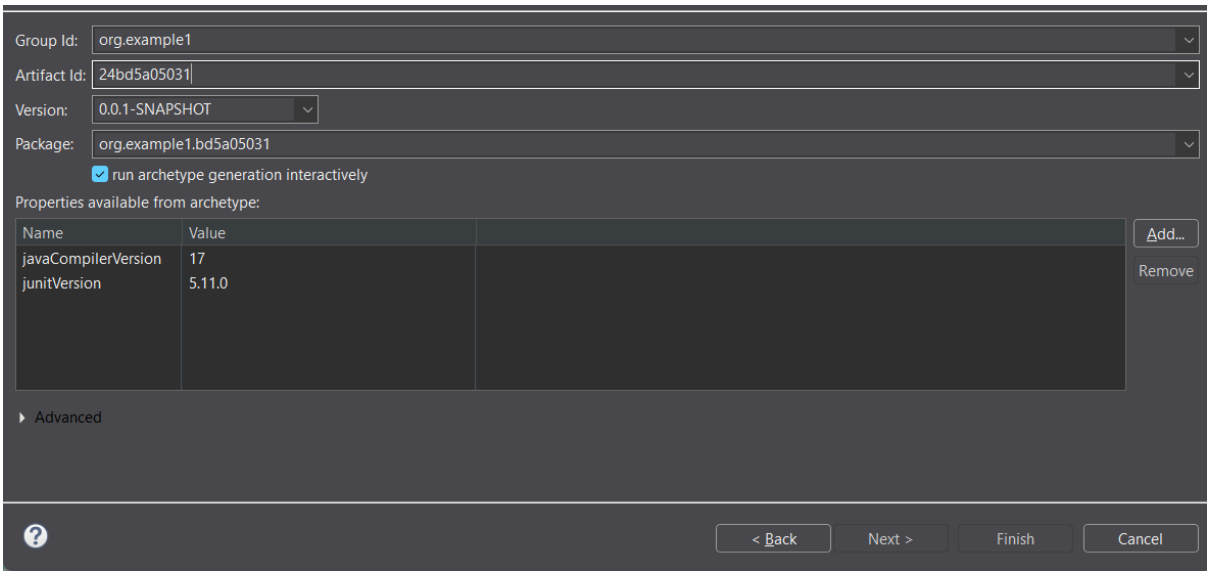
Step-2: select the default workspace and click on next



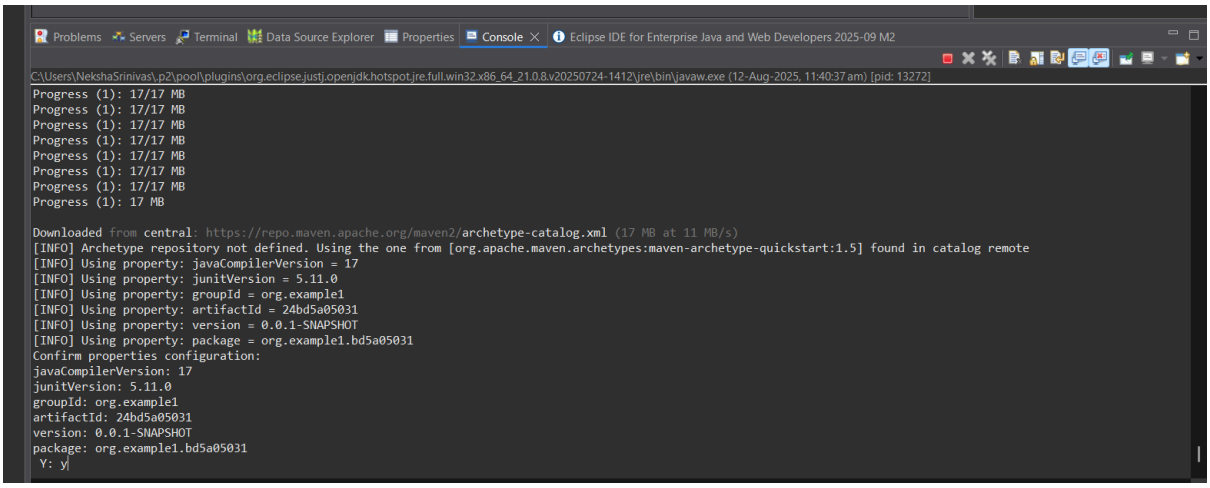
Step-3: in the filter option select the one maven-archetype-quickstart



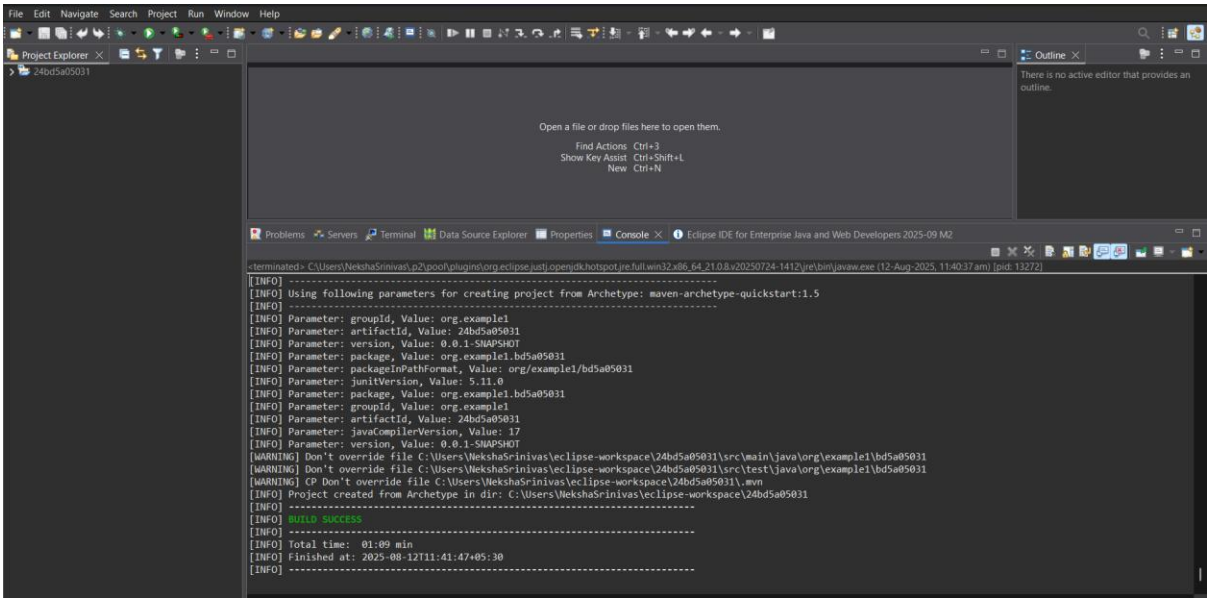
Step-4: give the Group Id and Artifact Id and click on next



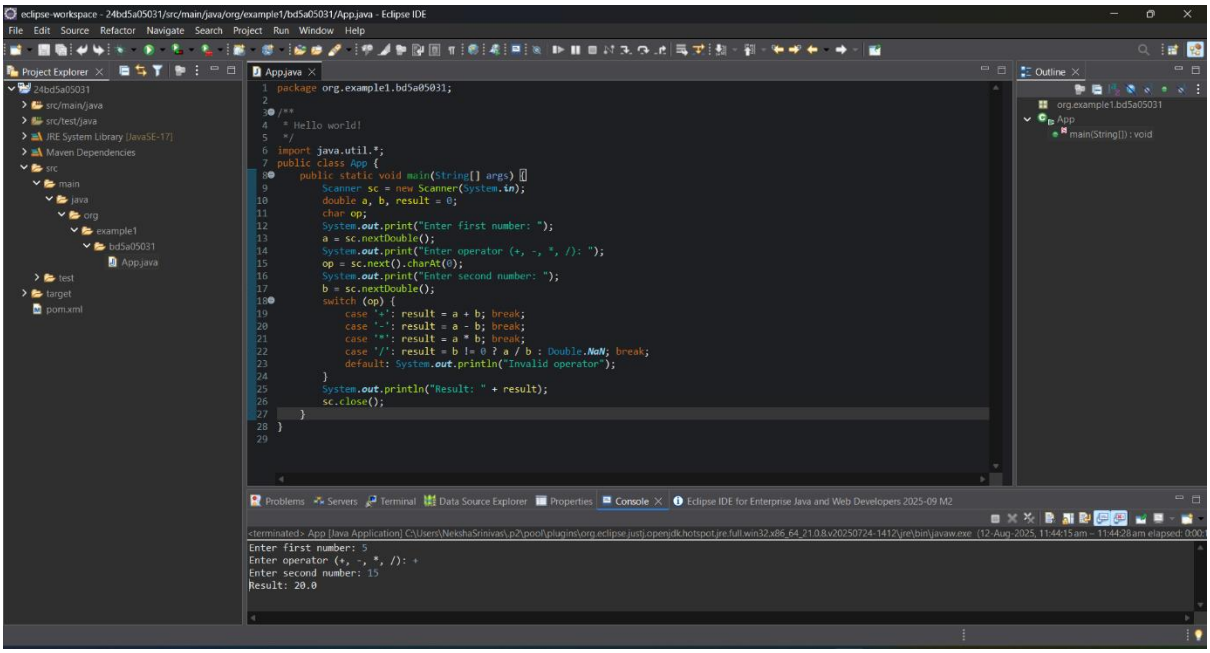
Step-5: In the console the progress will be showed type y (refers to yes) and press enter



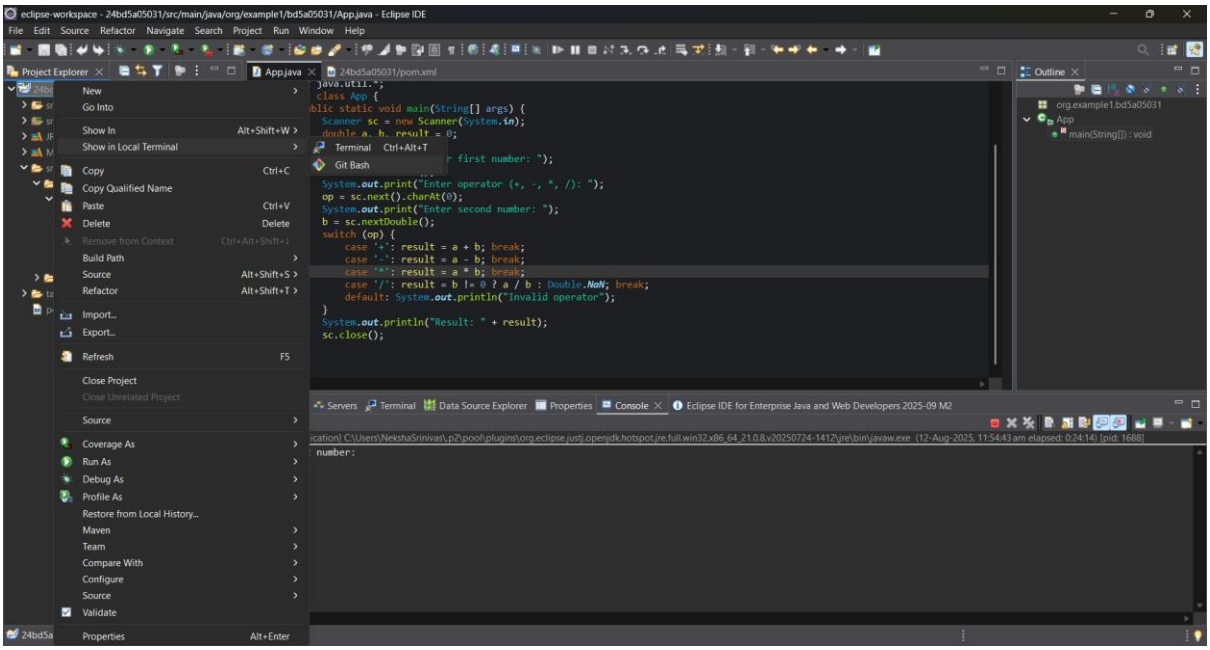
Step-6: BUILD SUCCESS will be shown



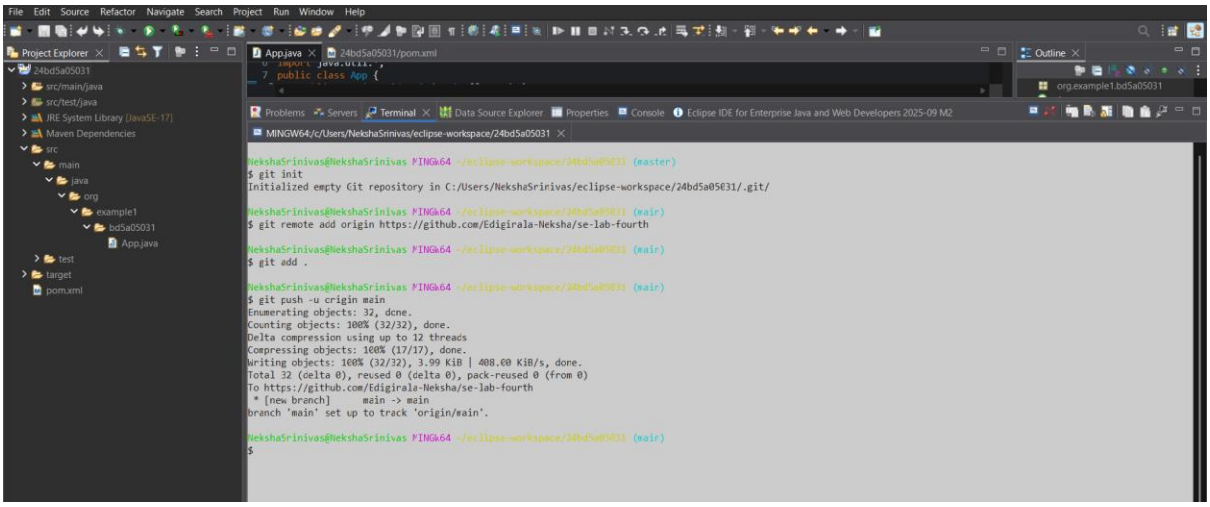
Step-6: write the code in the App.java file



Step-7: right click on the root folder and select show in git bash

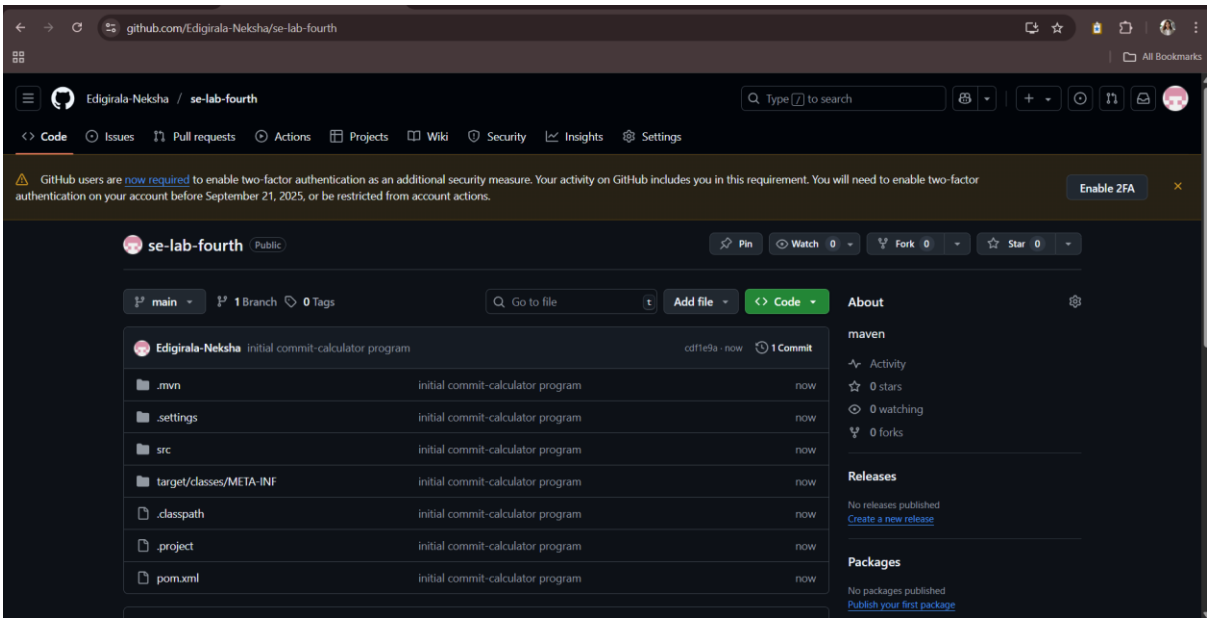


Step-8: push to the git repo



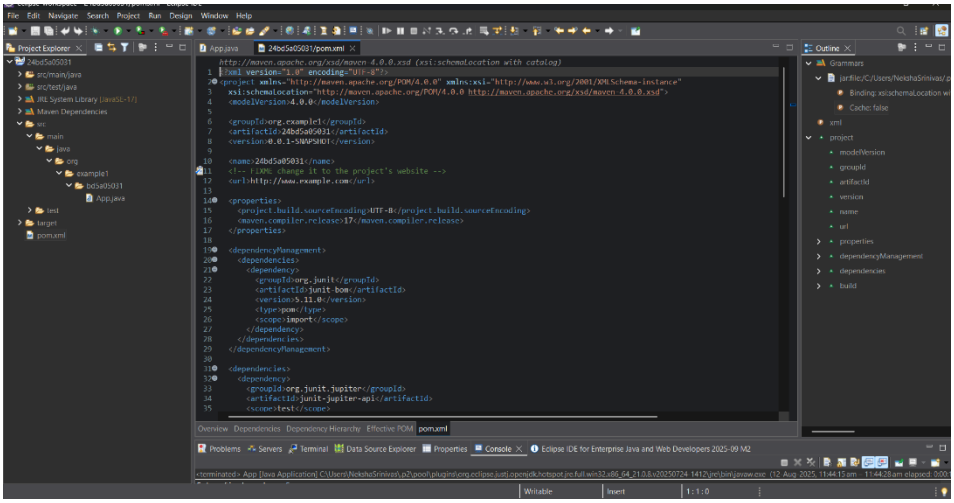
Git repo:

Git repo link: https://github.com/Edigirala-Neksha/se-lab-fourth



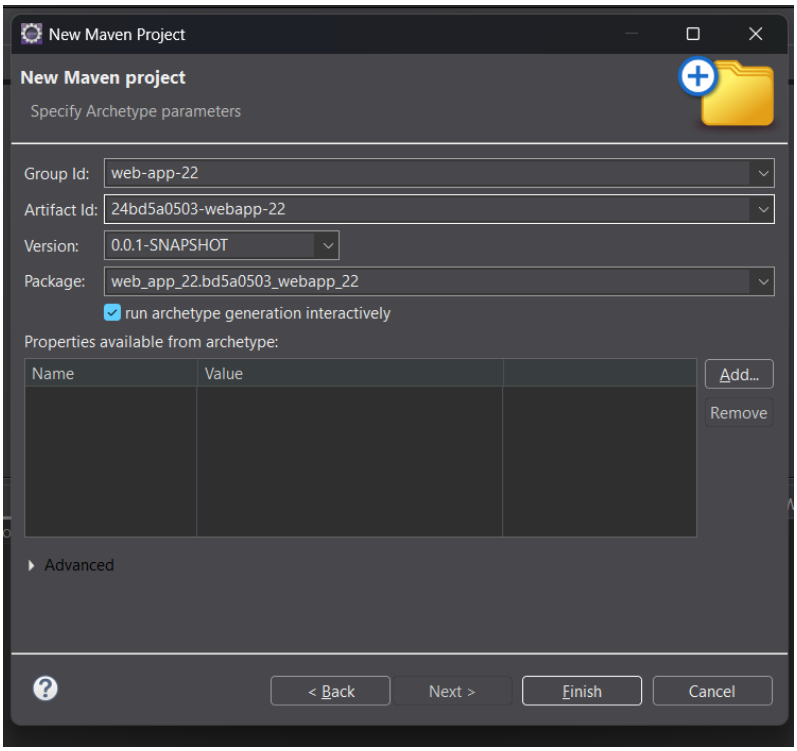
pom.xml file:

Shows the structure-



Creating maven-web project:

Step 1: Create a new maven project and give the details



Step 2: Click y to continue the creation of project

```
C:\Users\NekshaSrinivas\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_21.0.8.v20250724-1412\jre\bin\javaw.exe (02-Sept-2025, 7:19:56 pm) [pid: 13772]
Progress (1): 17/17 MB
Progress (1): 17/17 MB
Progress (1): 17/17 MB
Progress (1): 17/17 MB
Progress (1): 17/17 MB
Progress (1): 17 MB

Downloaded from central: https://repo.maven.apache.org/maven2/archetype-catalog.xml (17 MB at 9.1 MB/s)
[INFO] Archetype repository not defined. Using the one from [org.apache.maven.archetypes:maven-archetype-webapp:1.5] found in catalog remote
[INFO] Using property: groupId = web-app-22
[INFO] Using property: artifactId = 24bd5a0503-webapp-22
[INFO] Using property: version = 0.0.1-SNAPSHOT
[INFO] Using property: package = web_app_22.bd5a0503_webapp_22
Confirm properties configuration:
groupId: web-app-22
artifactId: 24bd5a0503-webapp-22
version: 0.0.1-SNAPSHOT
package: web_app_22.bd5a0503_webapp_22
 Y: y
```

Step 3: If the build is success it will show the message

```
package: web_app_22.bd5a0503_webapp_22
 Y: y
[INFO] ----------------------------------------------------------------------------
[INFO] Using following parameters for creating project from Old (1.x) Archetype: maven-archetype-webapp:1.0
[INFO] ----------------------------------------------------------------------------
[INFO] Parameter: basedir, Value: C:\Users\NekshaSrinivas\eclipse-workspace
[INFO] Parameter: package, Value: web_app_22.bd5a0503_webapp_22
[INFO] Parameter: groupId, Value: web-app-22
[INFO] Parameter: artifactId, Value: 24bd5a0503-webapp-22
[INFO] Parameter: packageName, Value: web_app_22.bd5a0503_webapp_22
[INFO] Parameter: version, Value: 0.0.1-SNAPSHOT
[INFO] project created from Old (1.x) Archetype in dir: C:\Users\NekshaSrinivas\eclipse-workspace\24bd5a0503-webapp-22
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  43.500 s
[INFO] Finished at: 2025-09-02T19:20:41+05:30
[INFO] ------------------------------------------------------------------------
```

Step 4: write the html code for the web page:

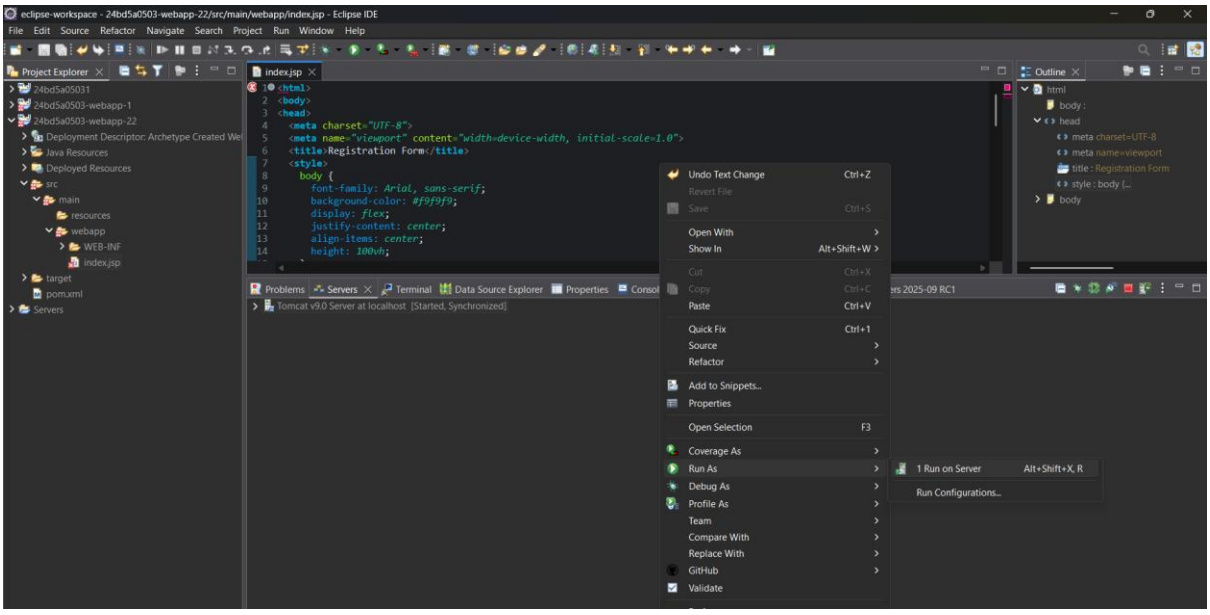
index.jsp

```
<html>
<body>
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Registration Form</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f9f9f9;
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
    }
    .form-container {
      background: #fff;
      padding: 20px 30px;
      border-radius: 10px;
      box-shadow: 0 4px 10px rgba(0,0,0,0.1);
      width: 300px;
    }
    .form-container h2 {
      text-align: center;
      margin-bottom: 20px;
    }
    .form-container input {
      width: 100%;
      padding: 10px;
      margin: 8px 0;
      border: 1px solid #ccc;
      border-radius: 5px;
    }
    .form-container button {
      width: 100%;
      padding: 10px;
      background: #4CAF50;
```

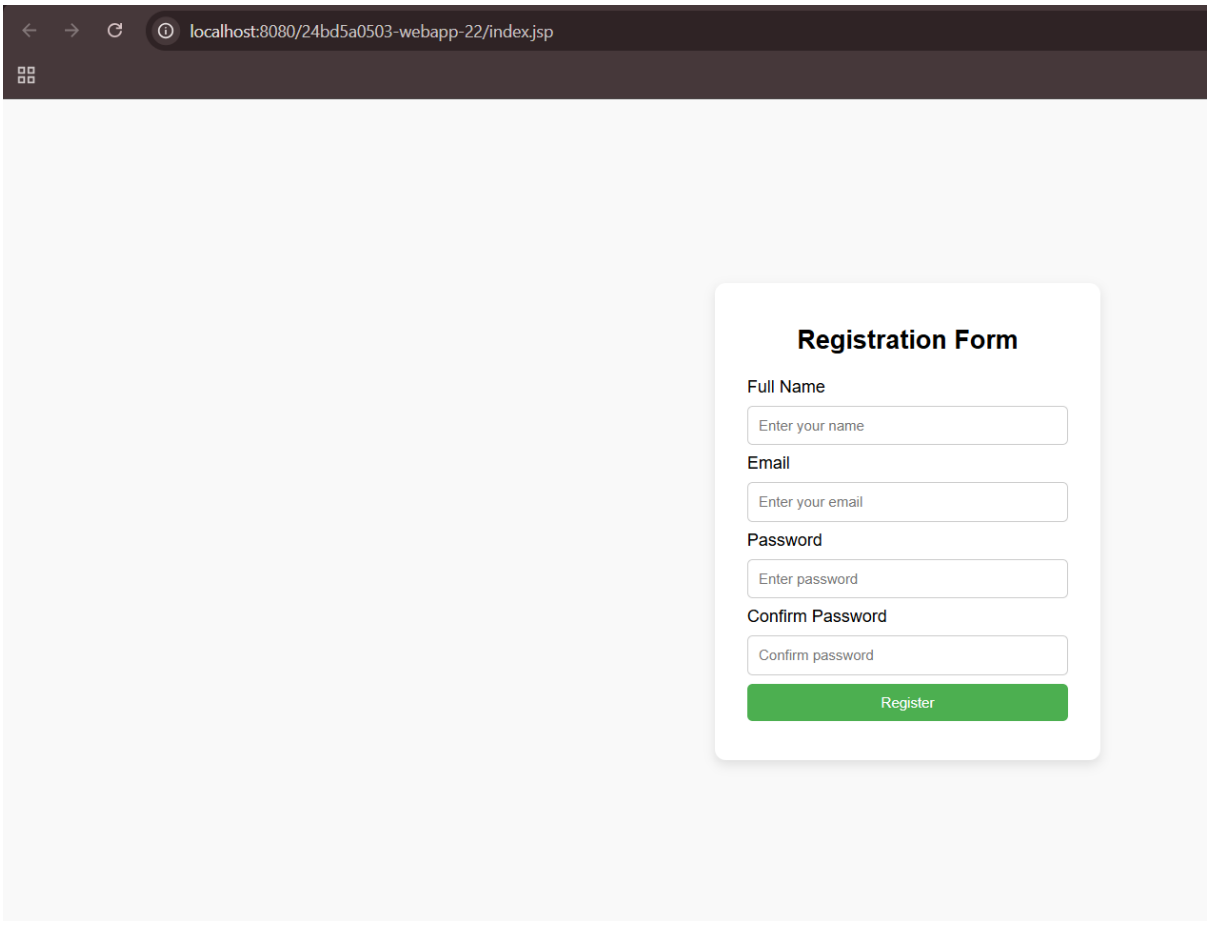
index.jsp

```
      border-radius: 5px;
    }
    .form-container button {
      width: 100%;
      padding: 10px;
      background: #4CAF50;
      color: white;
      border: none;
      border-radius: 5px;
      cursor: pointer;
    }
    .form-container button:hover {
      background: #45a049;
    }
  </style>
</head>
<body>
  <div class="form-container">
    <h2>Registration Form</h2>
    <form action="#" method="post">
      <label for="fullname">Full Name</label>
      <input type="text" id="fullname" name="fullname" placeholder="Enter your name" required>

      <label for="email">Email</label>
      <input type="email" id="email" name="email" placeholder="Enter your email" required>

      <label for="password">Password</label>
      <input type="password" id="password" name="password" placeholder="Enter password" required>

      <label for="confirm">Confirm Password</label>
      <input type="password" id="confirm" name="confirm" placeholder="Confirm password" required>

      <button type="submit">Register</button>
    </form>
  </div>
</body>
</html>

```

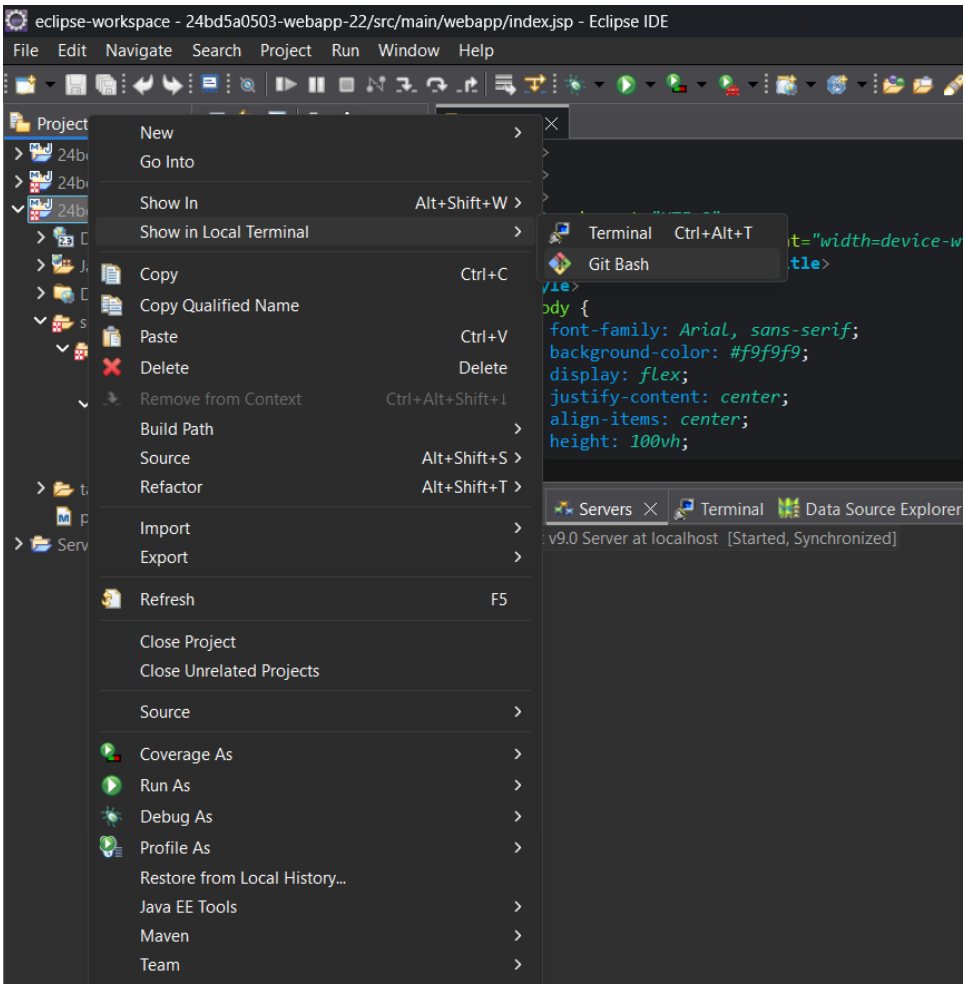
Step 5: Select run on server



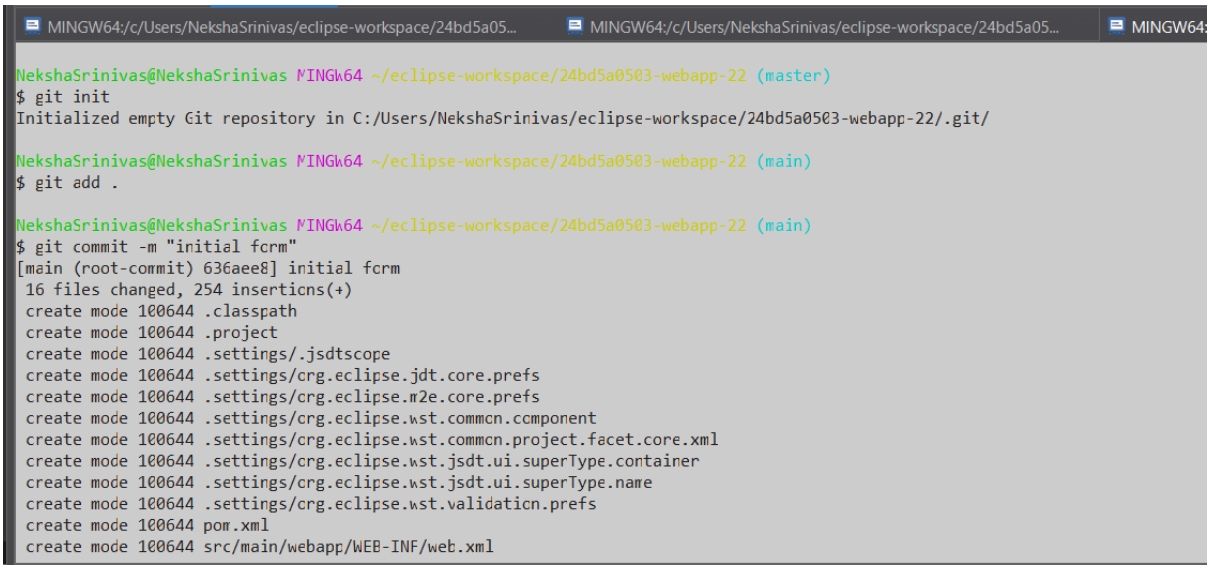
Step 6: It will show the following output:



Step 7: To push it into git, select git bash from show in local terminal

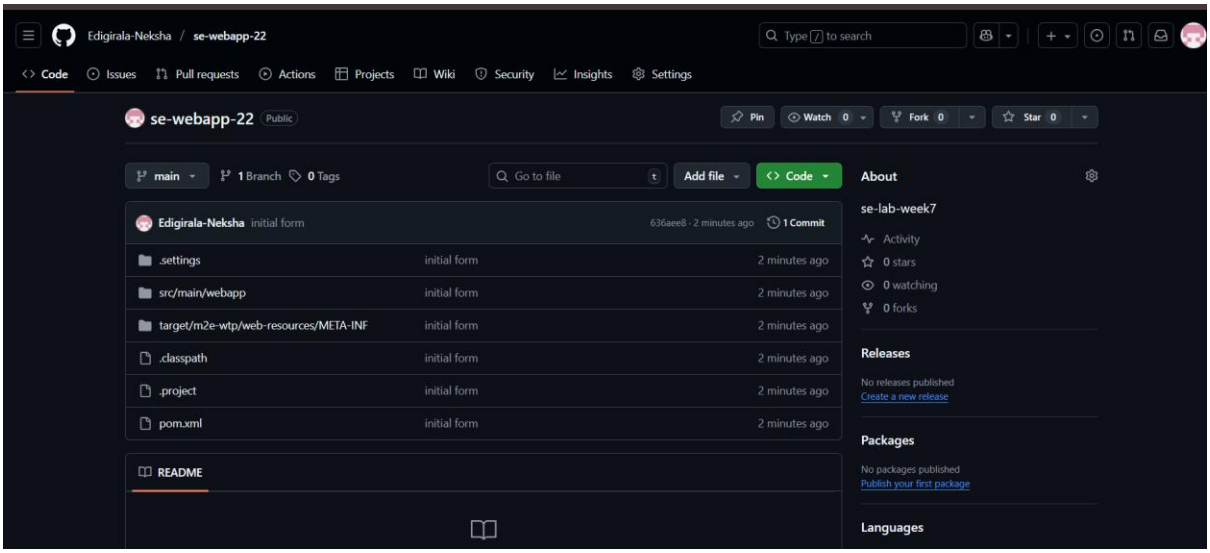


Step 8: use the command of git to push the maven web project



```
NekshaSrinivas@NekshaSrinivas MINGW64 ~/eclipse-workspace/24bd5a0503-webapp-22 (main)
$ git branch
* main

NekshaSrinivas@NekshaSrinivas MINGW64 ~/eclipse-workspace/24bd5a0503-webapp-22 (main)
$ git push origin main
Enumerating objects: 29, done.
Counting objects: 100% (29/29), done.
Delta compression using up to 12 threads
Compressing objects: 100% (18/18), done.
Writing objects: 100% (29/29), 4.43 KiB | 283.00 KiB/s, done.
Total 29 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/Edigirala-Neksha/se-webapp-22.git
 * [new branch]      main -> main

NekshaSrinivas@NekshaSrinivas MINGW64 ~/eclipse-workspace/24bd5a0503-webapp-22 (main)
$
```

Step 9: verify the repo in git hub

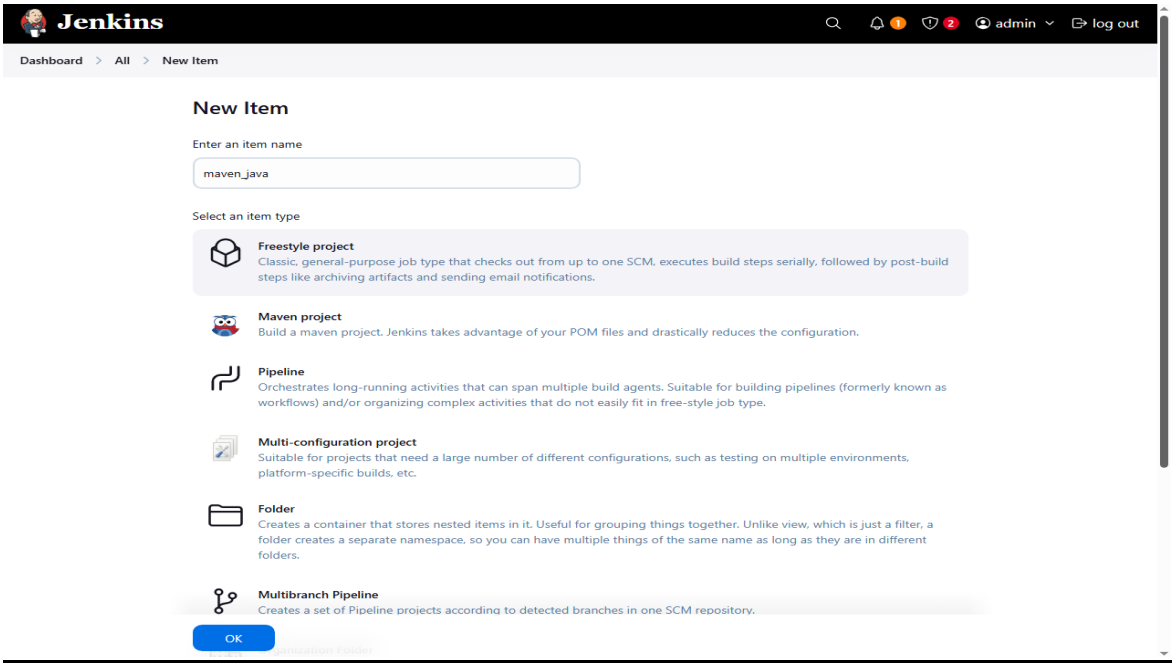


8. Jenkins Automation

Steps for MavenJava Automation

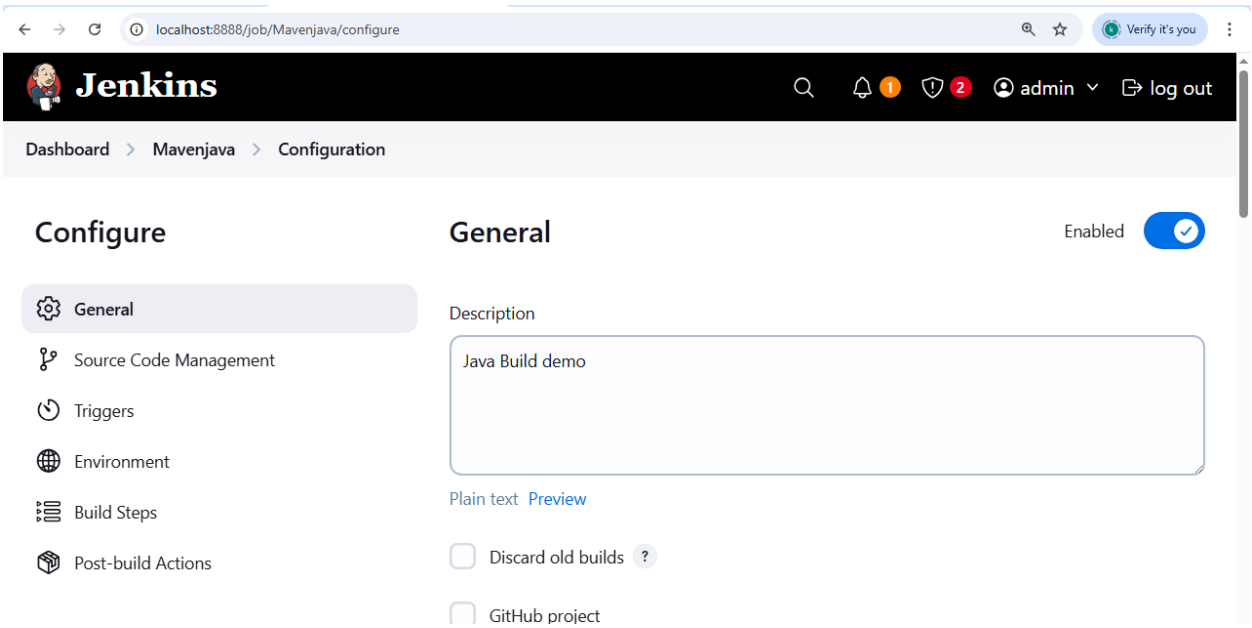
Step 1: Open Jenkins (localhost:8888)

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

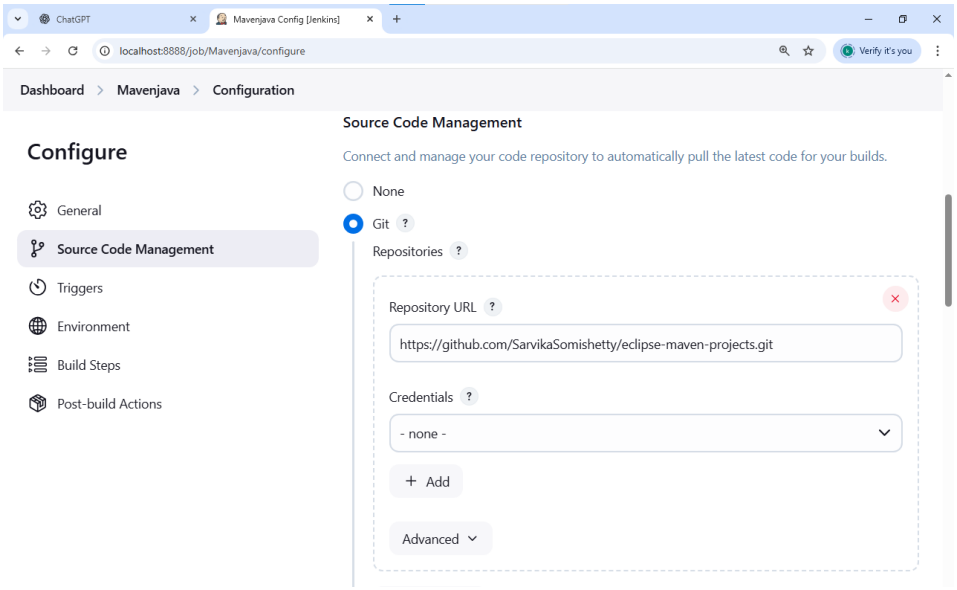


Step 2: Configuration of maven_java project

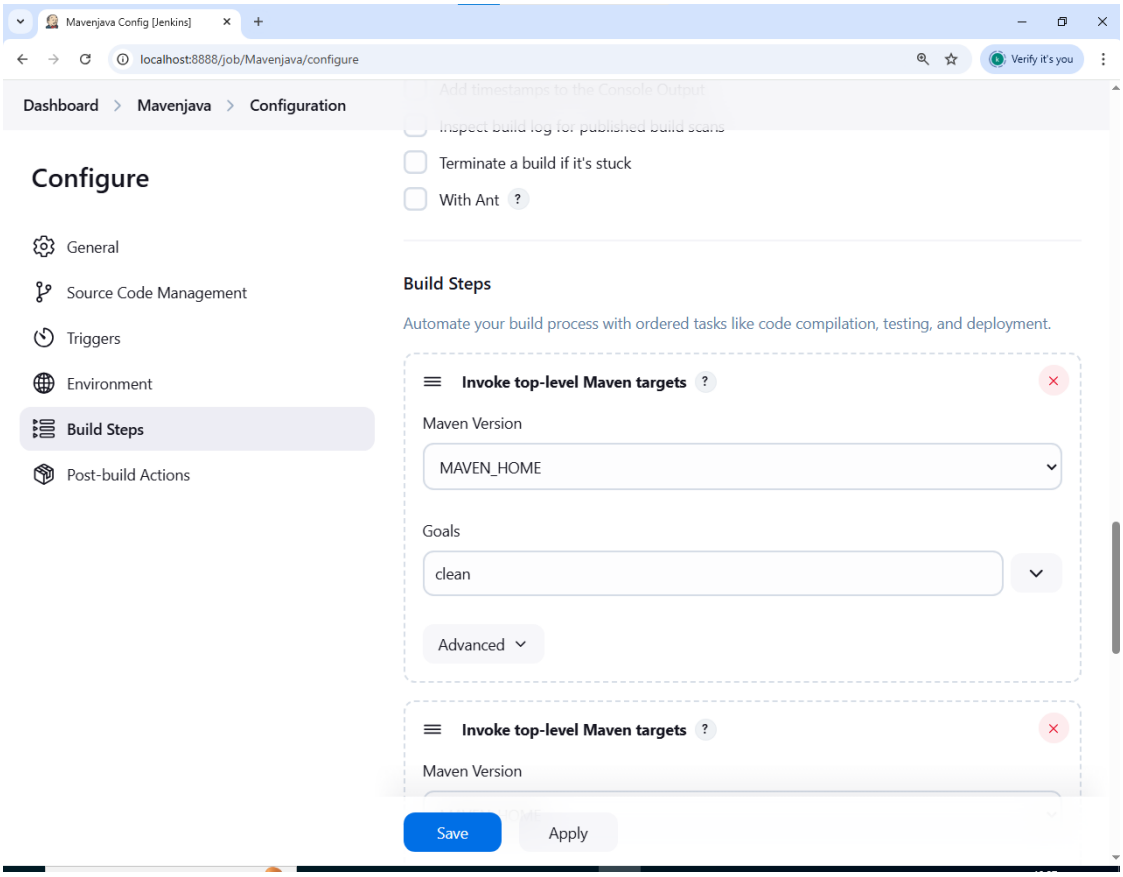
Give the description



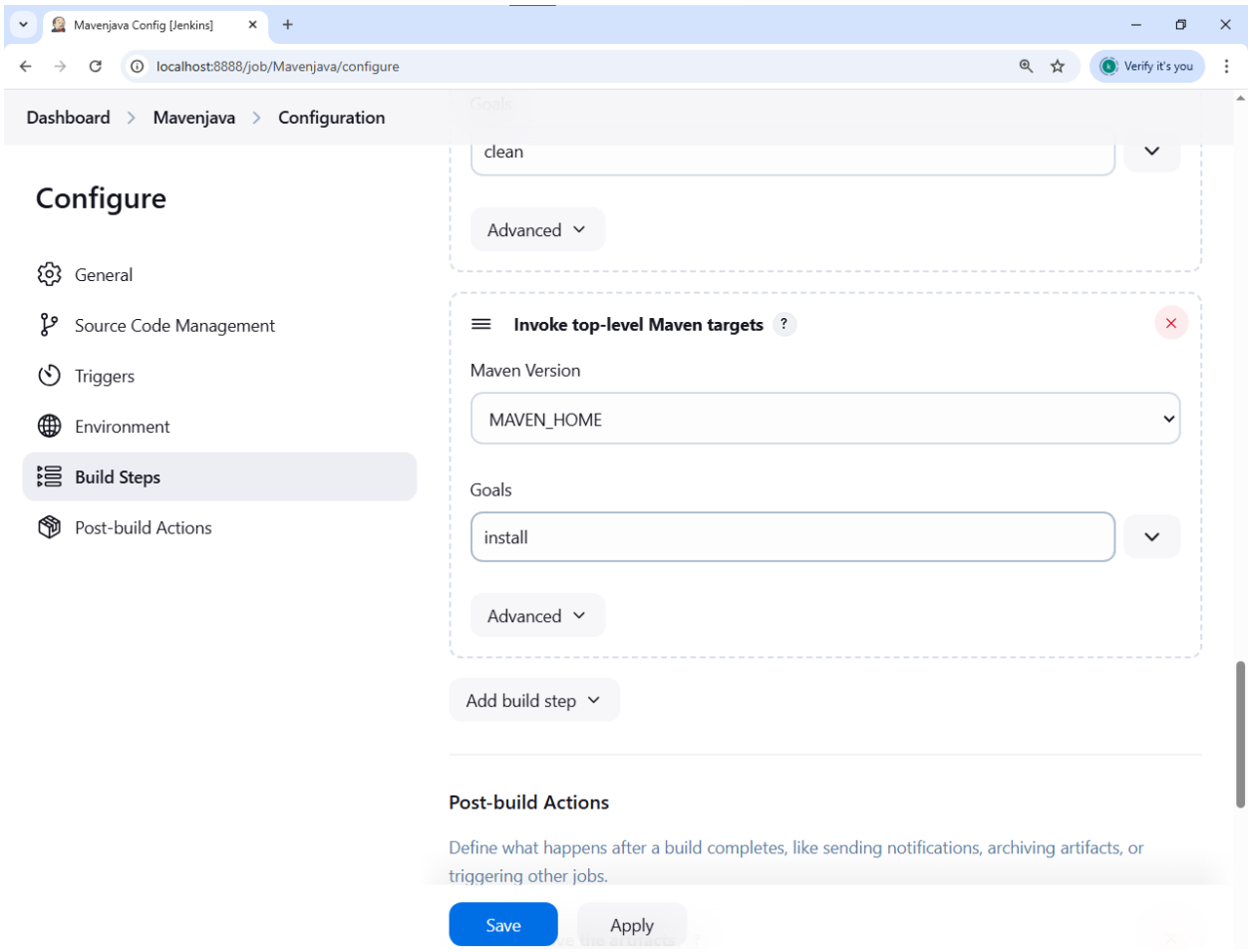
In the source code management select git and give the git repo link



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

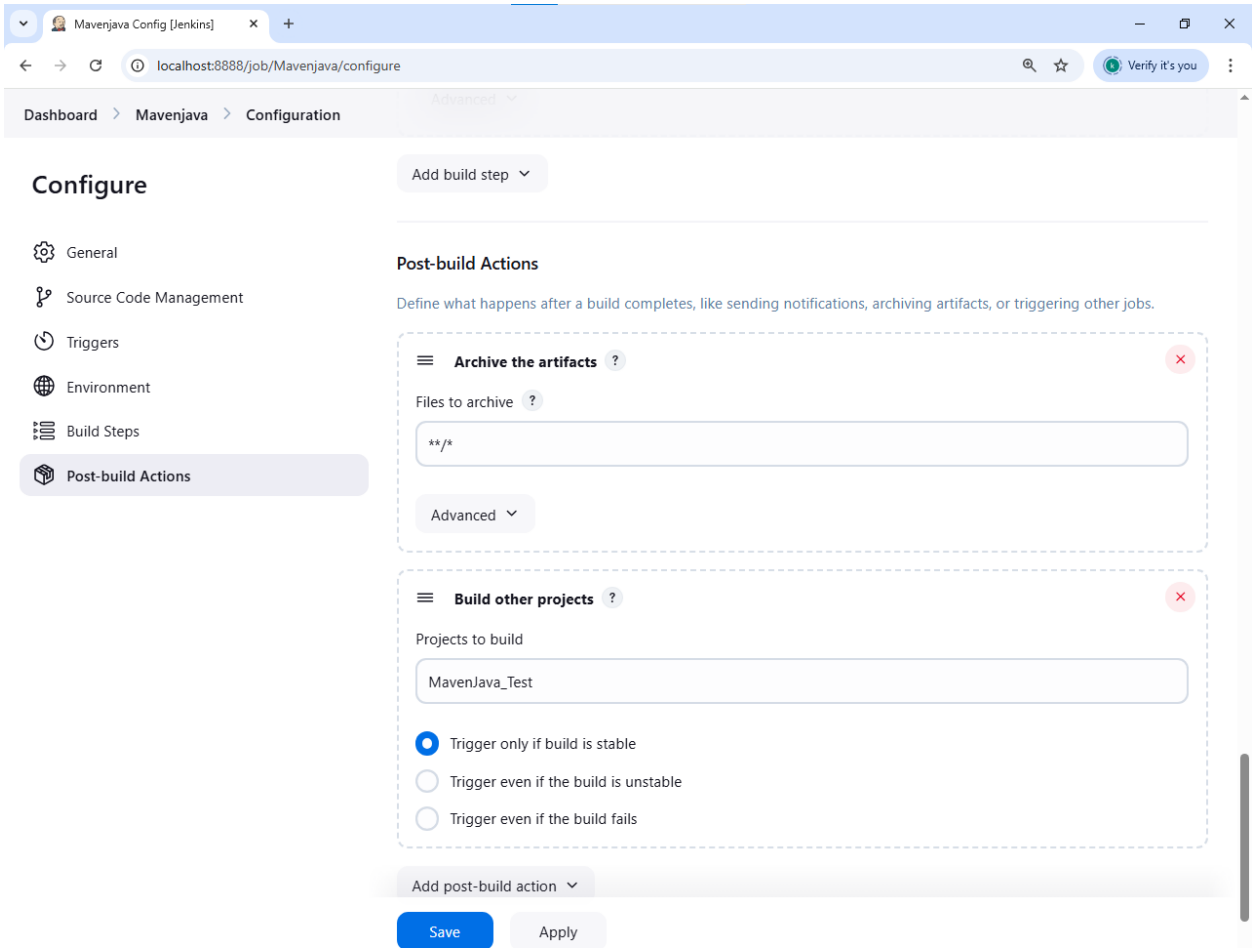


In the post build actions > click on add post build action > select the archive the artifacts > in the file to archive give “**/*”

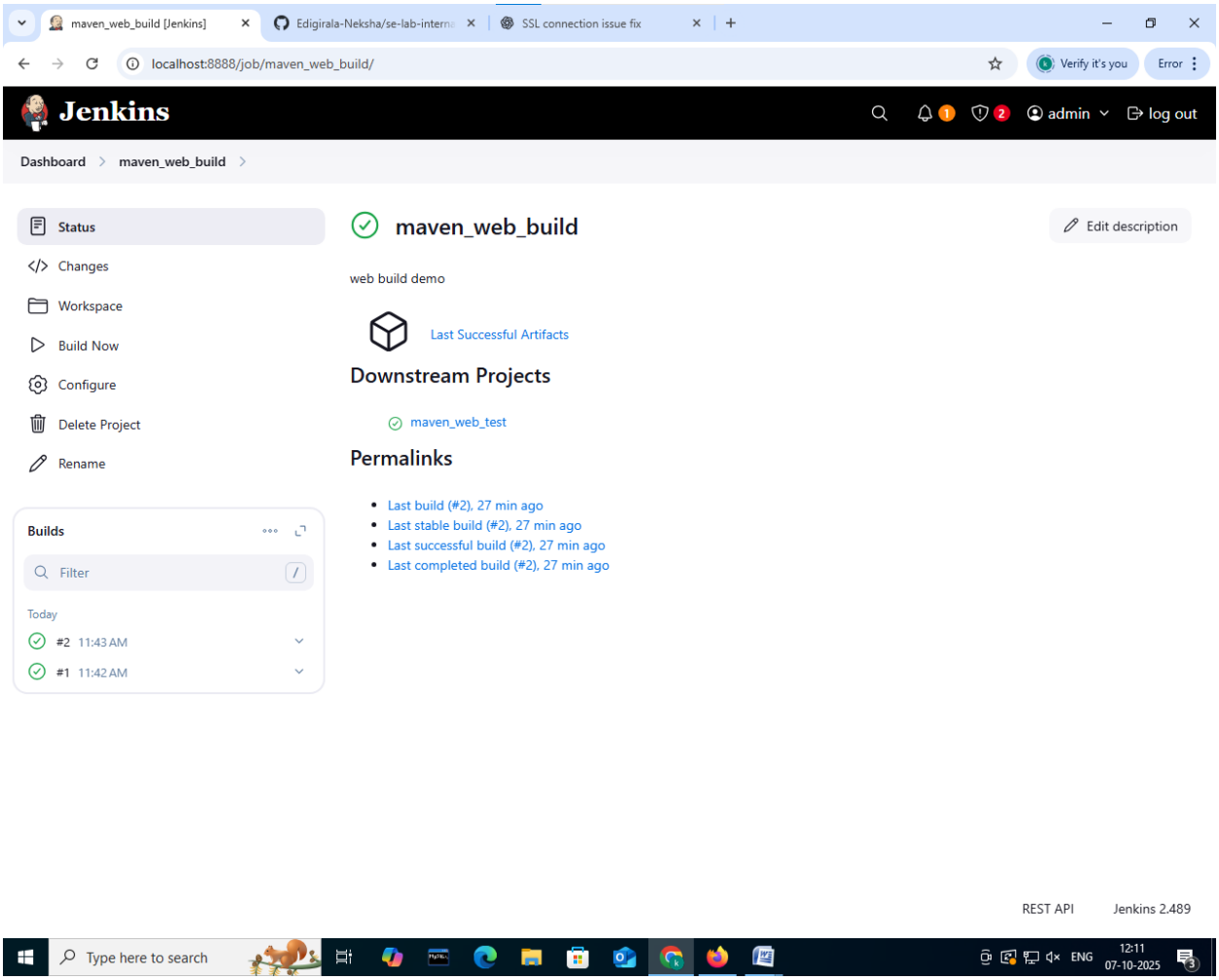
For the second post build action,

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

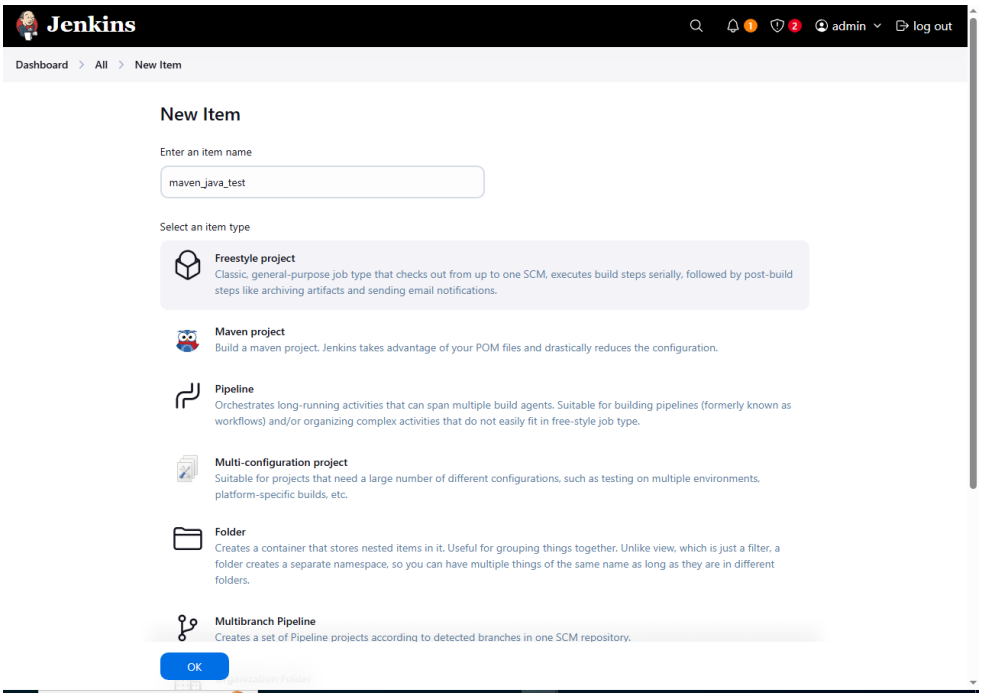


If the build is success:



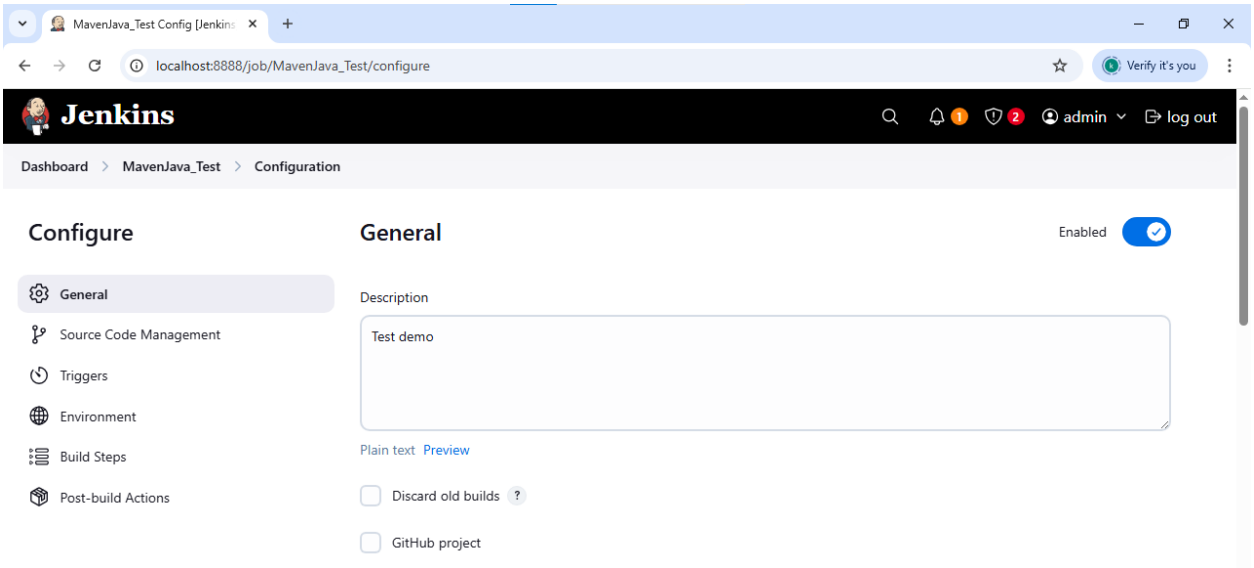
Step 3: Create Freestyle Project (e.g., MavenJava_Test)

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

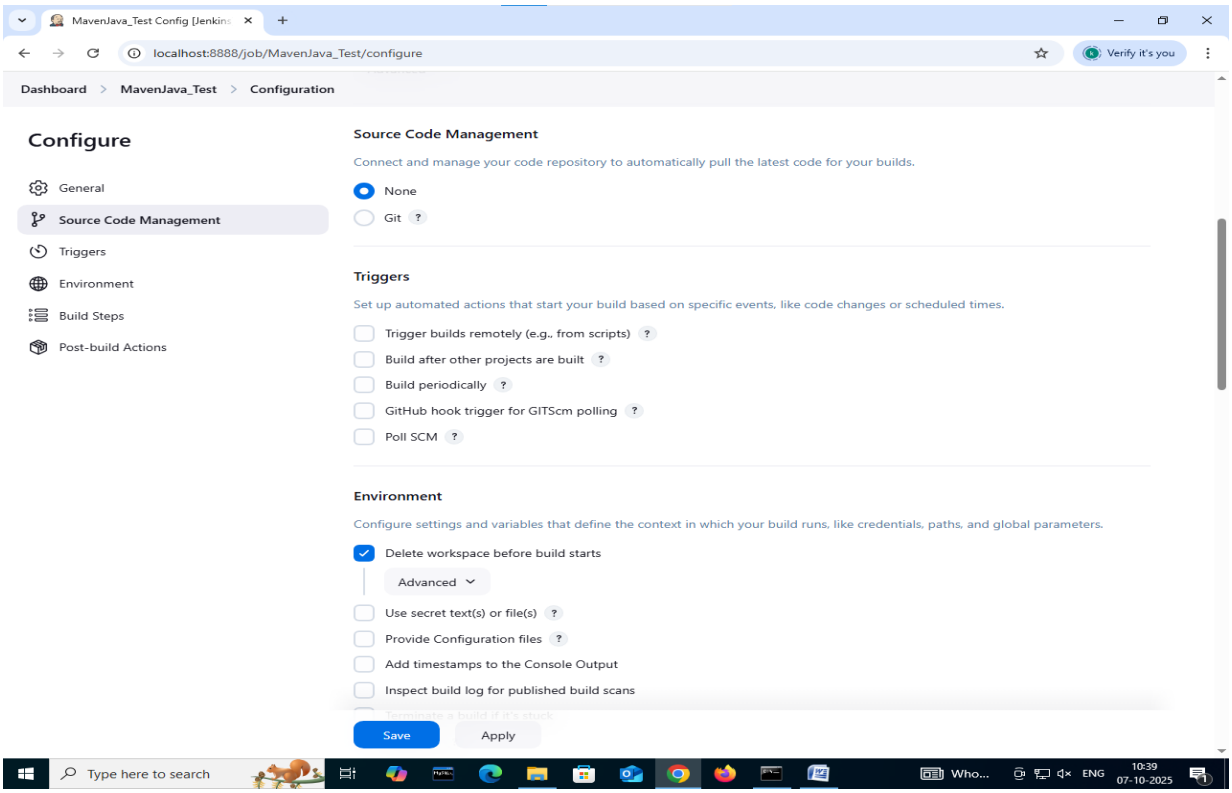


Step 4: Configuration of maven_java project

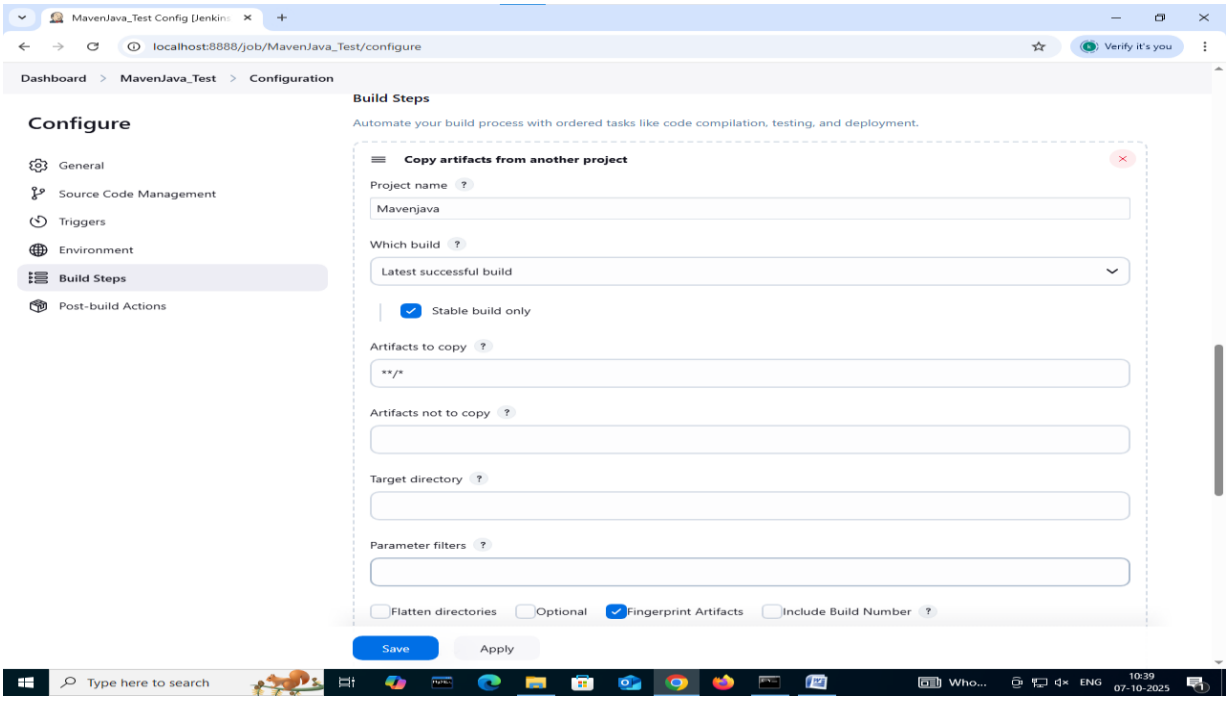
Give the description



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

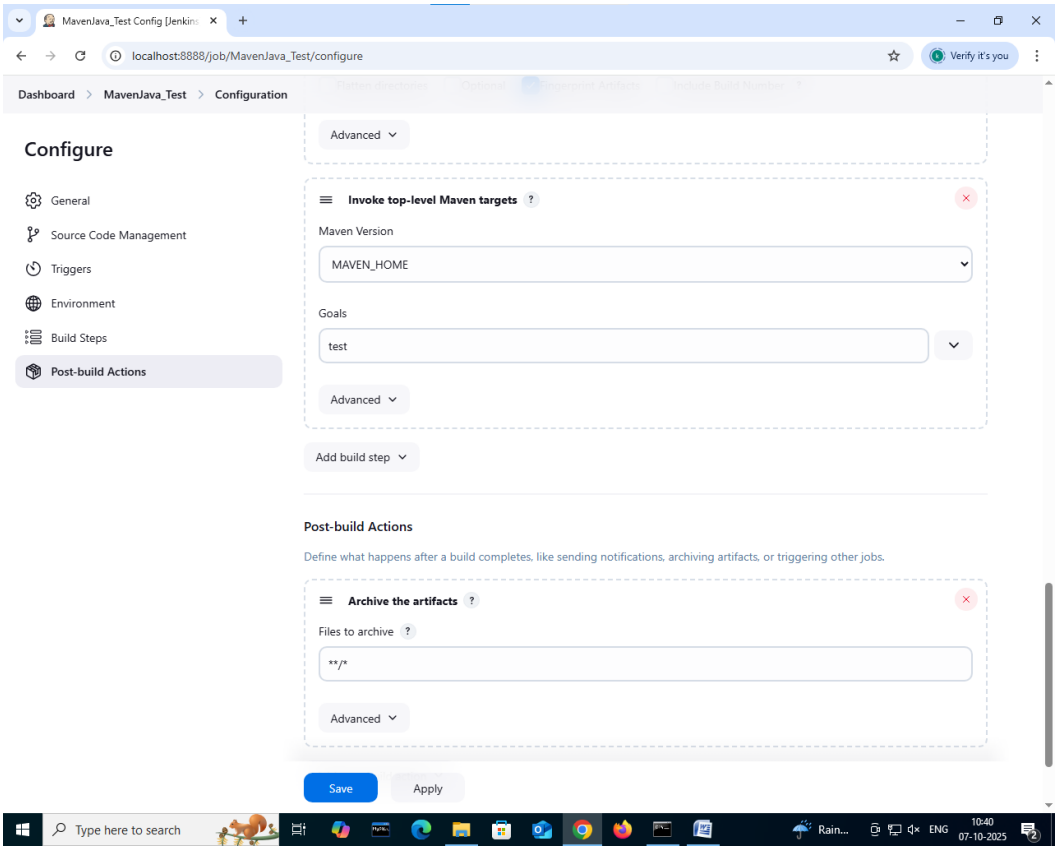


In the build steps> select add a build step> select “copy artifacts from another project” > give project name as Maven java and artifacts to copy as **/*

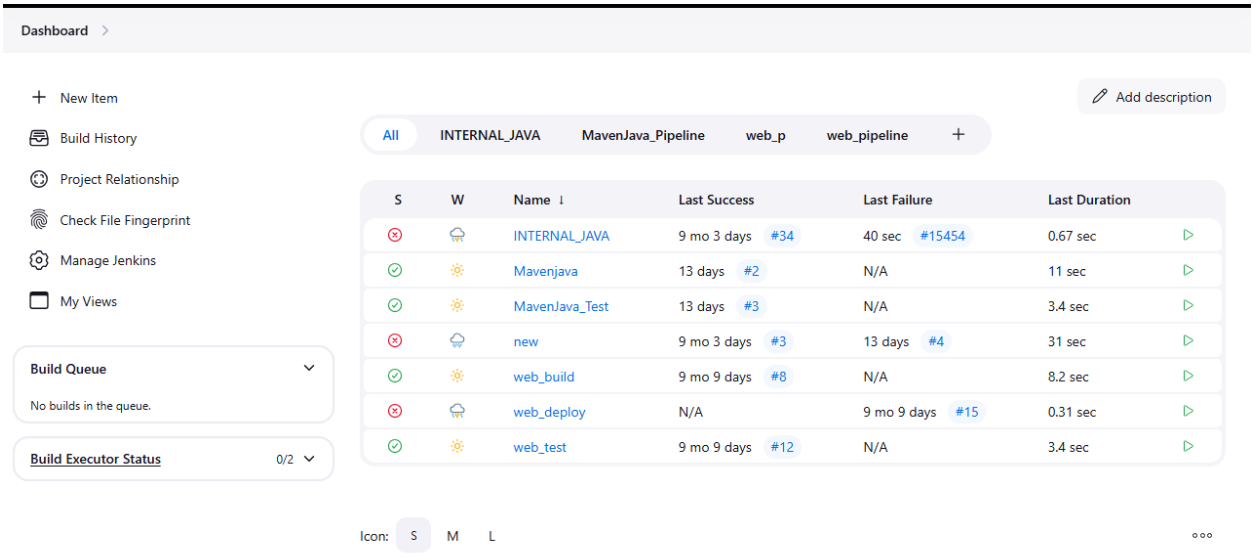


In the post build actions> select archive the artifacts and enter files as **/*

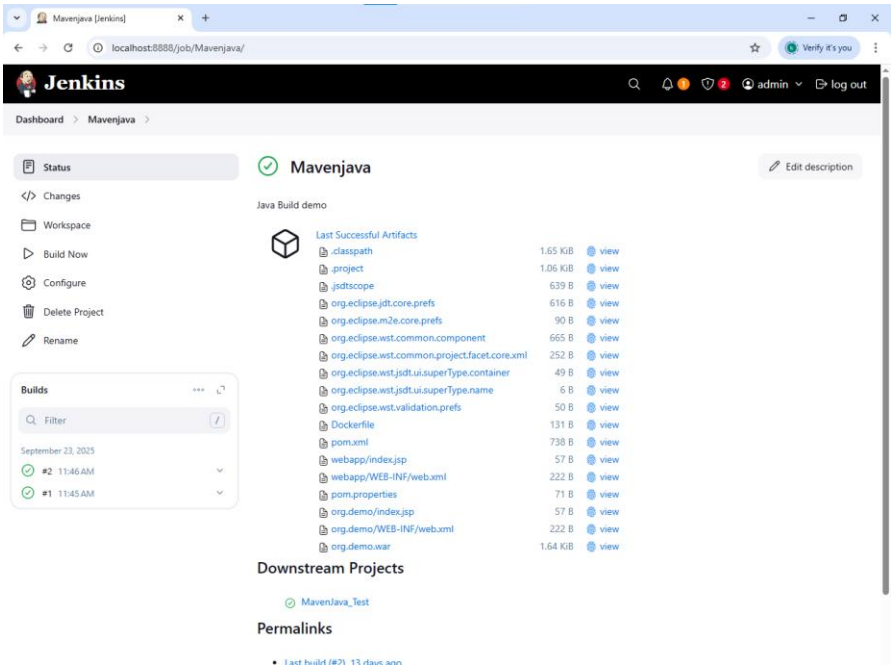
Click on apply and save



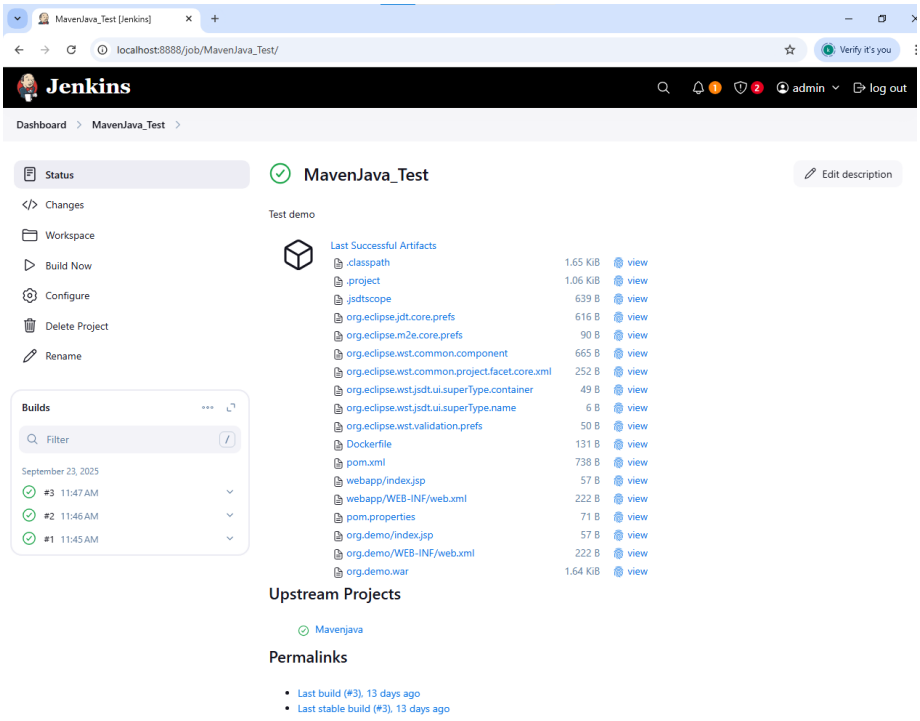
In the dashboard you will find MavenJava and MavenJava_Test



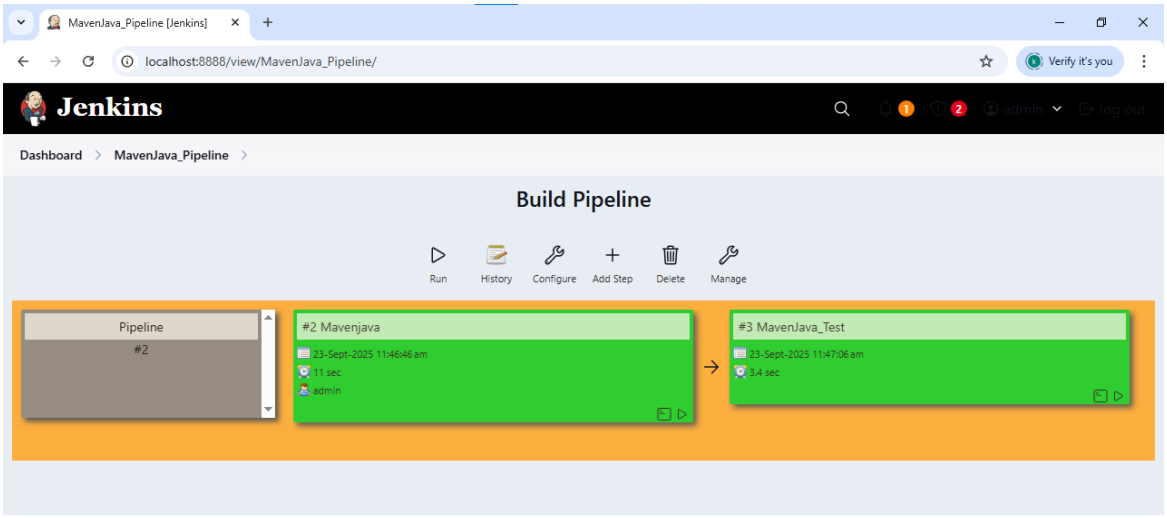
If you open the MavenJava file the following will be shown in case on no errors



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



MavenJava_pipeline

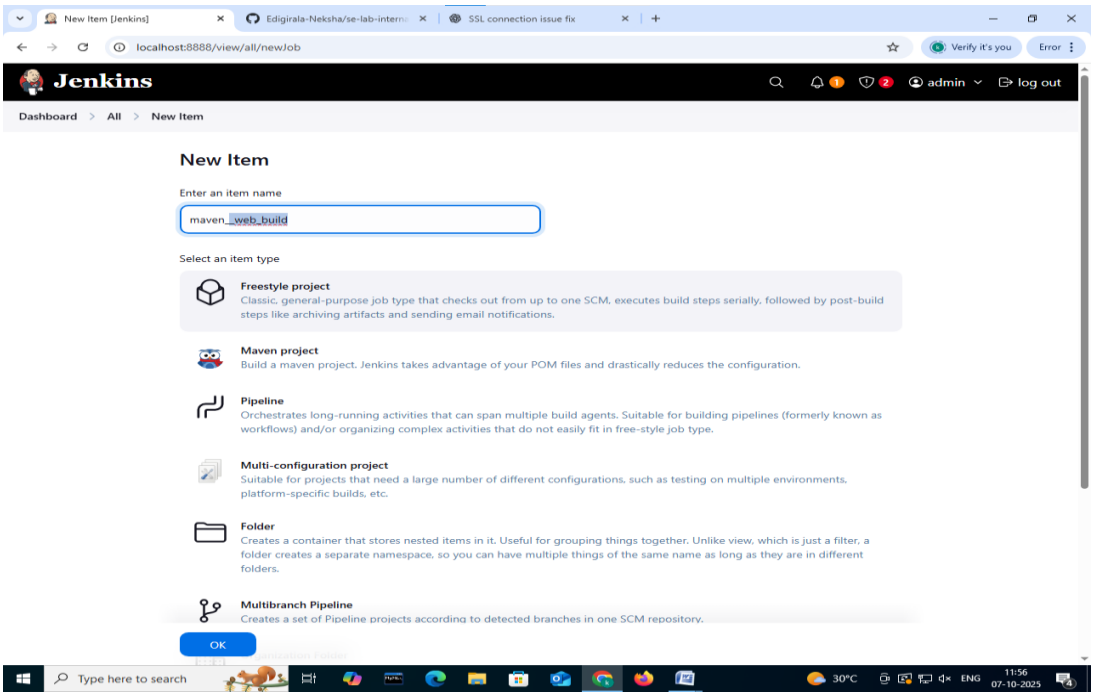


II. Maven Web Automation Steps:

Create Freestyle Project (e.g., MavenWeb_Build)

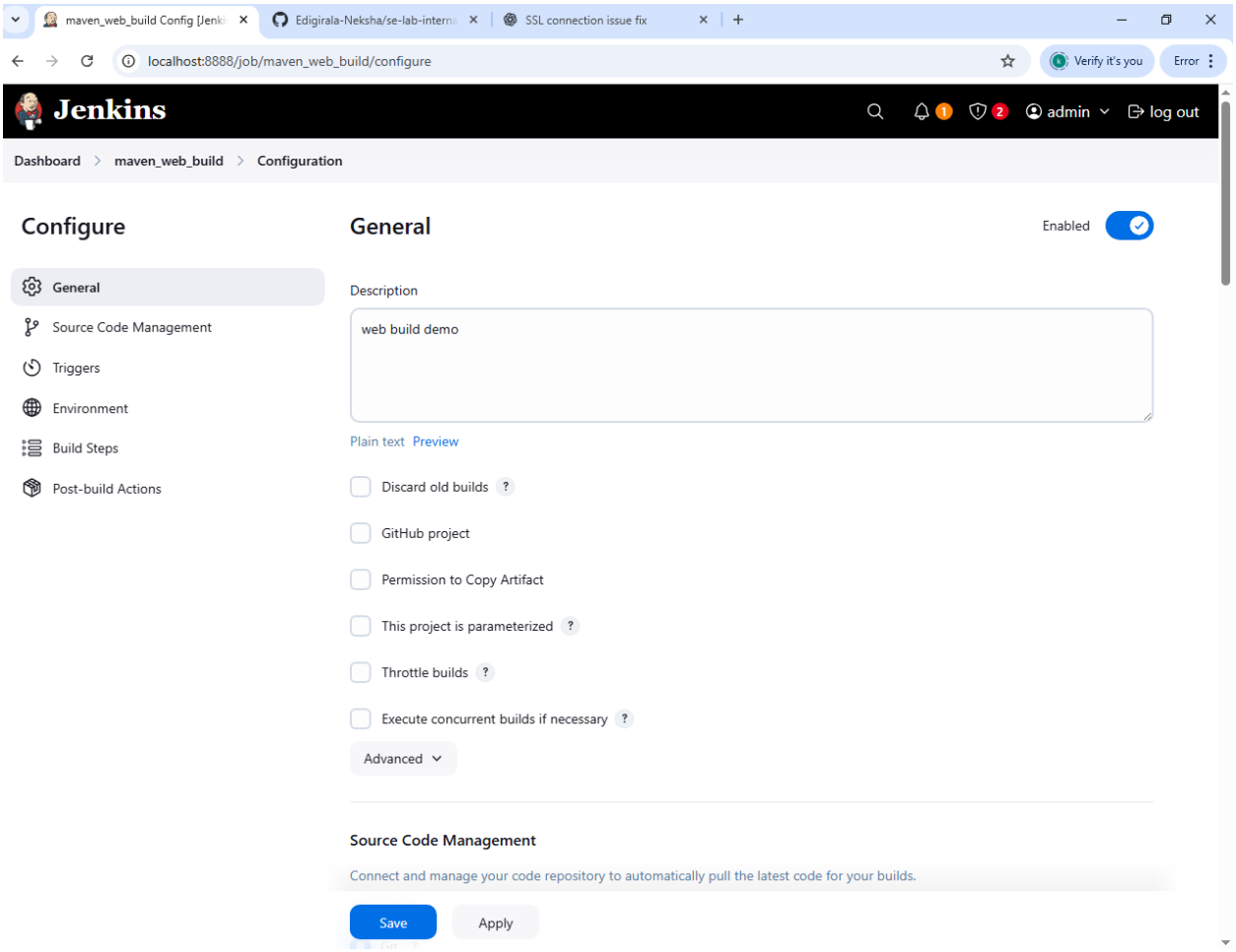
Step 1: Open Jenkins (localhost:8888)

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

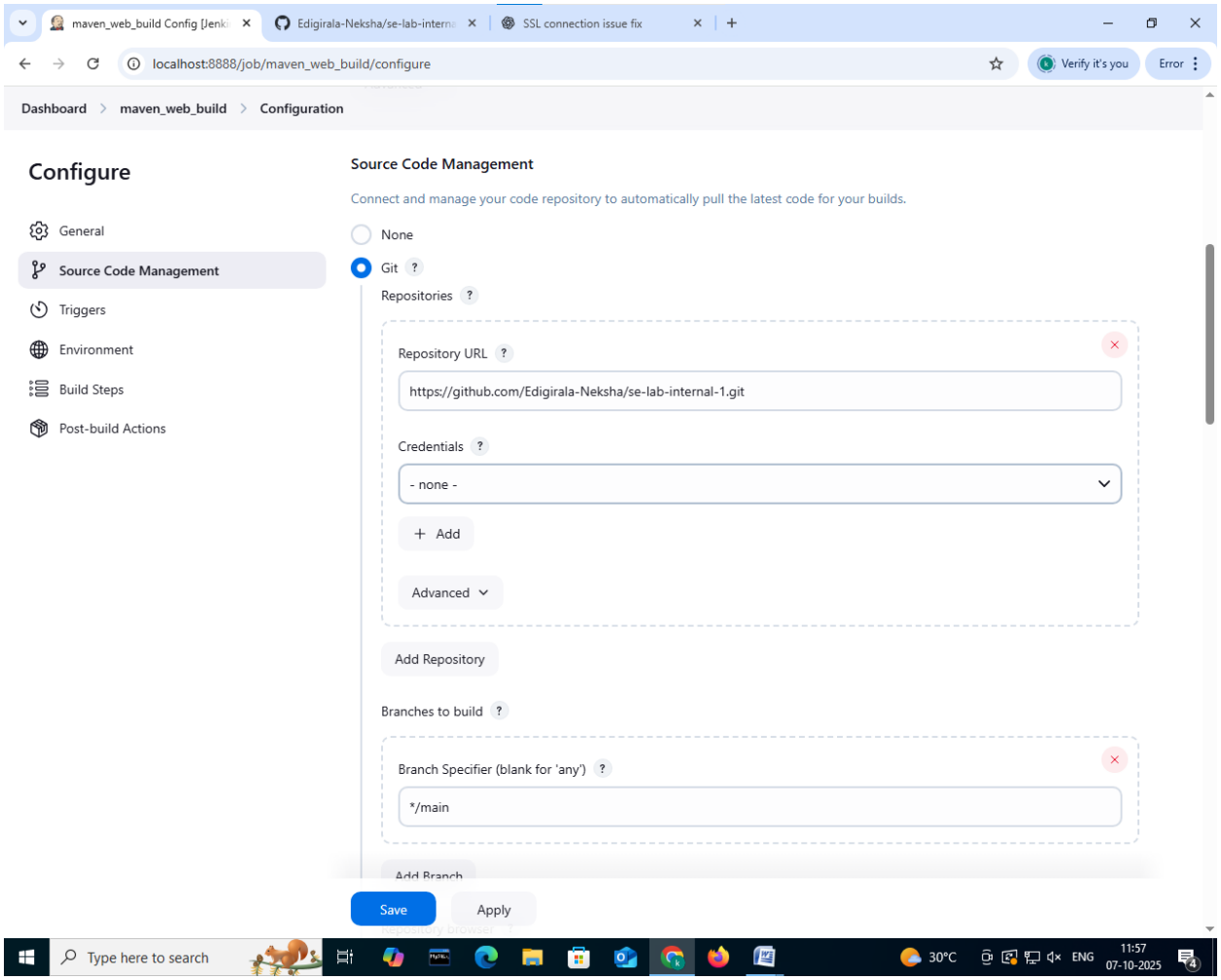


Step 2: Configuration of maven_web_build project

Give the description



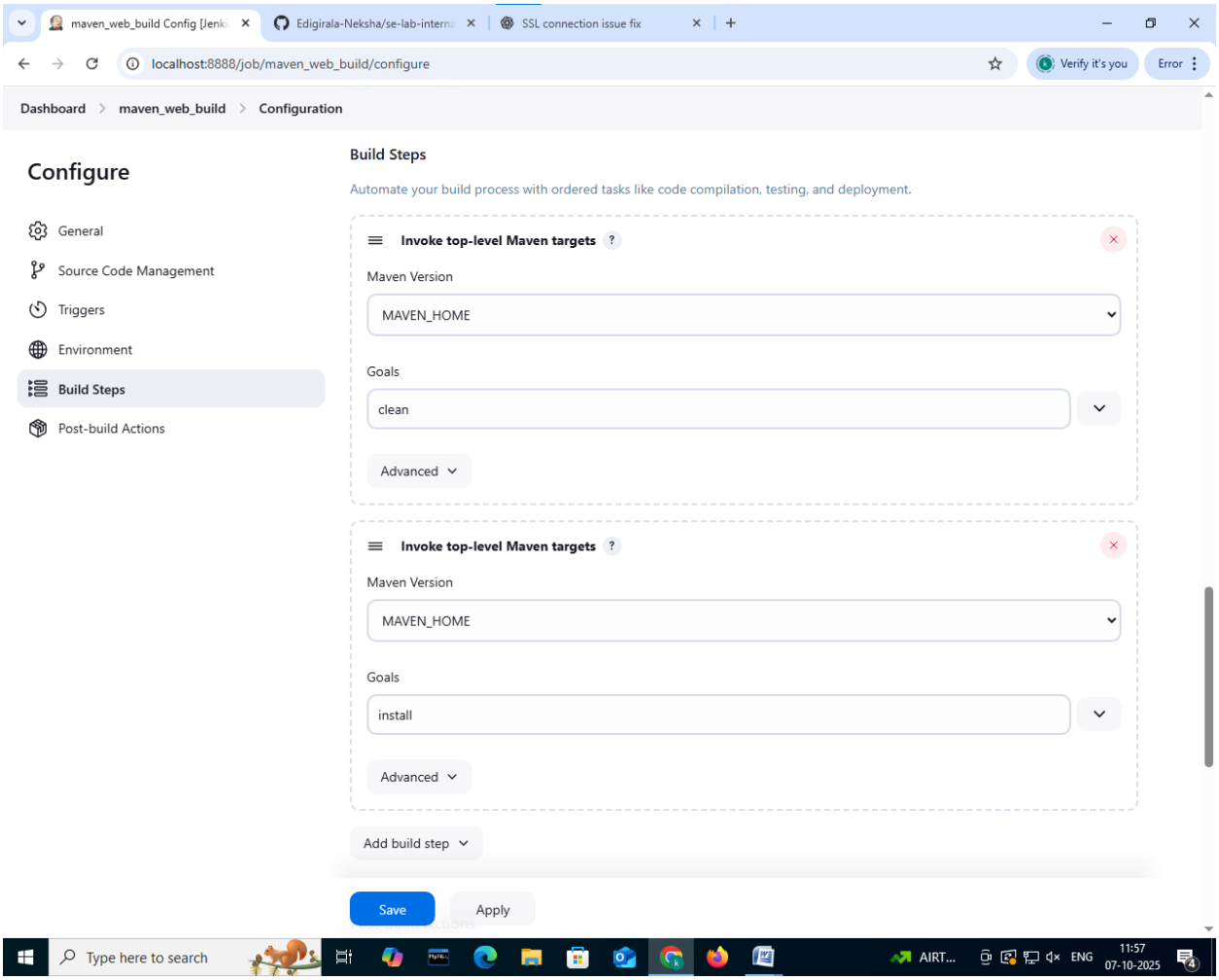
In the source code management select git and give the git repo link



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

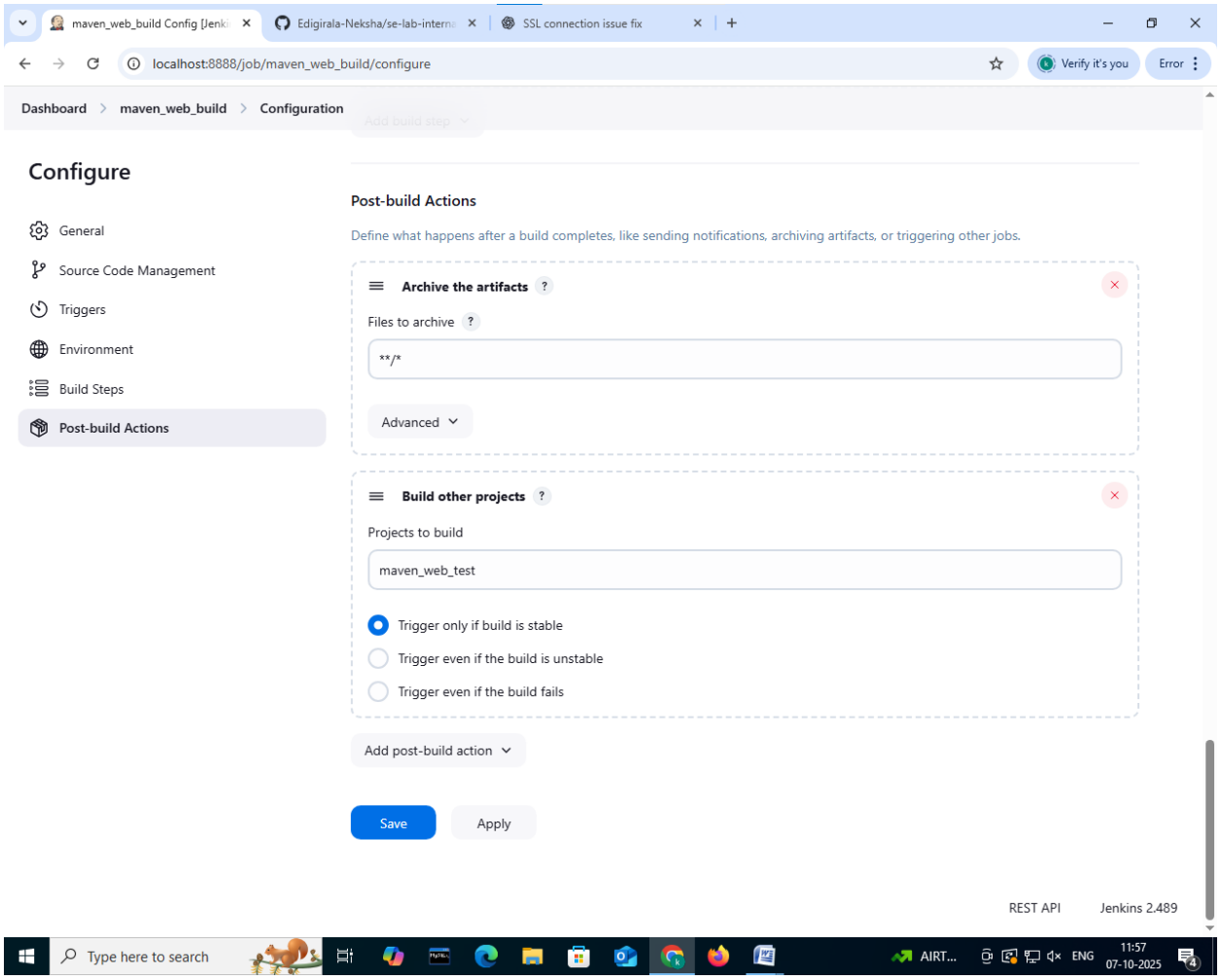


In the post build actions > click on add post build action > select the archive the artifacts > in the file to archive give “**/*”

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

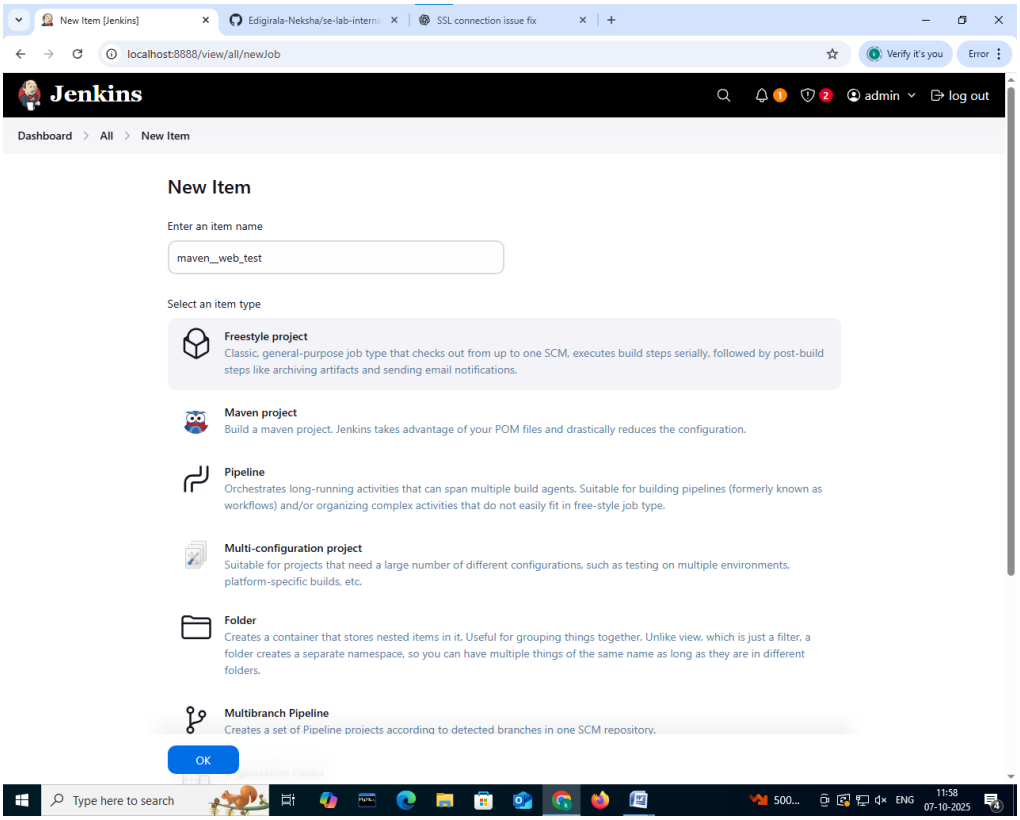
Click on apply and save



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

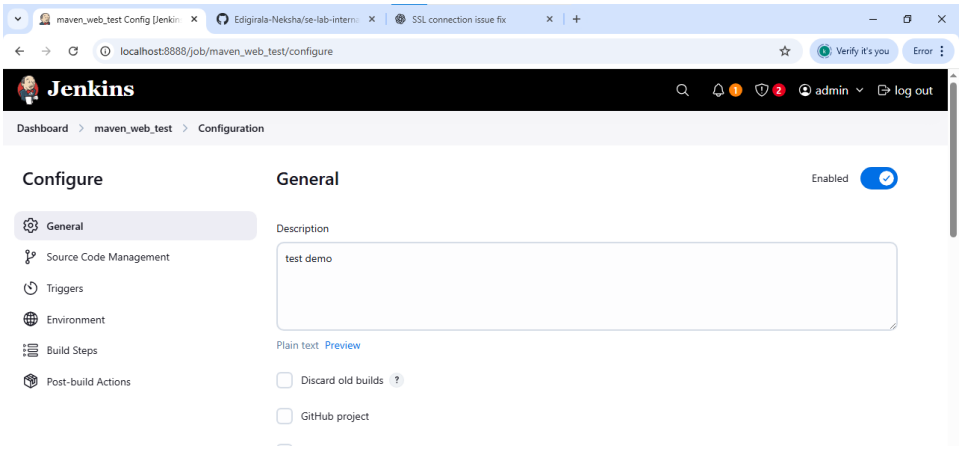
Step 1: Open Jenkins (localhost:8888)

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

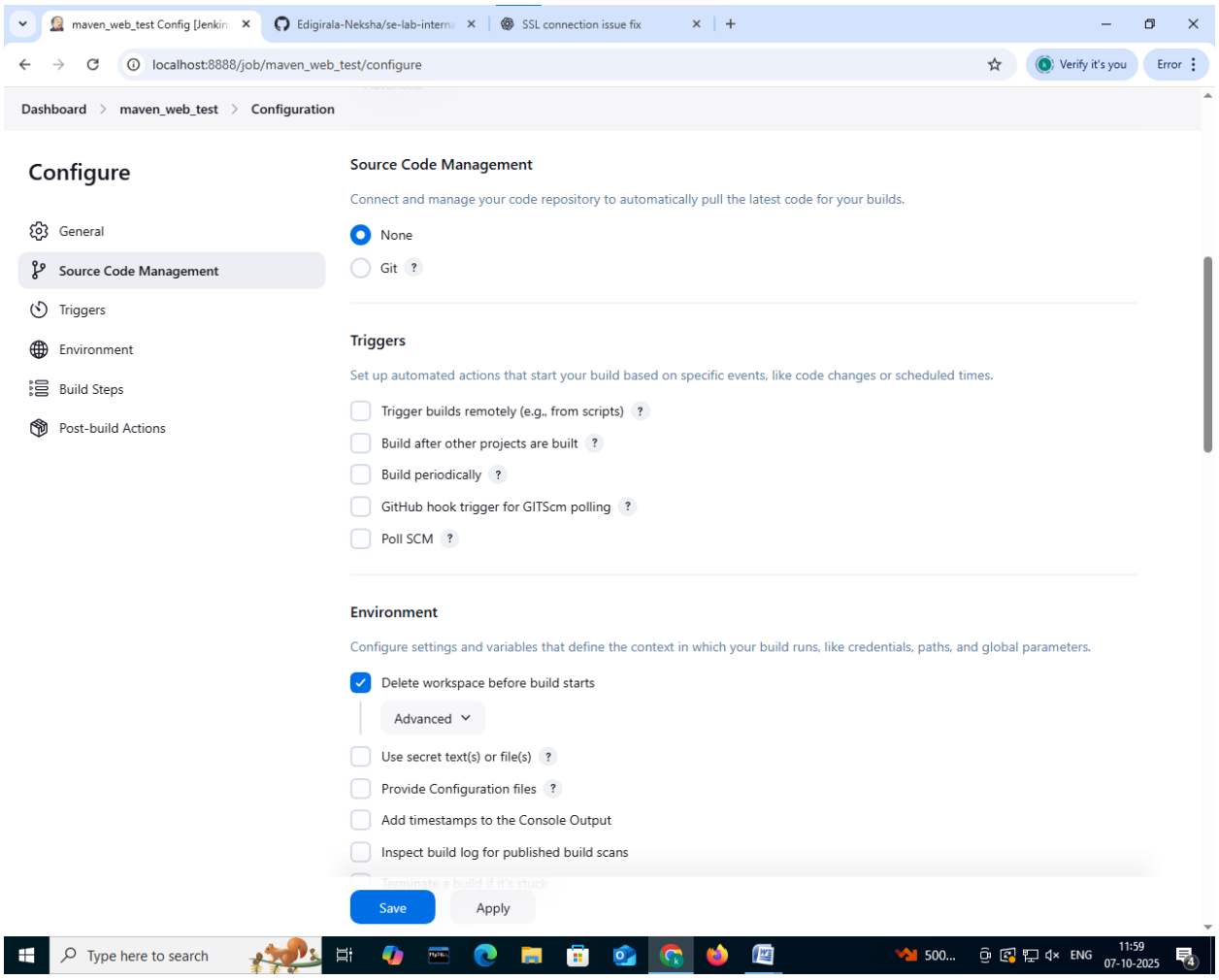


Step 2: Configuration of maven_web_test project

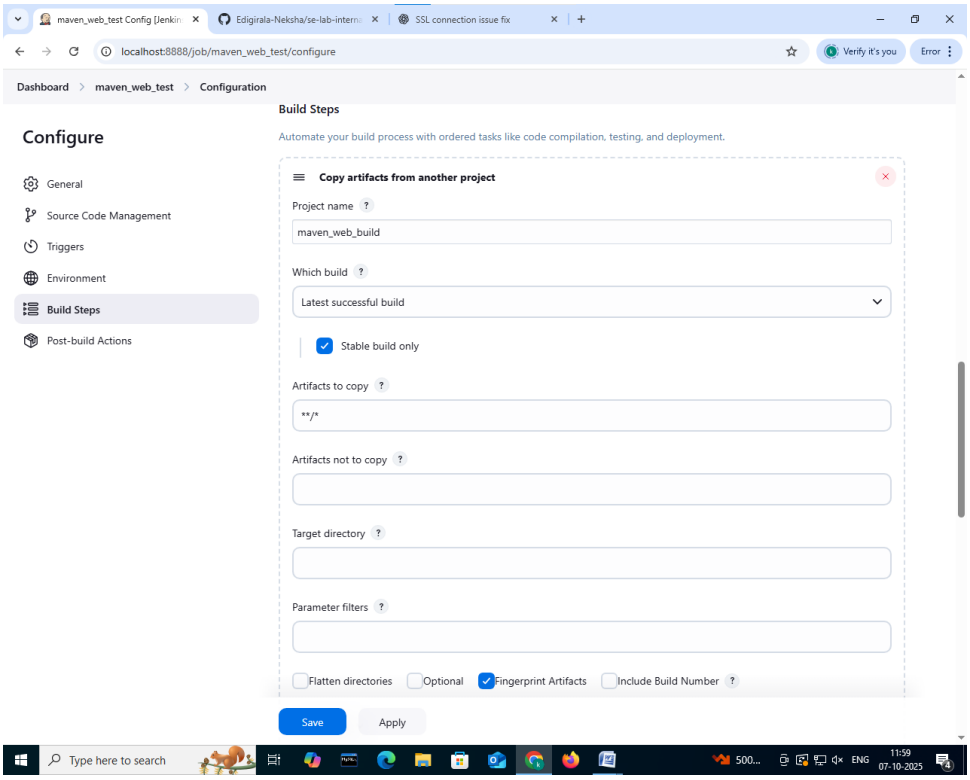
Give the description



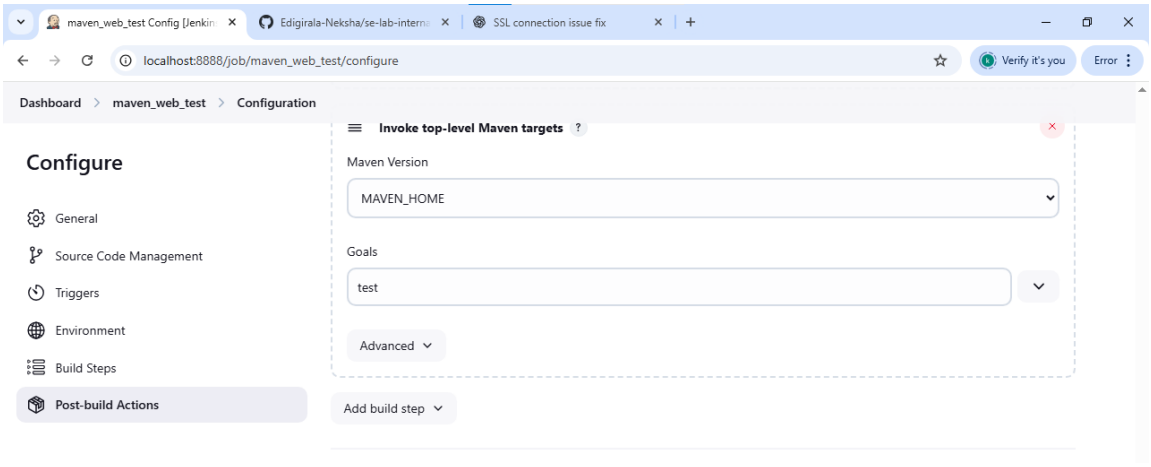
In the source code management select none and in environment select “delete workspace before build starts”



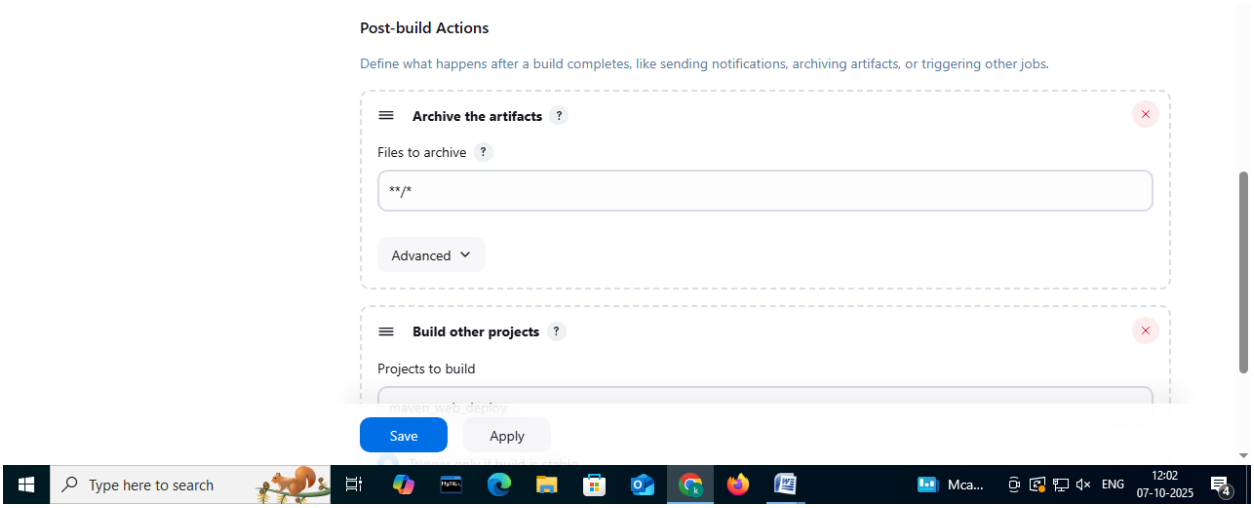
In the build steps click on add build step > select copy artifacts from another project > give project name as maven_web_build > give artifacts to copy as **/*



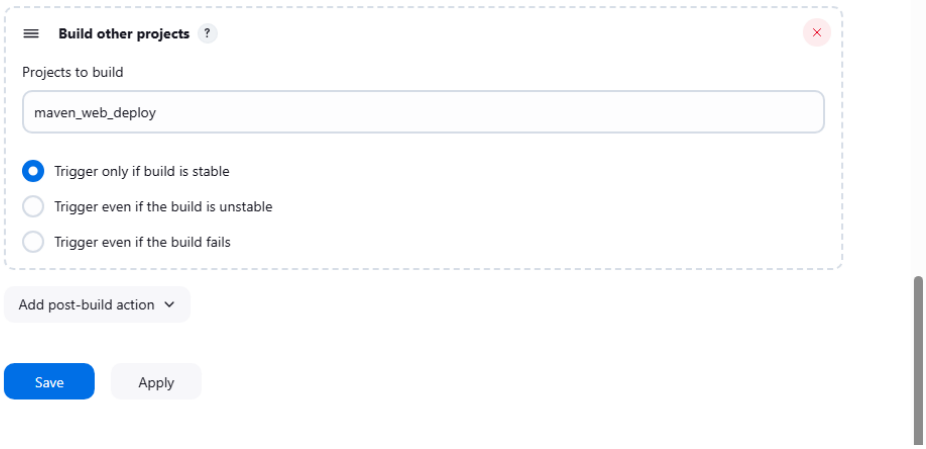
In the build steps click on add build step > give maven version as MAVEN_HOME > select invoke top-level maven targets > goals as test



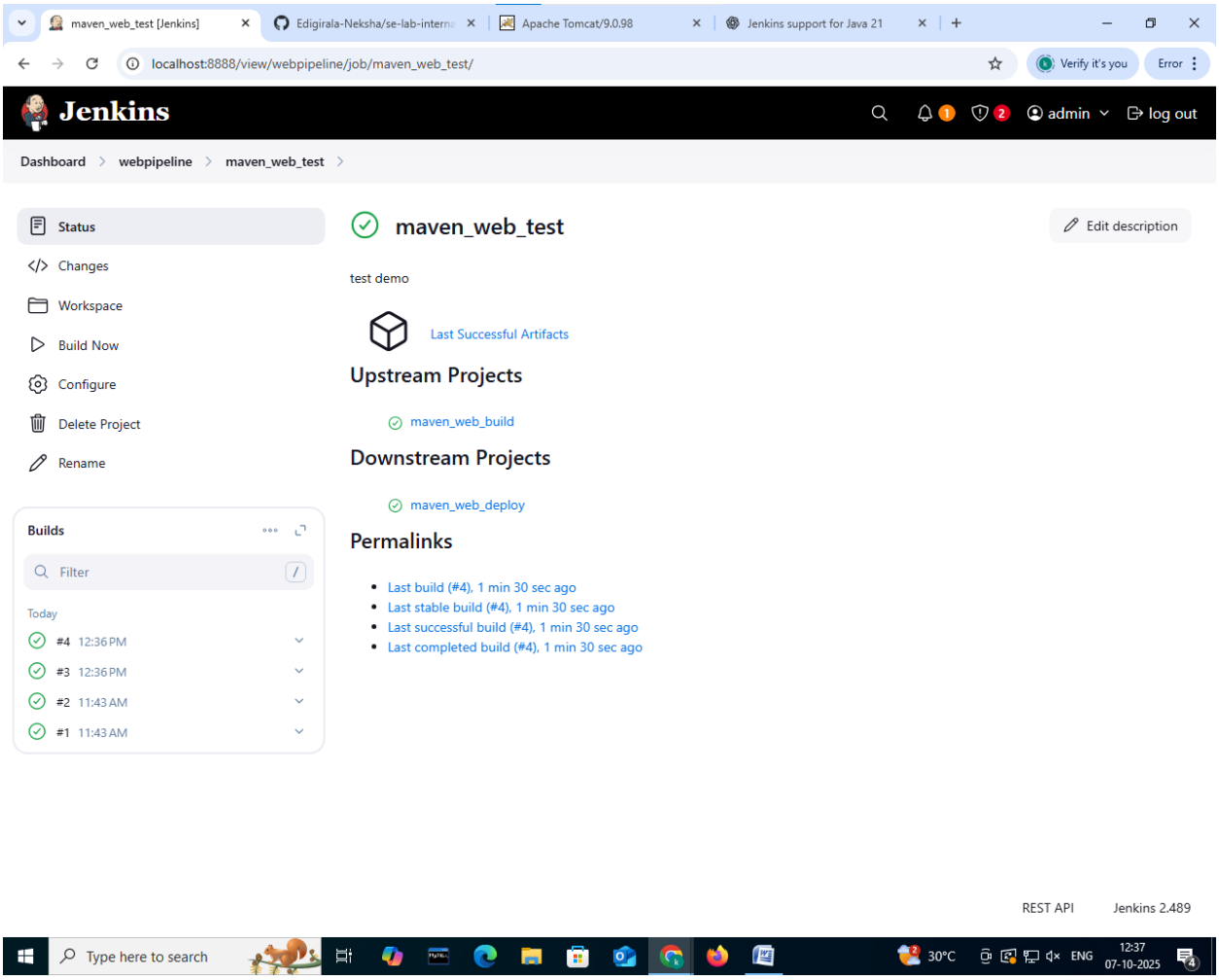
In the post build actions > click on add post build action > select the archive the artifacts > in the file to archive give **/*



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”



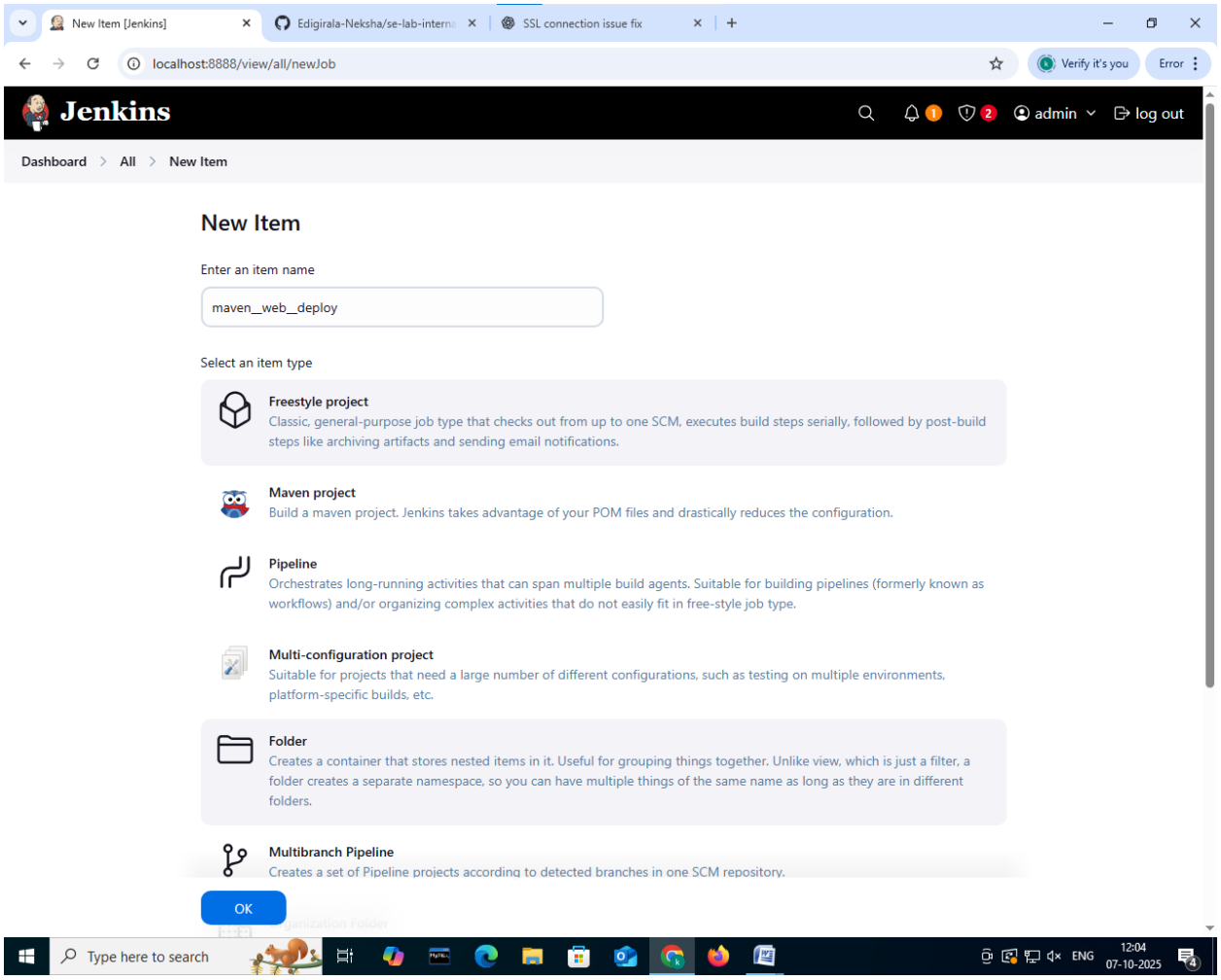
If the build is success:



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

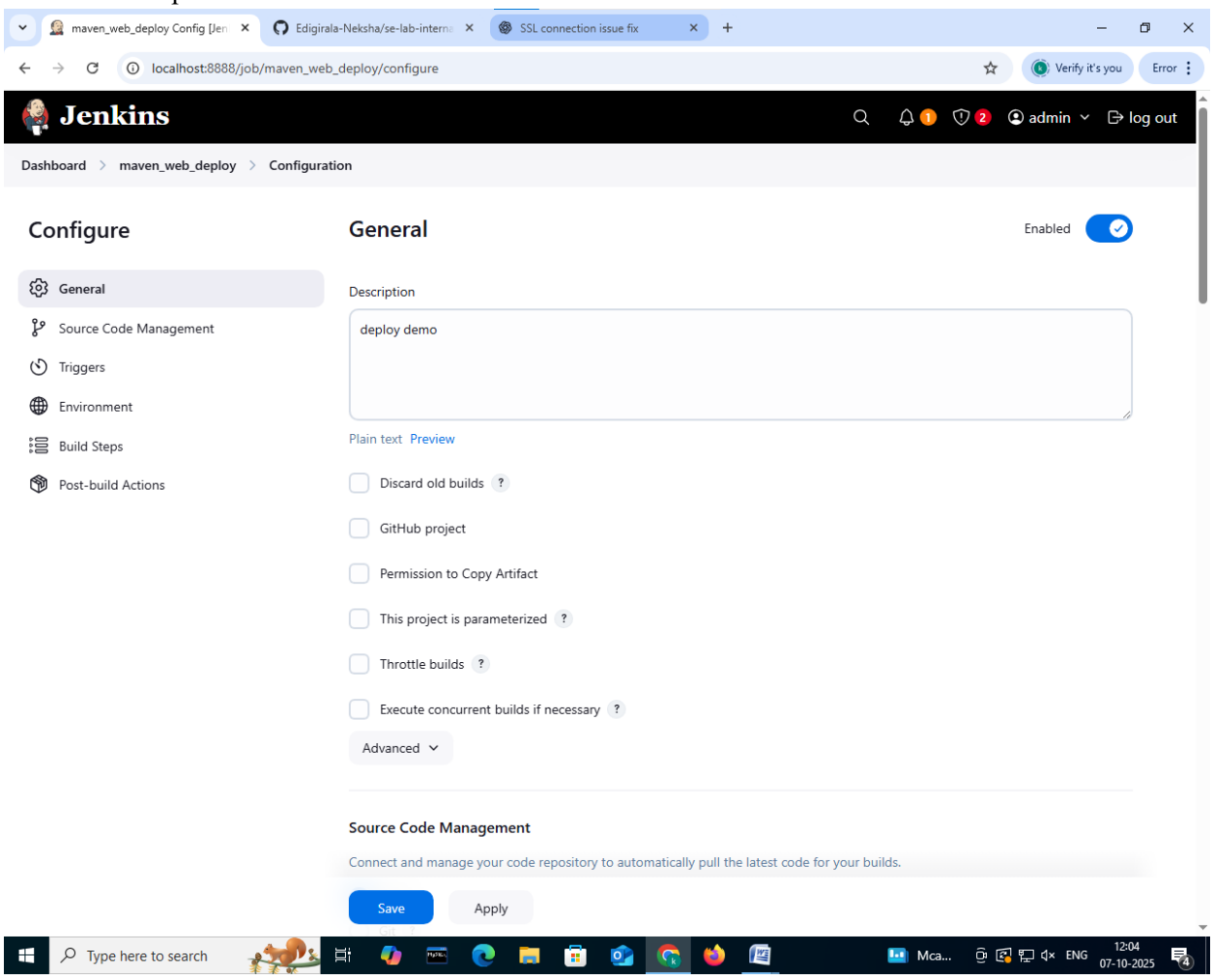
Step 1: Open Jenkins (localhost:8888)

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

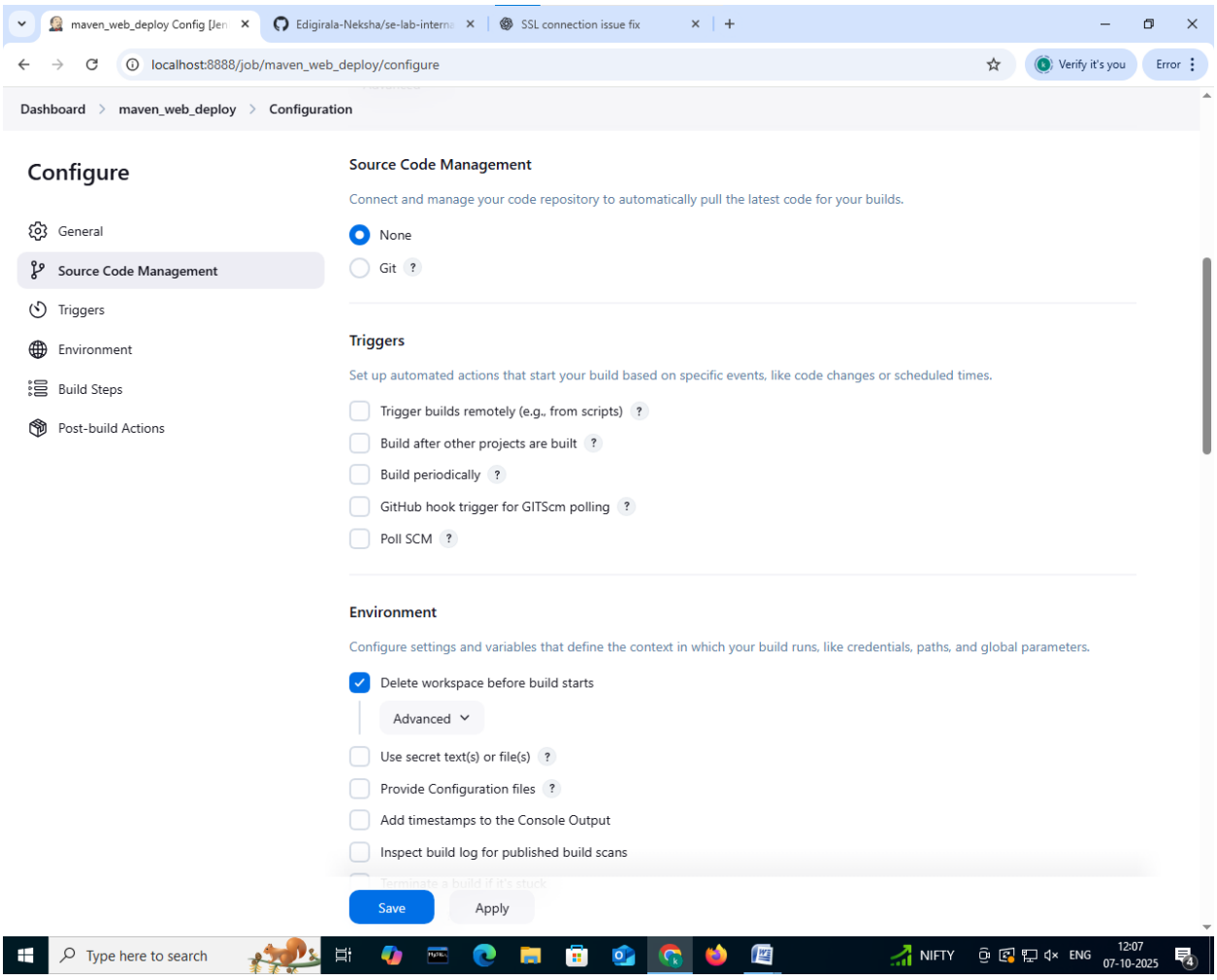


Step 2: Configuration of maven_web_deploy project

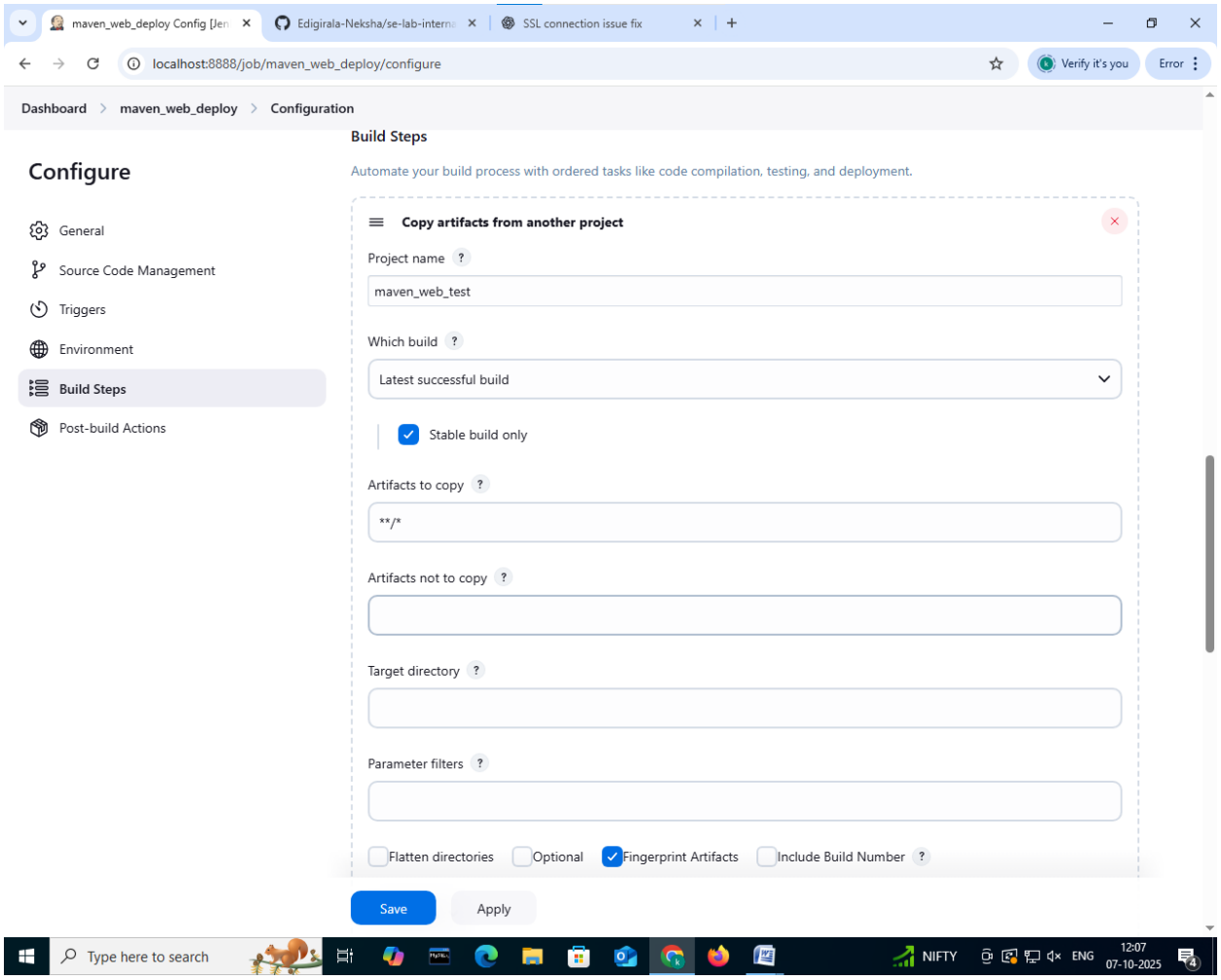
Give the description



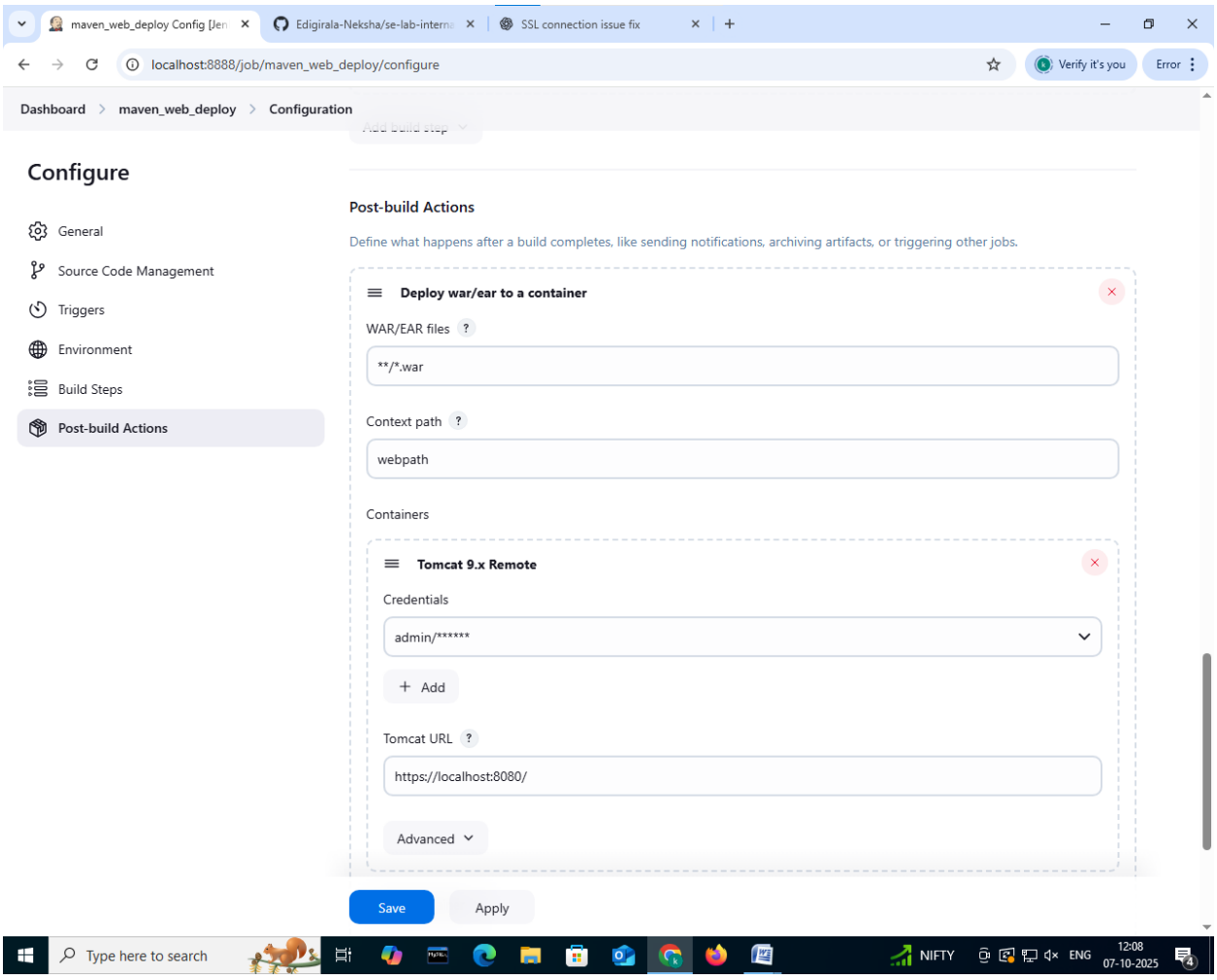
In the source code management select none and in environment select “delete workspace before build starts”



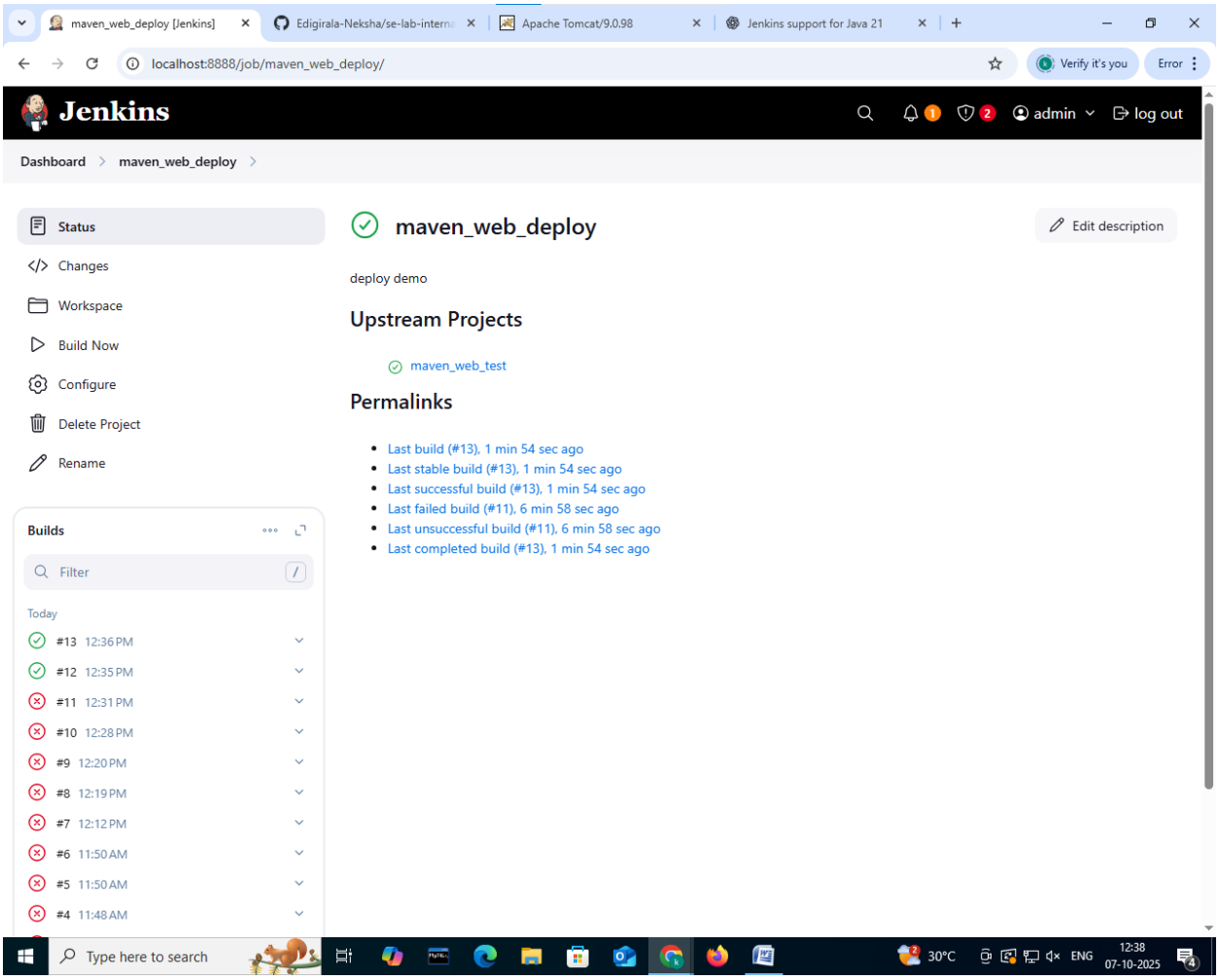
In the build steps click on add build step > select copy artifacts from another project > give project name as maven_web_test > give artifacts to copy as **/*



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



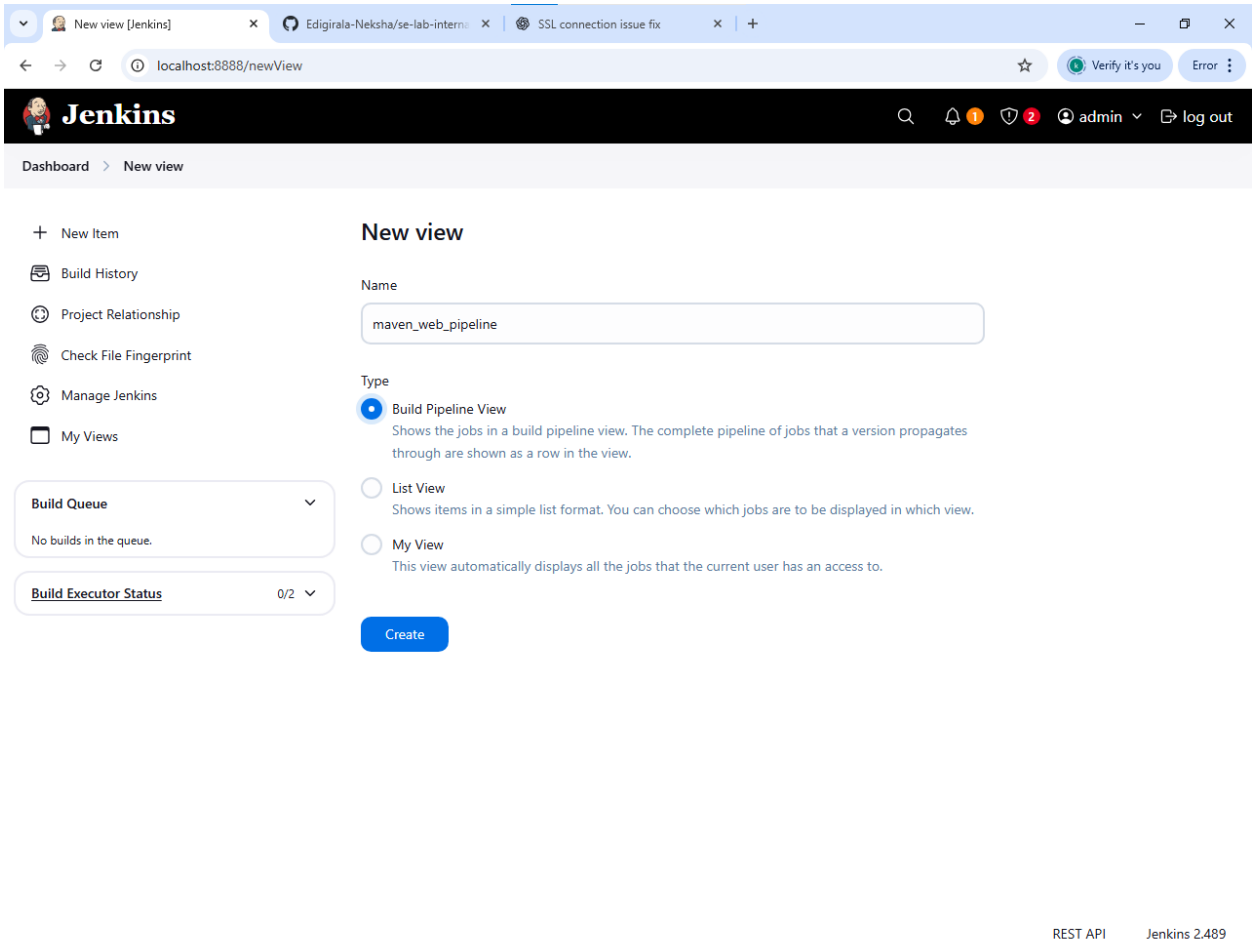
If the build is success:



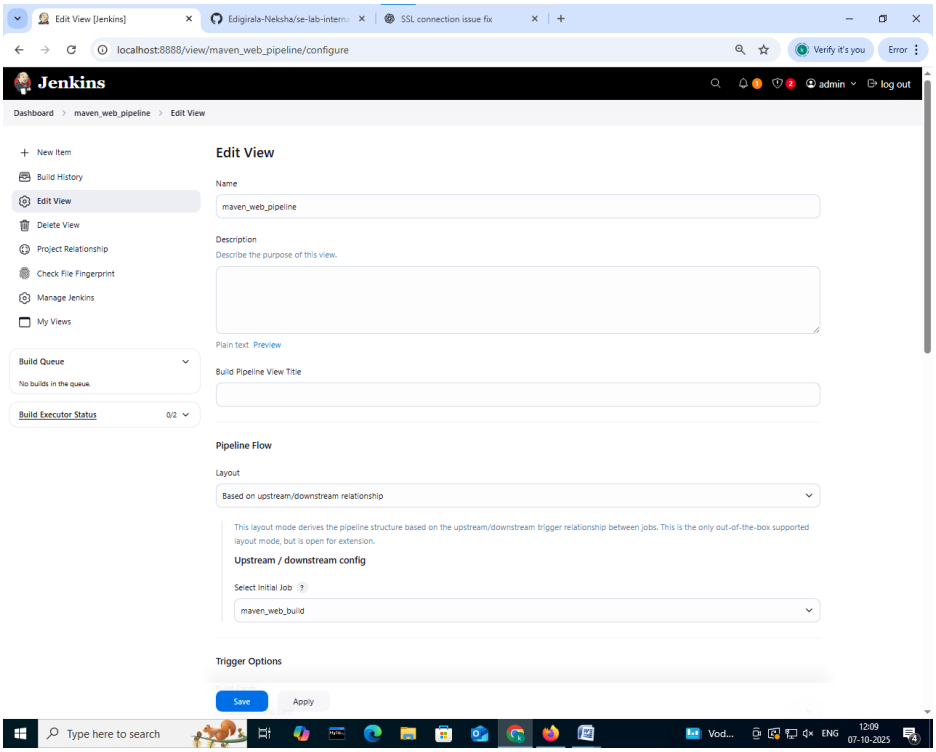
Create Pipeline View for MavenWeb

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

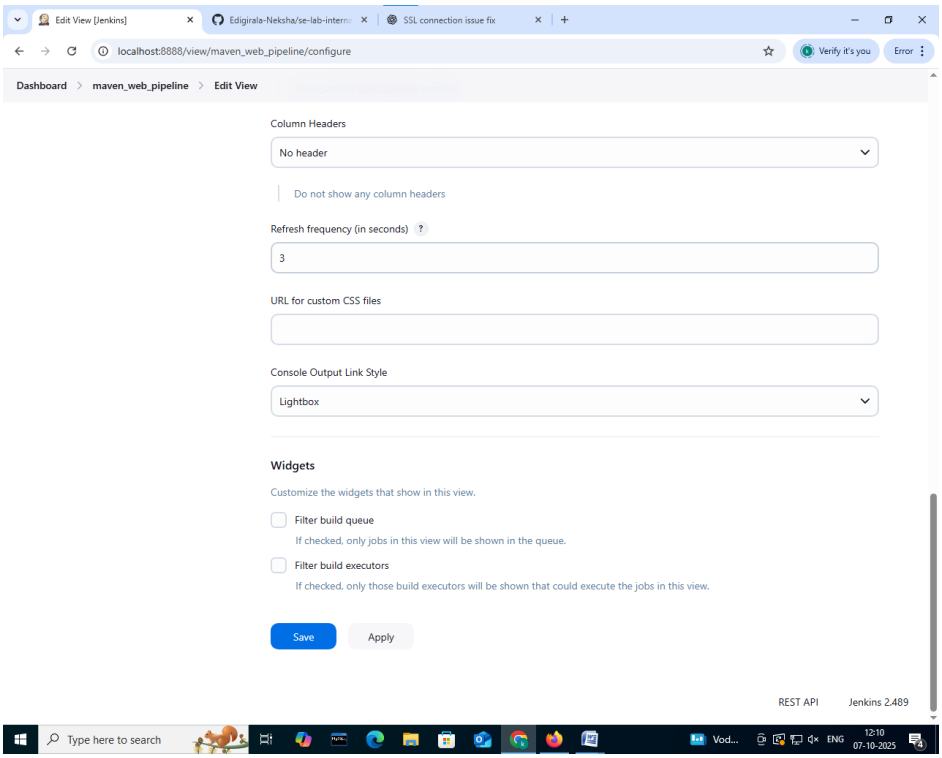
Select type as build pipeline view



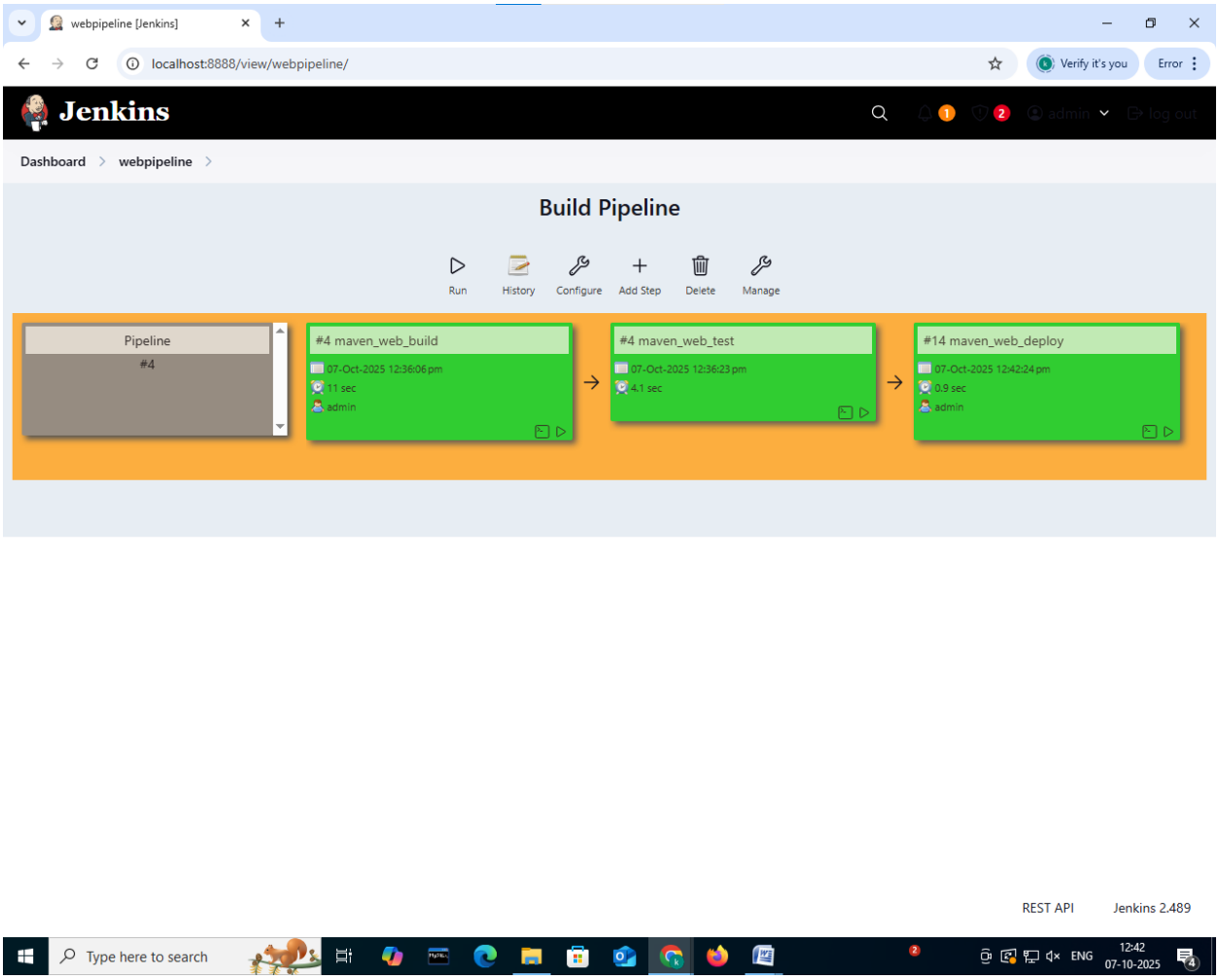
Give the description and in the upstream directly the maven_web_build will be shown



Click on apply and save

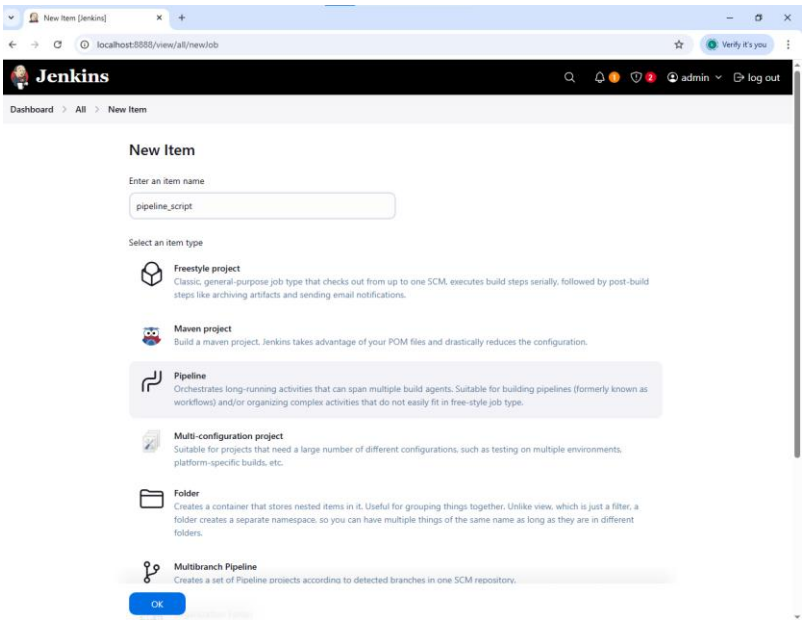


In the stage view it we be shown as:

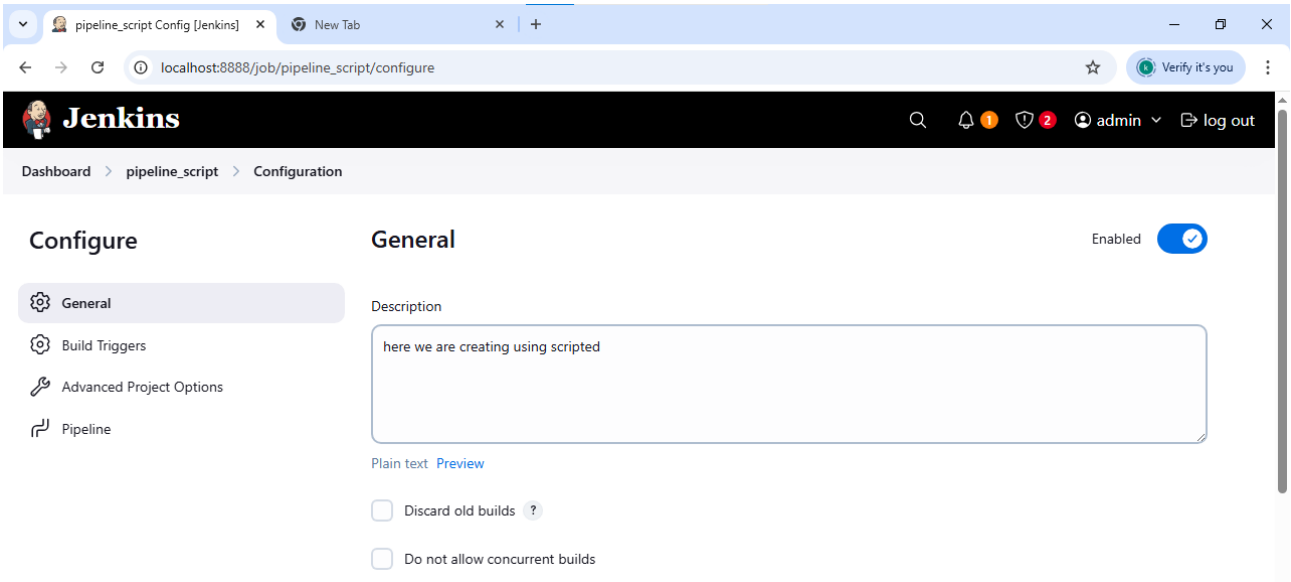


9.Pipeline Creation using script

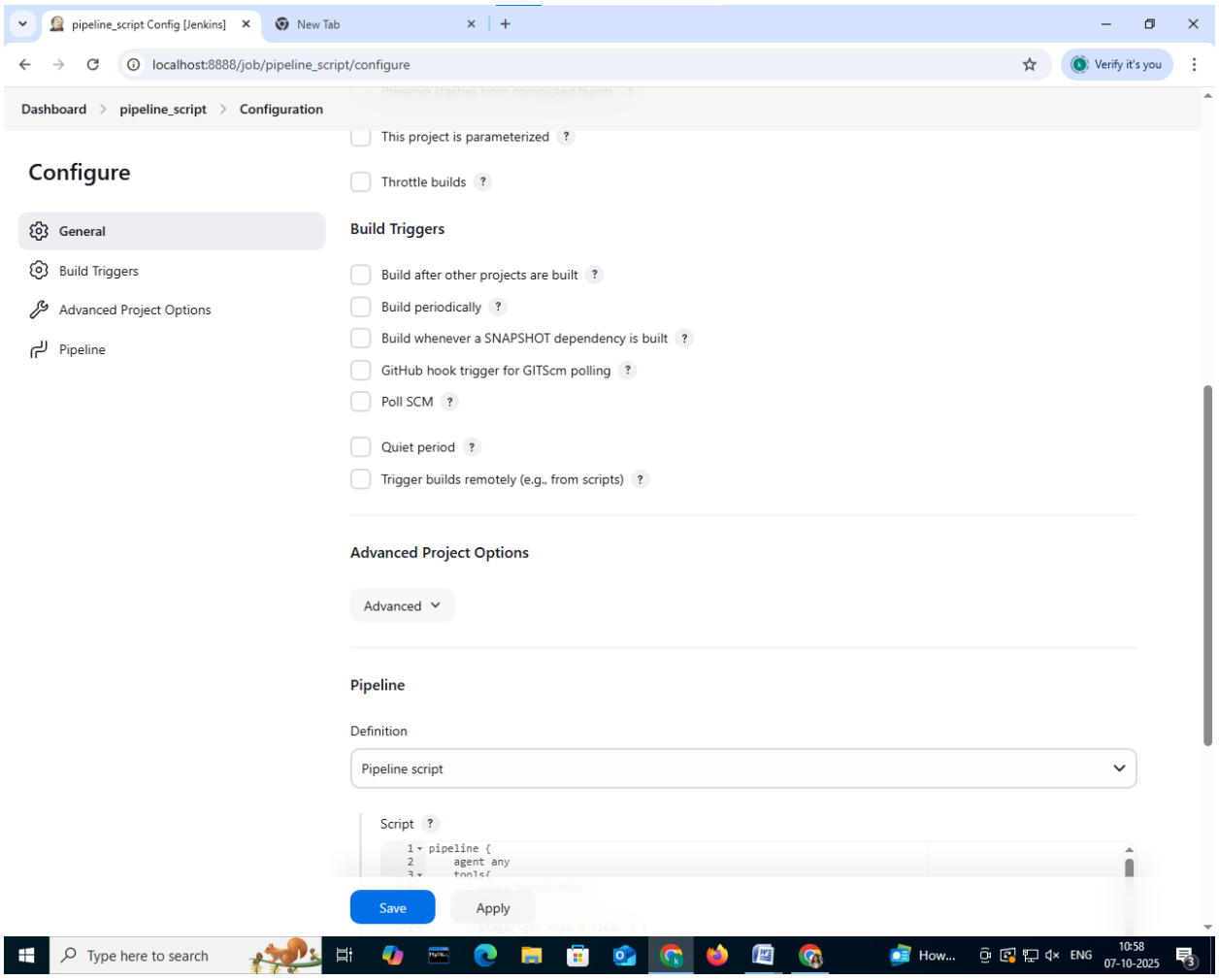
Step 1: In the Jenkins select the new item and give the name as pipeline_script and select pipeline and click ok



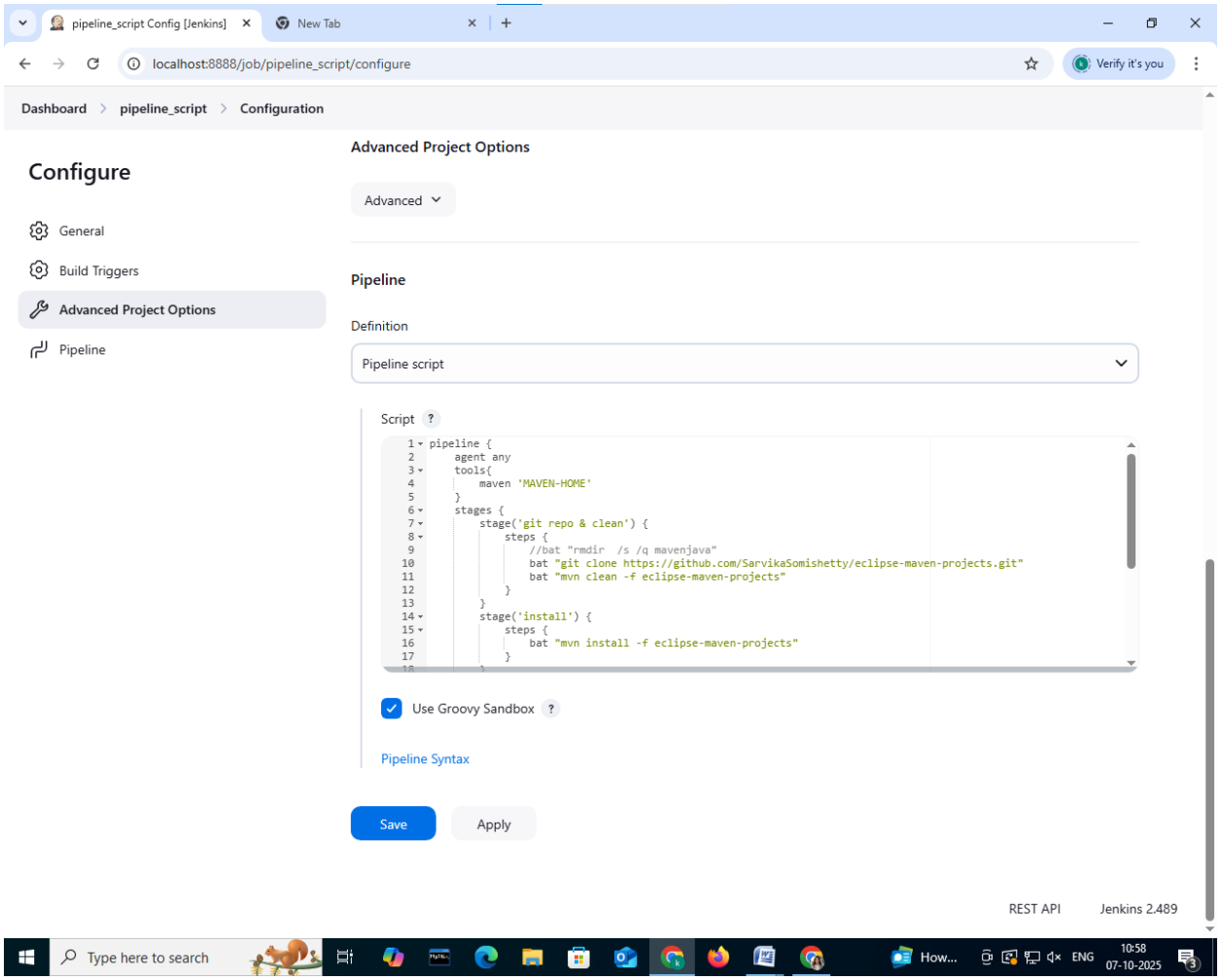
Step 2: In the configuration, give the description



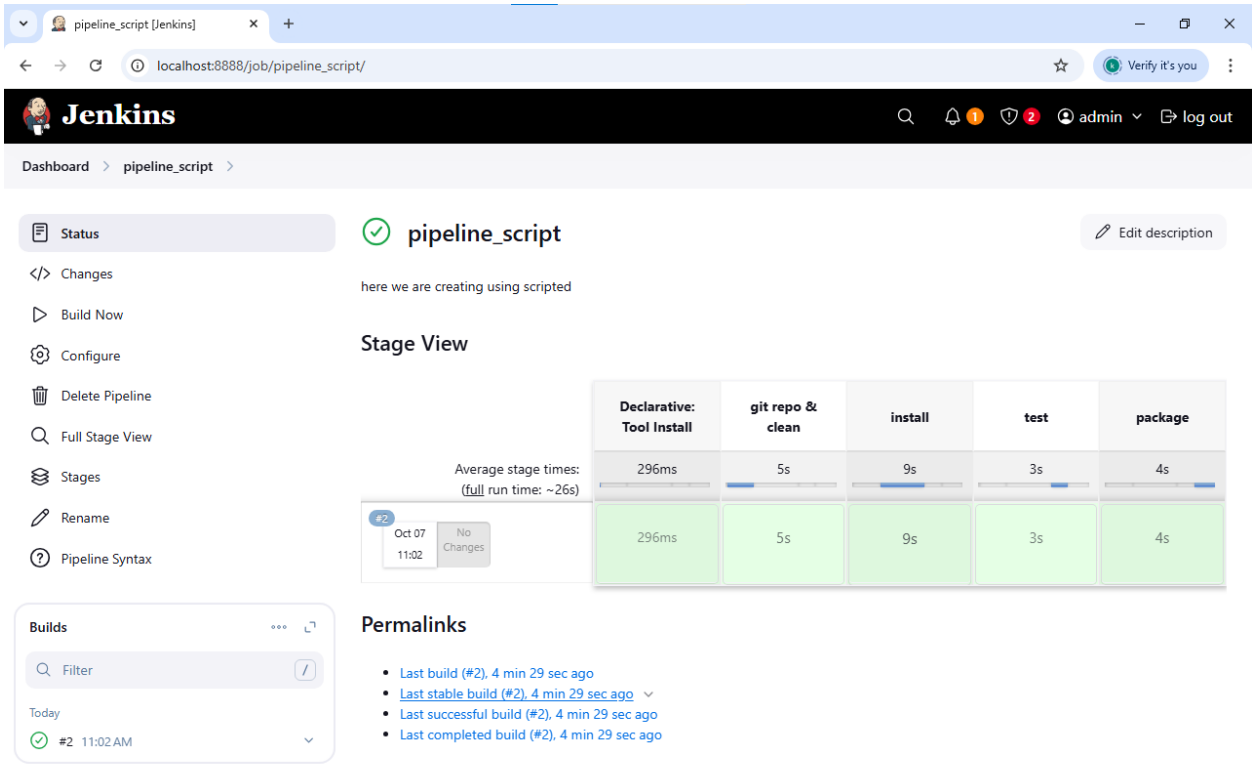
Step 3: In the pipeline section give definition as pipeline script and enter the script with git reop link and project name



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> 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:
```

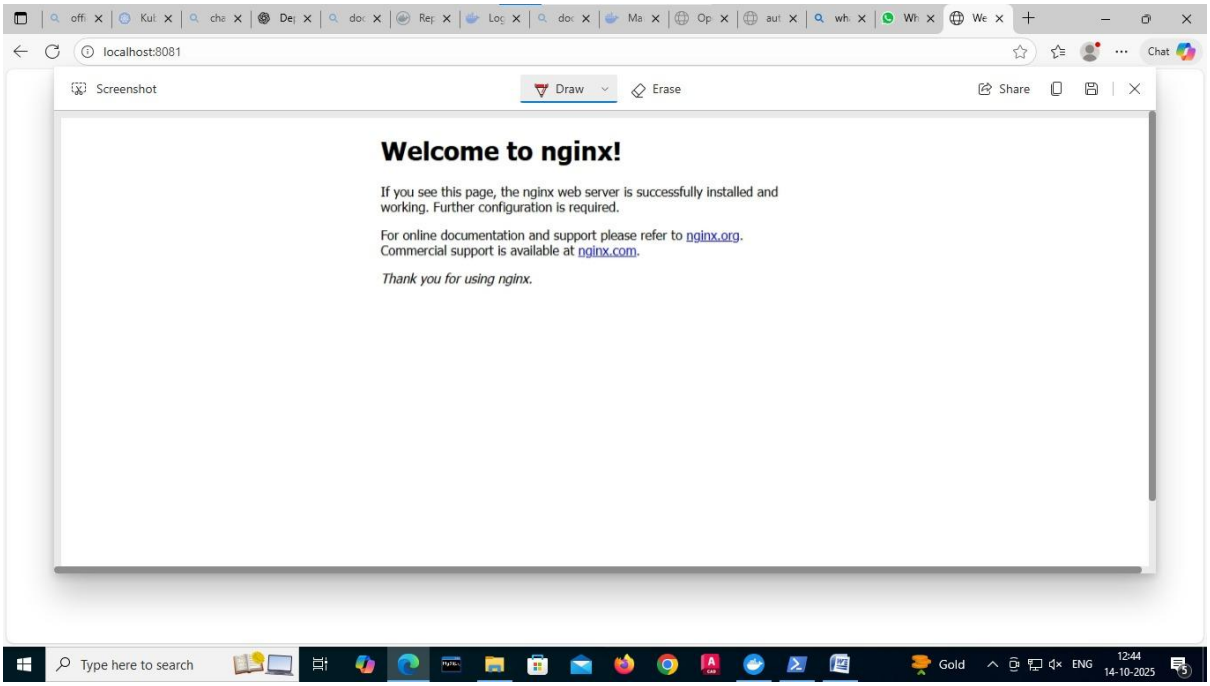
Step-6:Check the NGINX Service Details

- 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

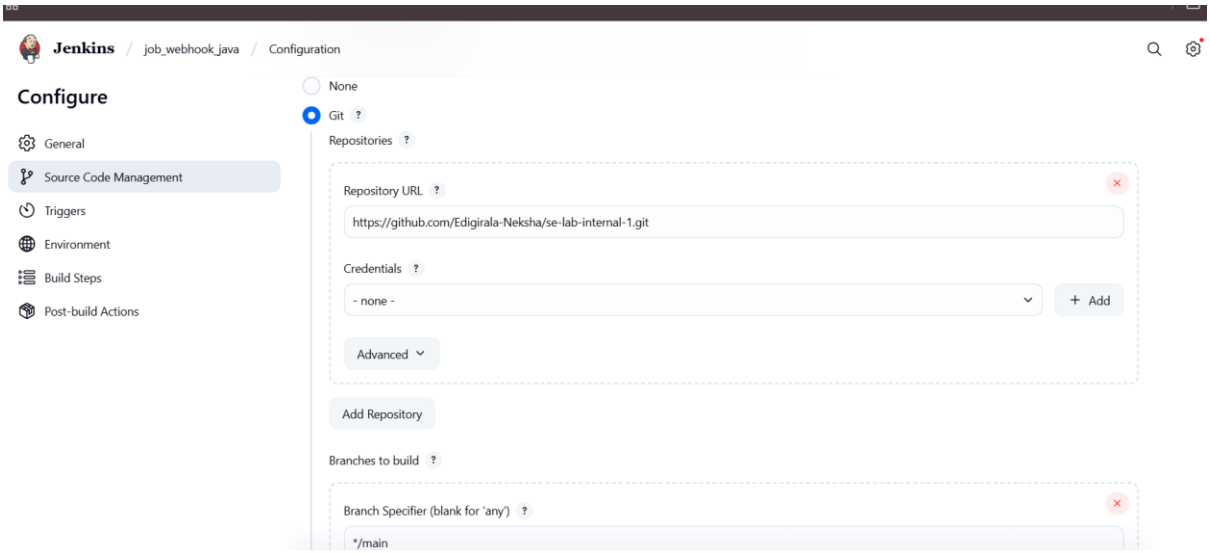
```
C:\Windows\System32>ngrok.exe http 8888
```

- Following output will be given:

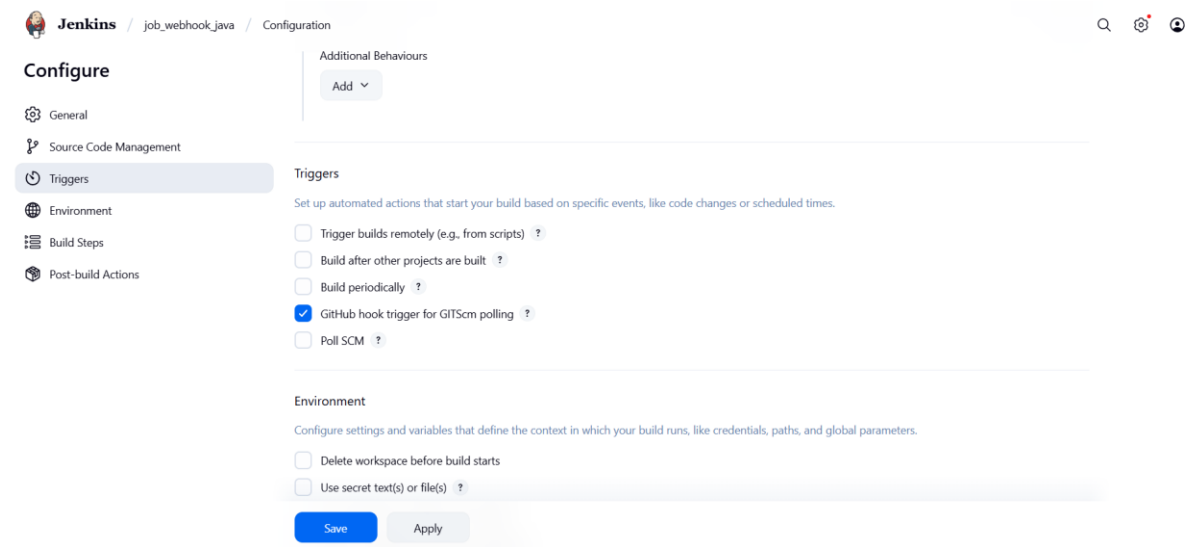
```
ngrok                                                                                    (Ctrl+C to quit)

◆  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)

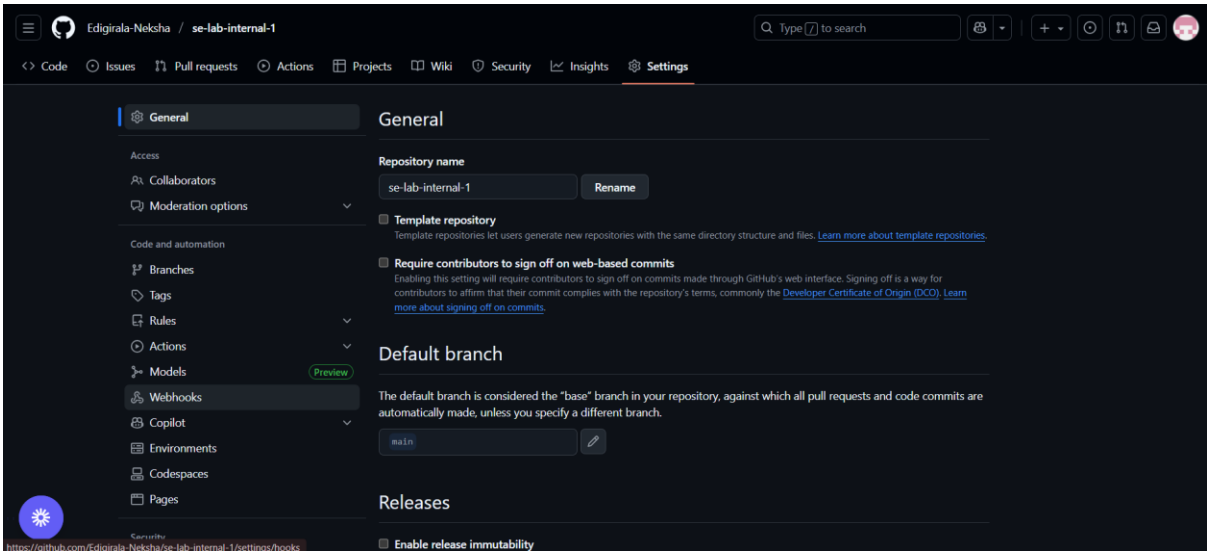


Step-4: In the triggers section select “Github hook trigger for GITScm polling”



Click on apply and save

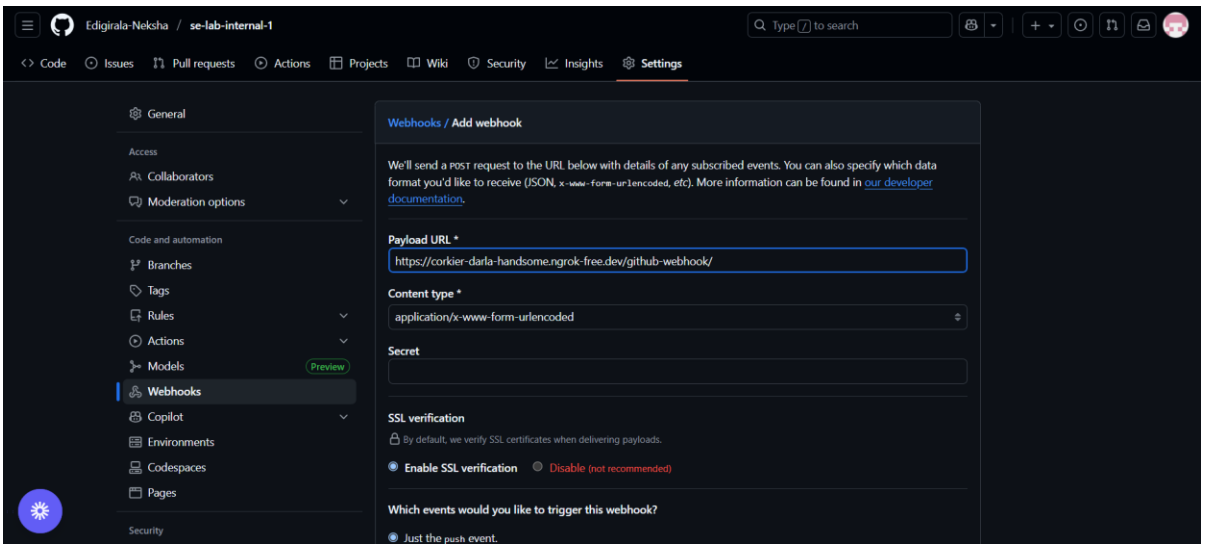
Step-6: open the git hub repo open setting of repo and then go to webhooks



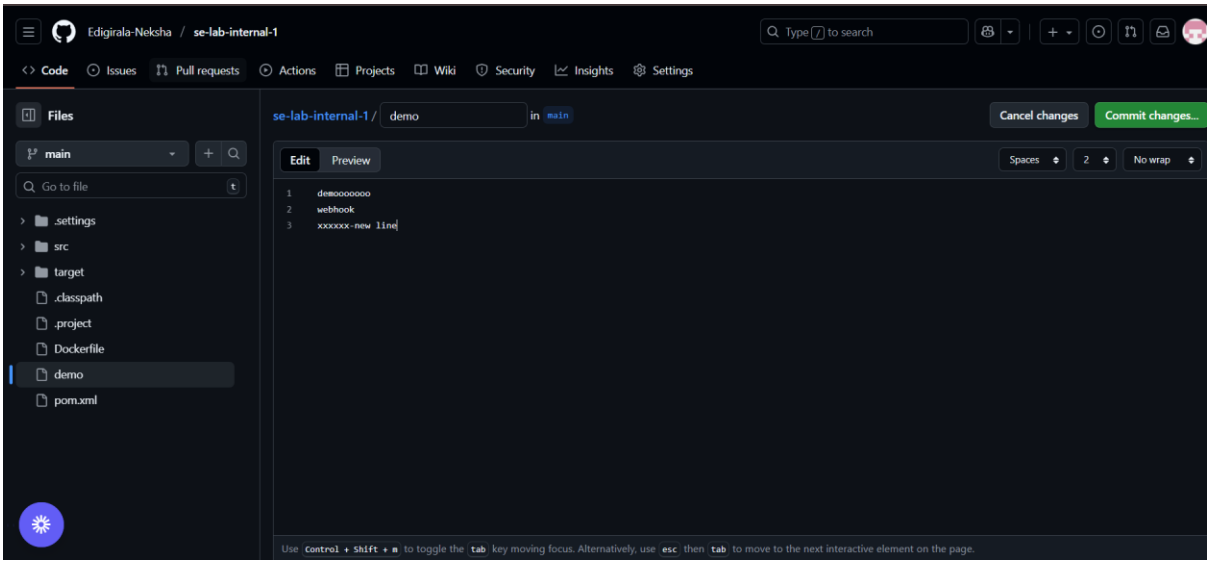
Step-7: Click on add a webhook and take the forwarding URL from ngrok and paste in payload URL and add /github-webhook/ along with the forwarding url

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

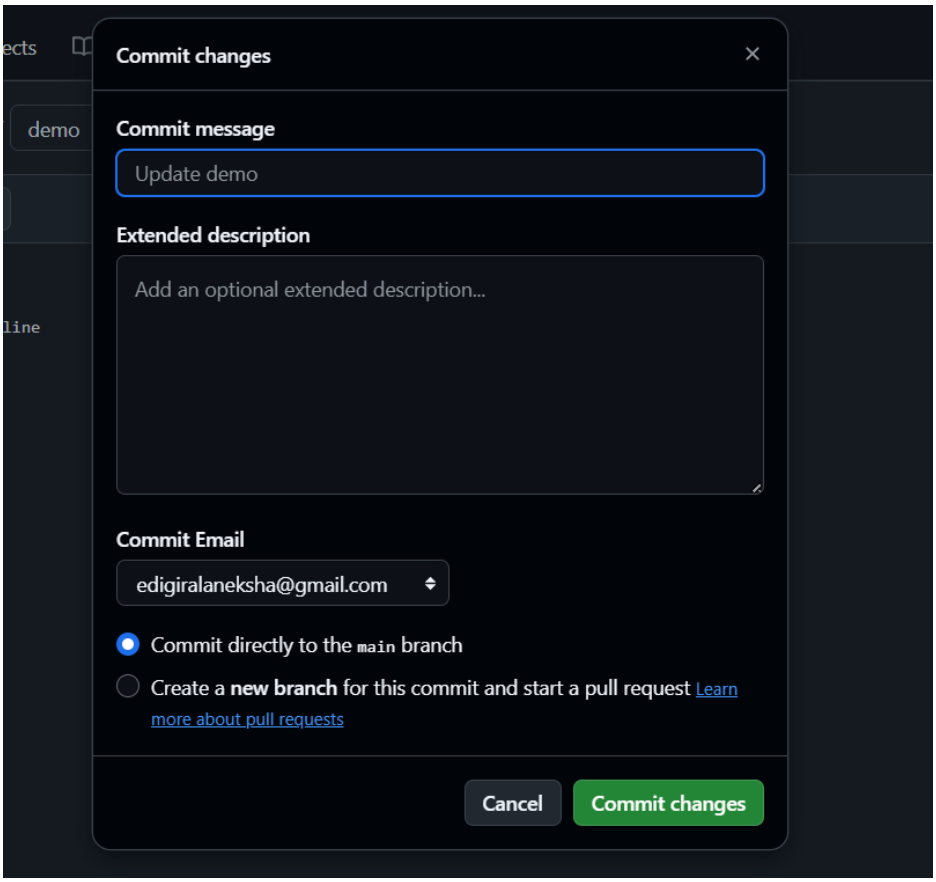
Payload url: https://corkier-darla-handsome.ngrok-free.dev/github-webhook/



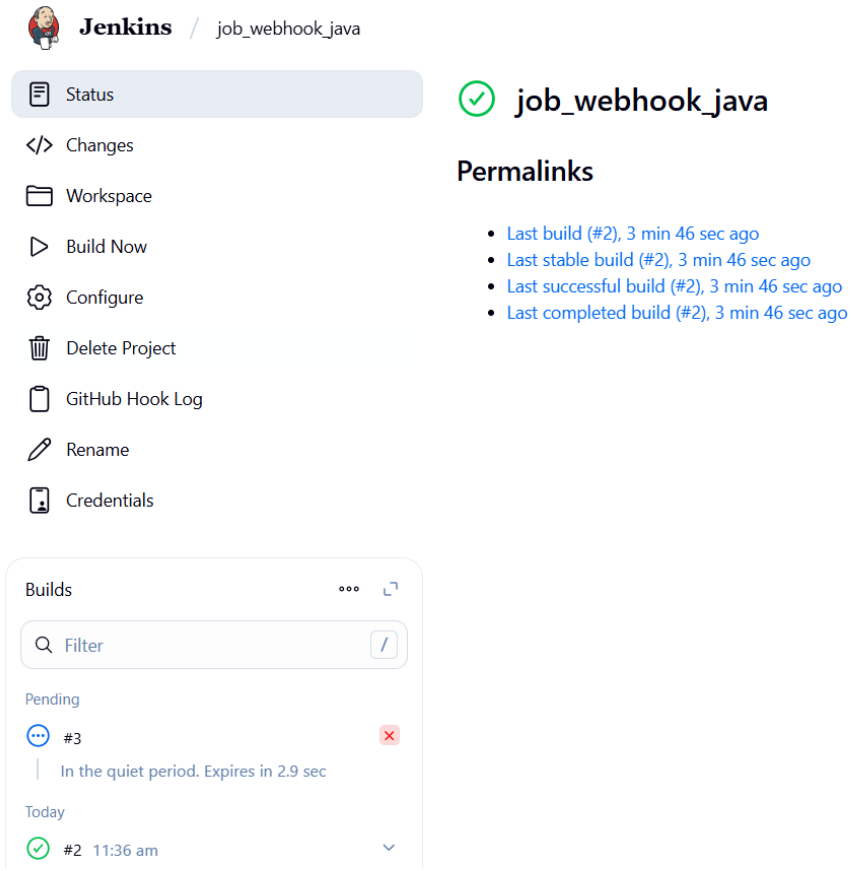
Step 8: make changes in the files in github

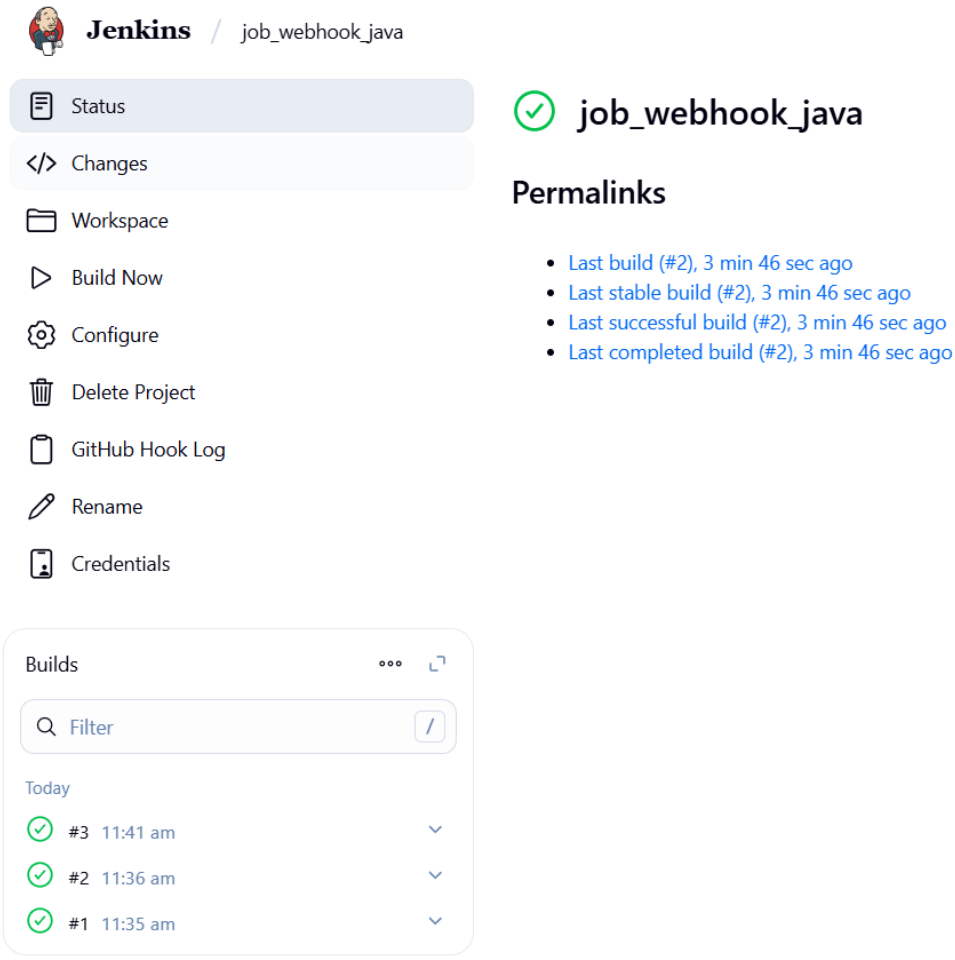


Step 9: click on commit changes

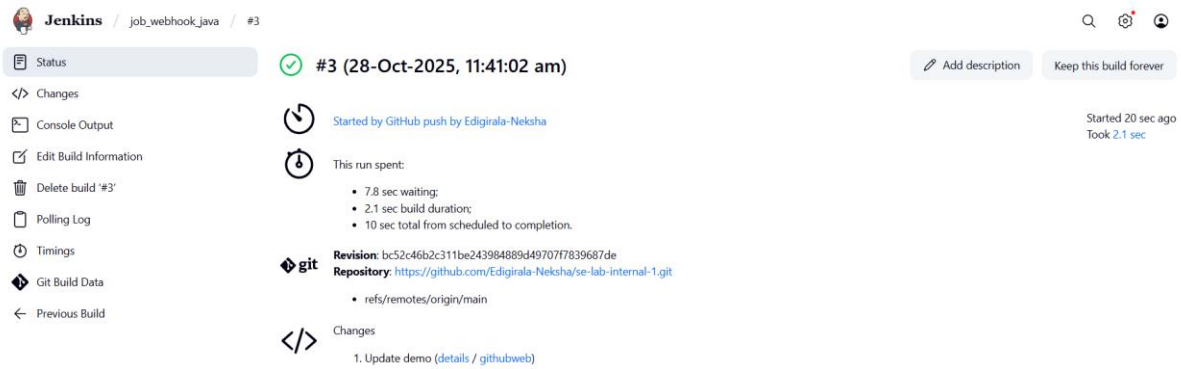


Step 10: open Jenkins the build will start automatically





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

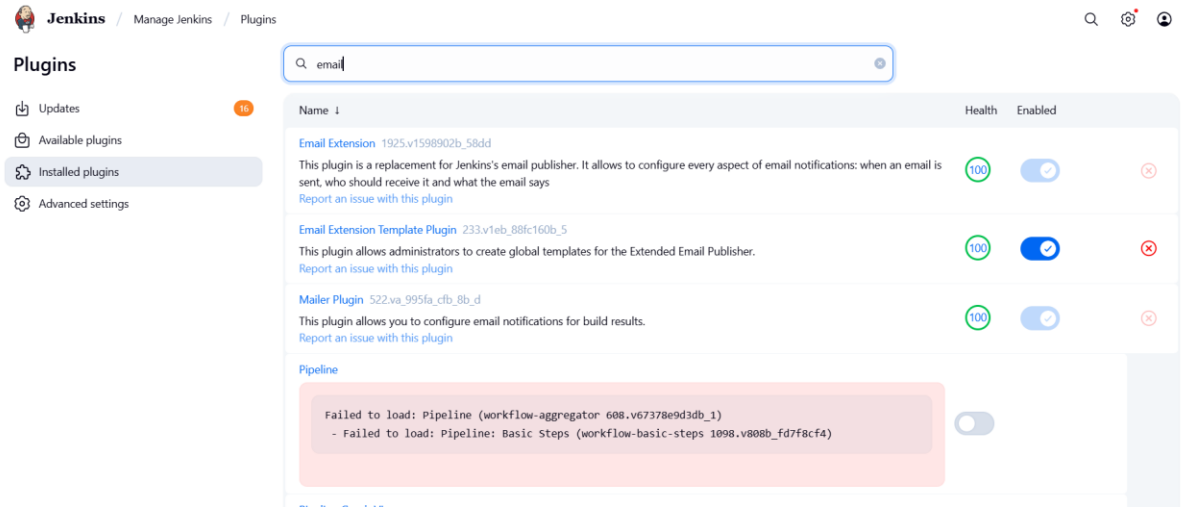
i. Open Jenkins Dashboard

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

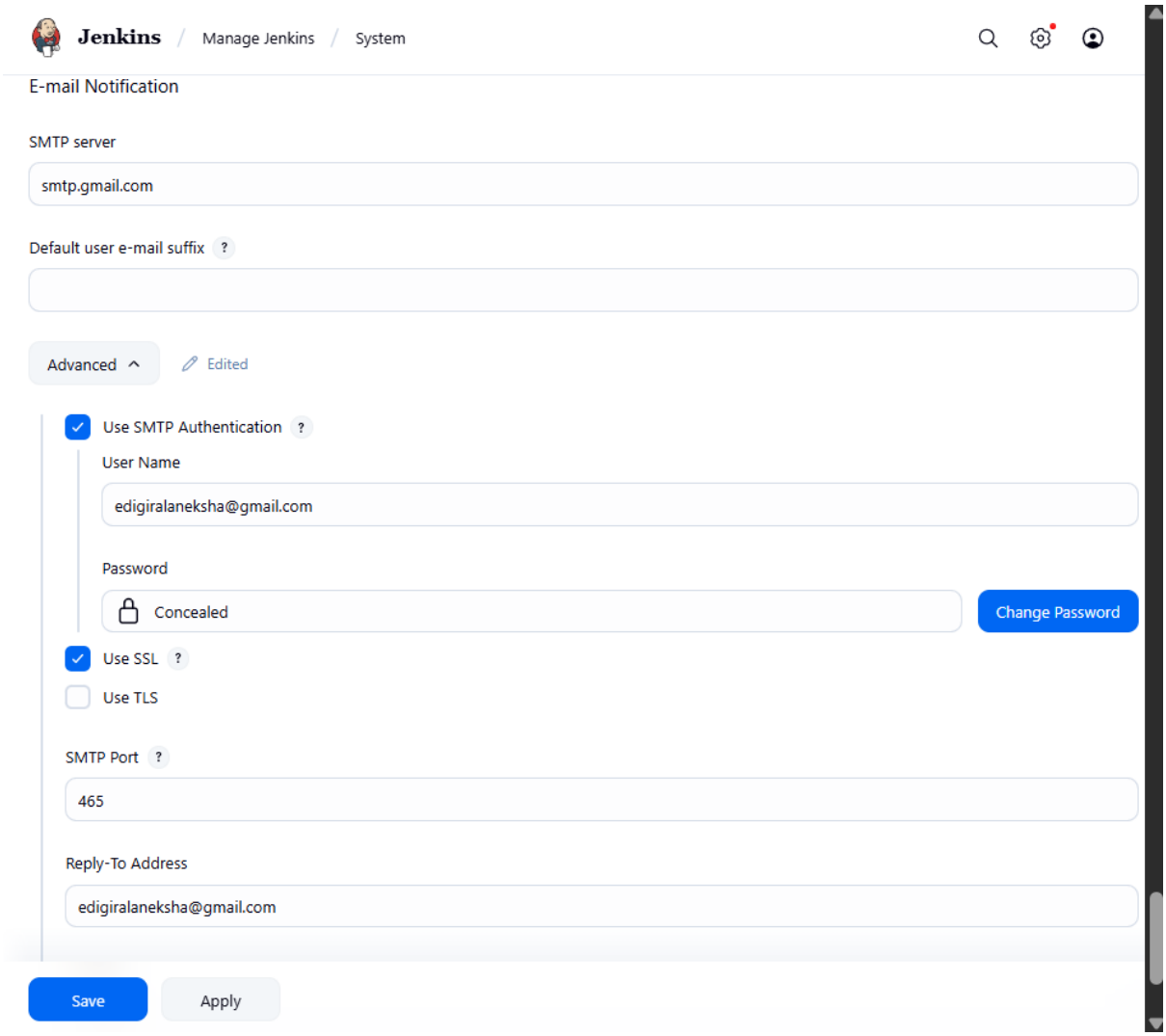
A. E-mail Notification Section

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

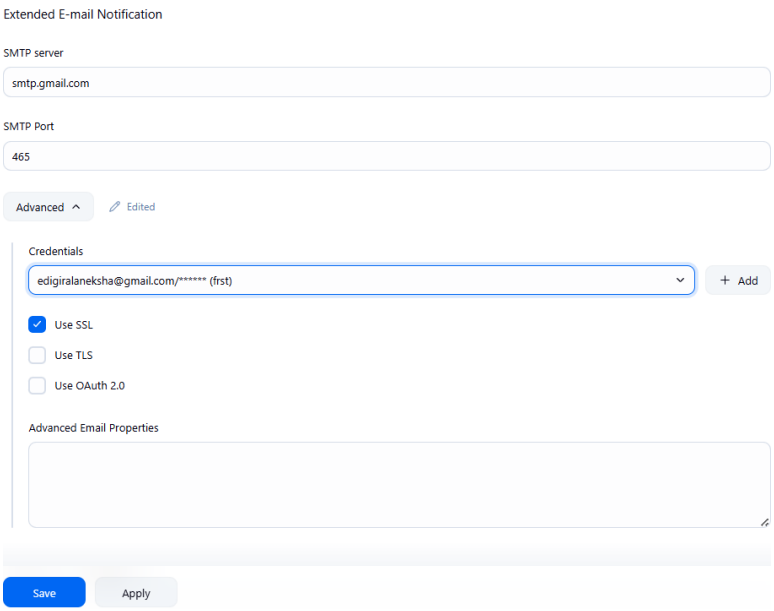
- Click: Test configuration by sending test e-mail
- Provide a valid email address to receive a test mail
- ✅ Should receive email from Jenkins

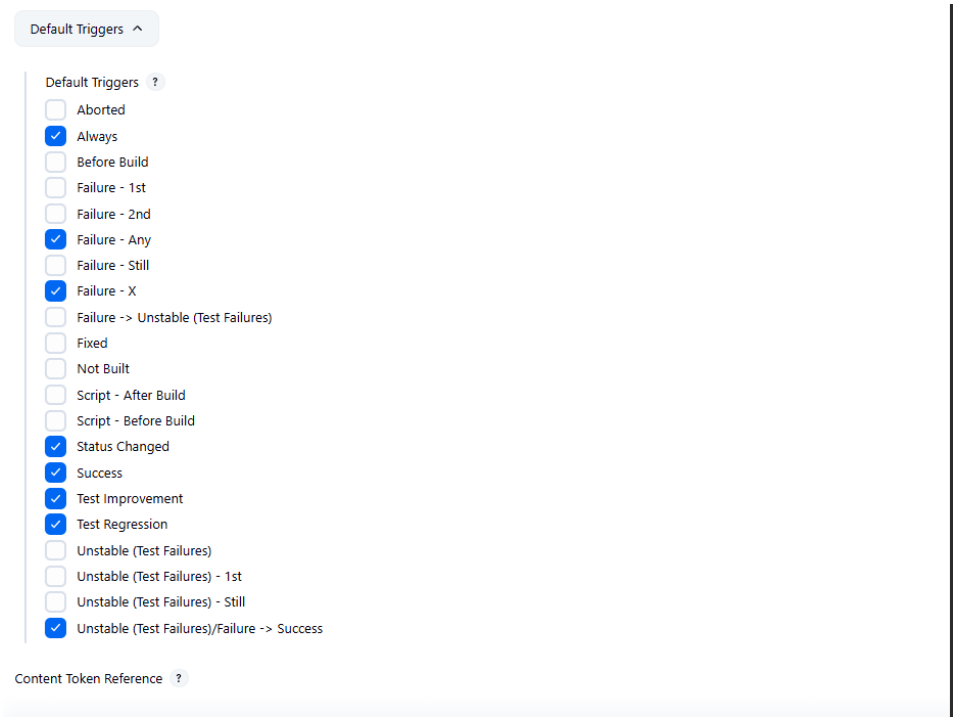


B. Extended E-mail Notification Section

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

Field	Value
Default Recipients	Leave empty or provide default emails
Triggers	Select as per needs (e.g., Failure)

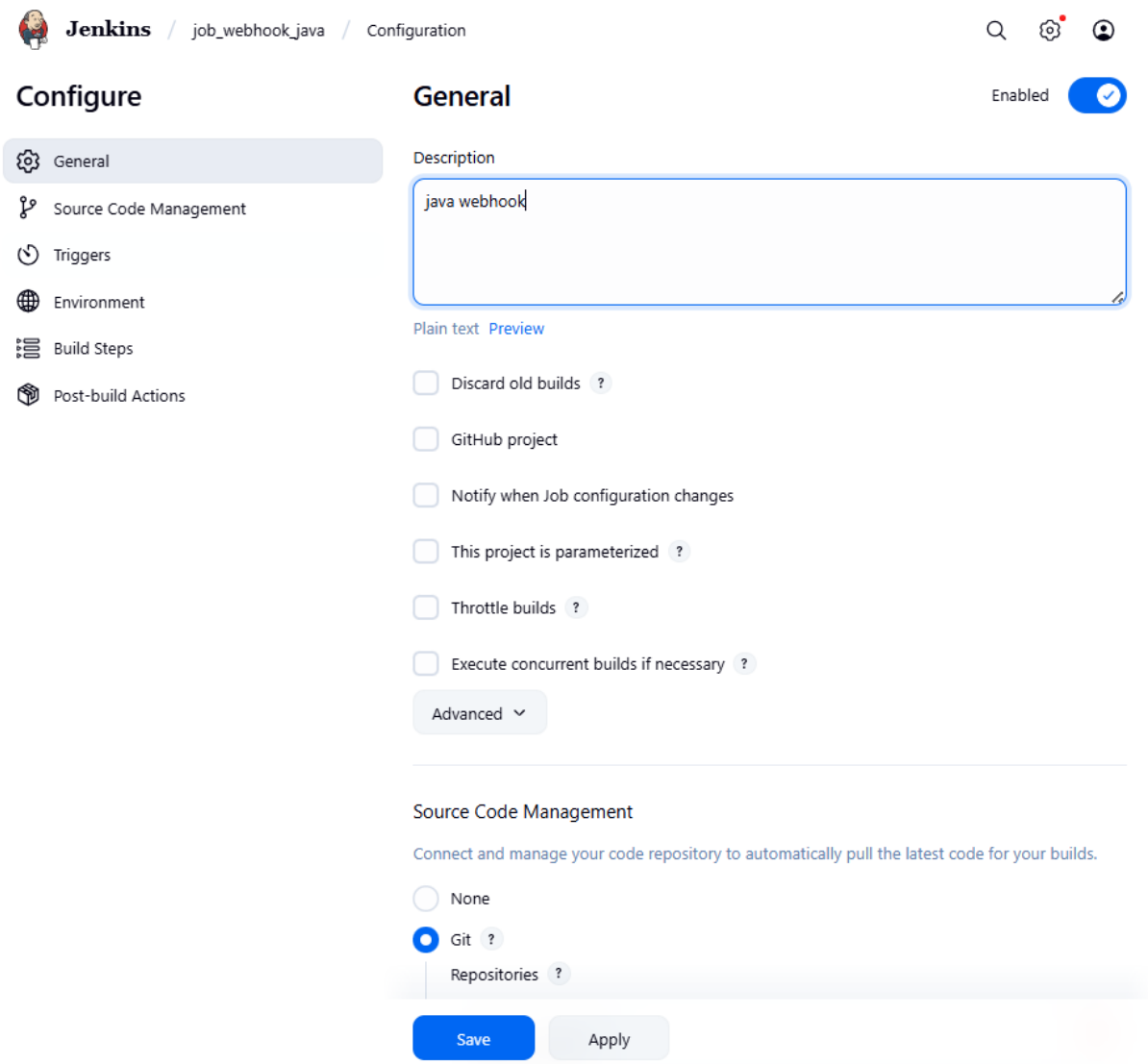




4. Configure Email Notifications for a Jenkins Job

i. Go to:

- Jenkins → Select a Job → Configure



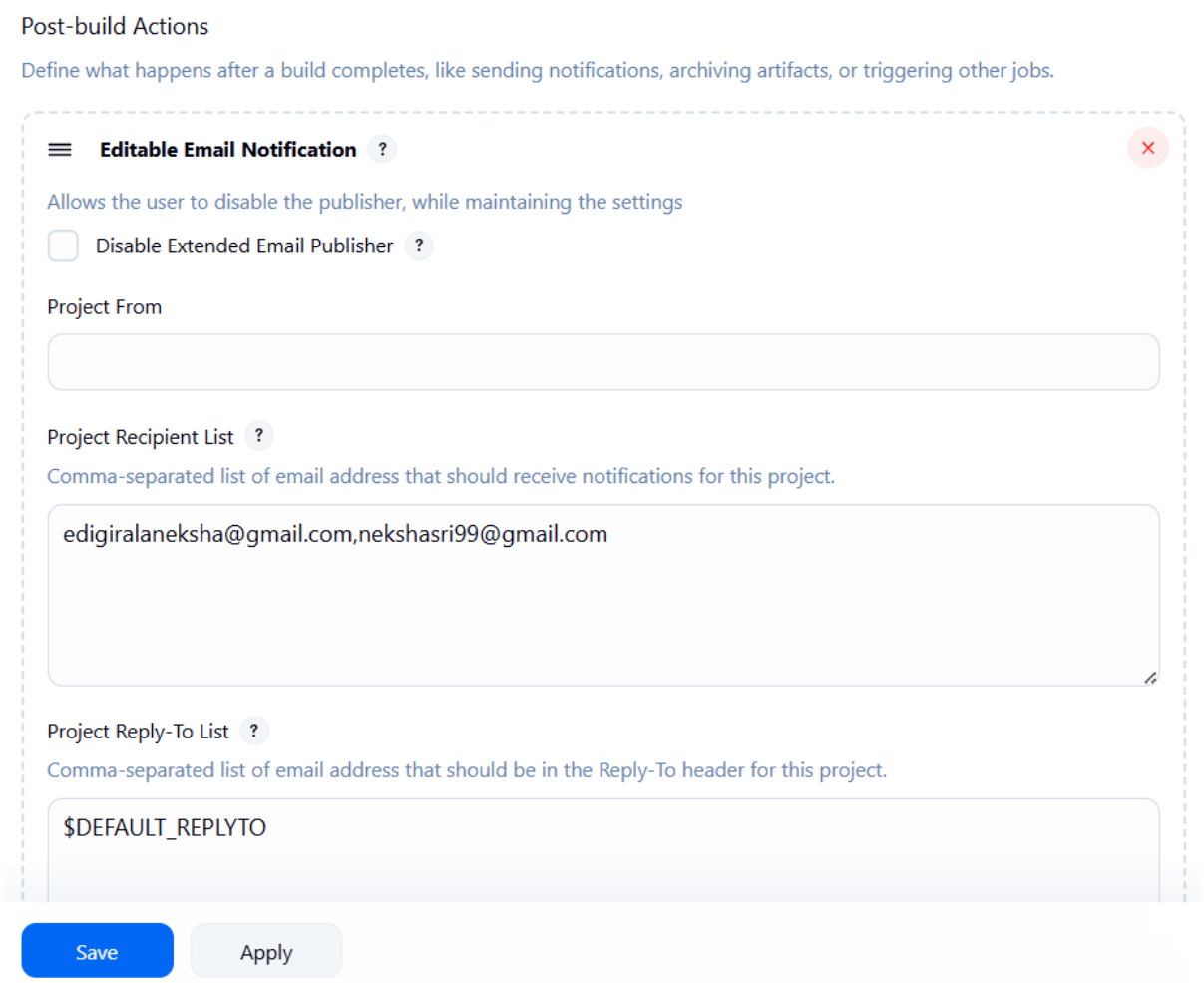
ii. In the Post-build Actions section:

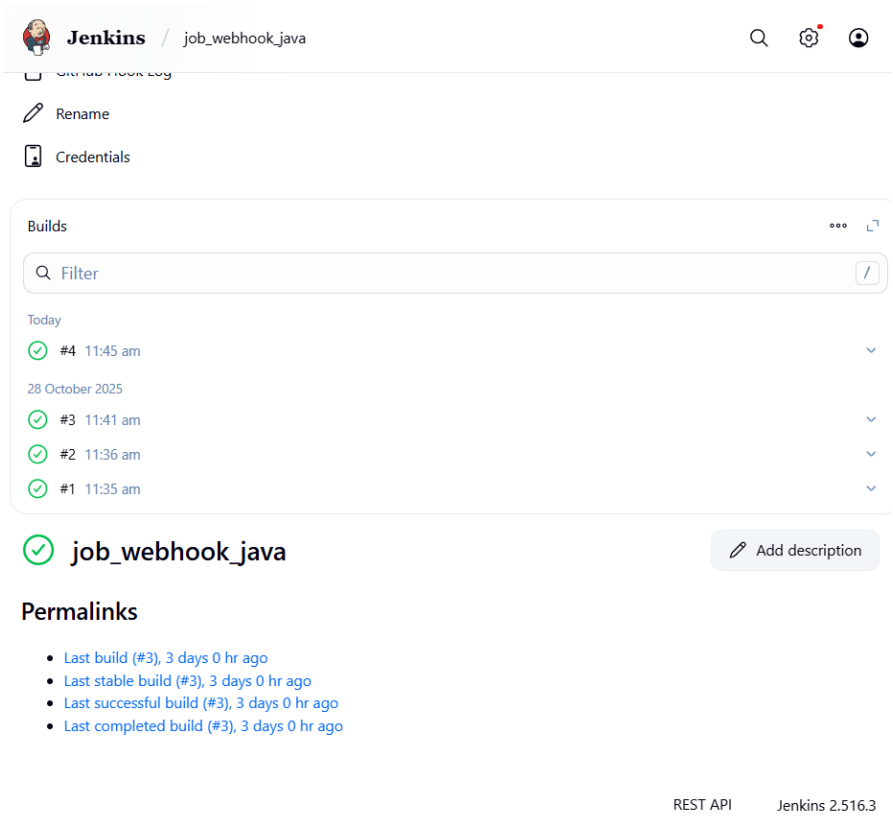
- Click: Add post-build action → **Editable Email Notification**

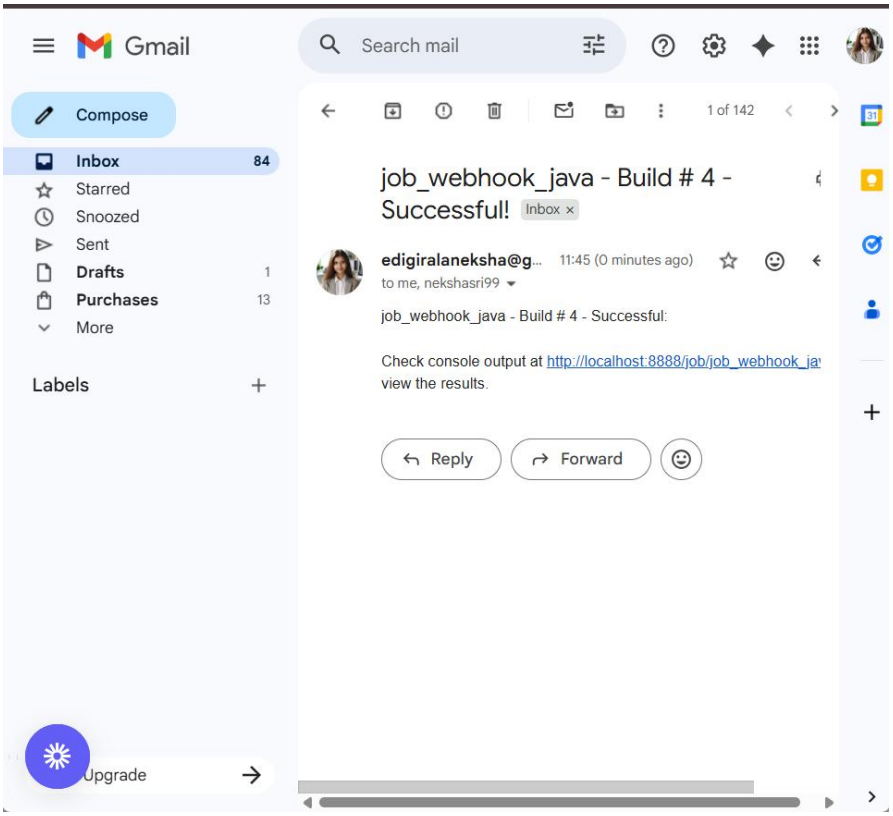
A. Fill in the fields:

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



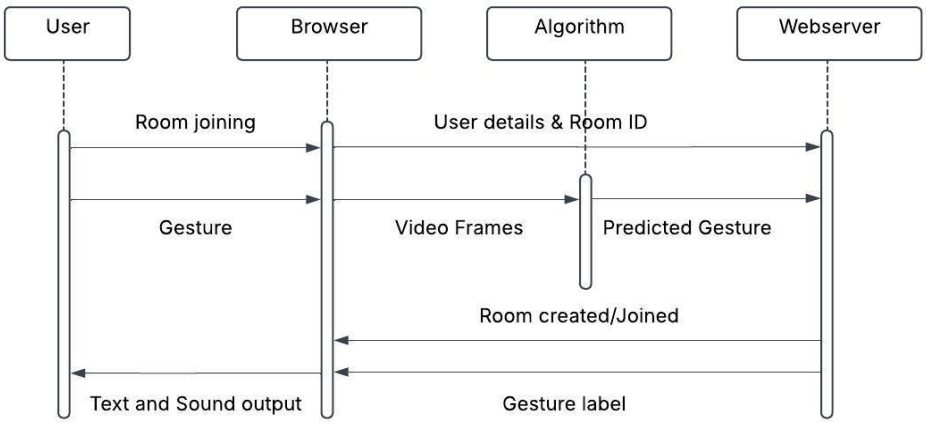

Jenkins / job_webhook_java
Rename
Credentials
Builds
Filter
Today
#4 11:45 am
28 October 2025
#3 11:41 am
#2 11:36 am
#1 11:35 am
job_webhook_java
Add description
Permalinks
Last build (#3), 3 days 0 hr ago
Last stable build (#3), 3 days 0 hr ago
Last successful build (#3), 3 days 0 hr ago
Last completed build (#3), 3 days 0 hr ago
REST API
Jenkins 2.516.3


Gmail
Search mail
Compose
Inbox 84
Starred
Snoozed
Sent
Drafts 1
Purchases 13
More
Labels
Upgrade
1 of 142
job_webhook_java - Build # 4 - Successful!
Inbox ×
edigiralaneksha@g... 11:45 (0 minutes ago)
to me, nekshasri99
job_webhook_java - Build # 4 - Successful:
Check console output at http://localhost:8888/job/job_webhook_ja
view the results.
Reply
Forward

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.

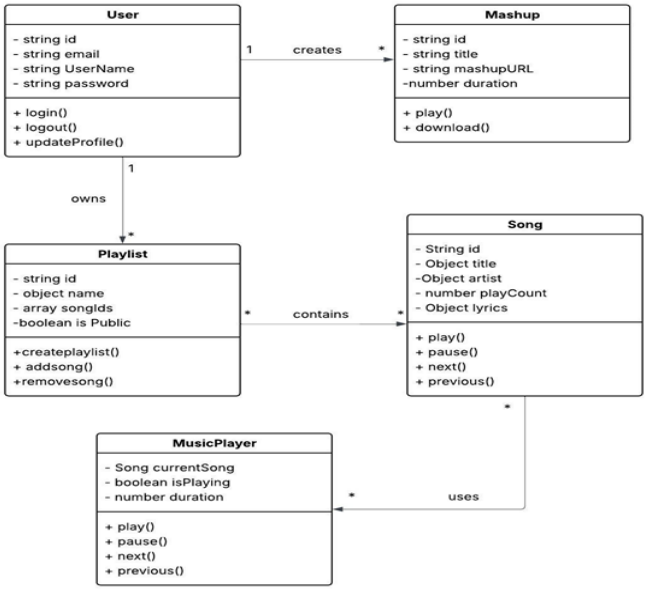
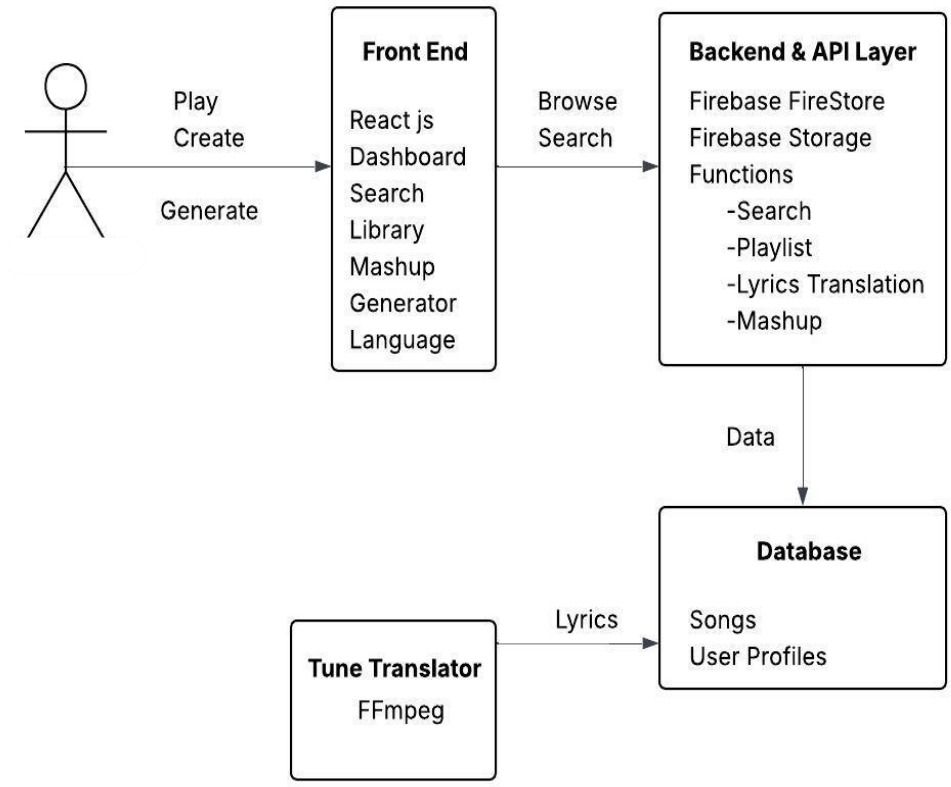


Fig 5: Class Diagram for TuneOra

3. Component Diagram:

A component diagram illustrates how different software components are connected and interact to form a complete system.

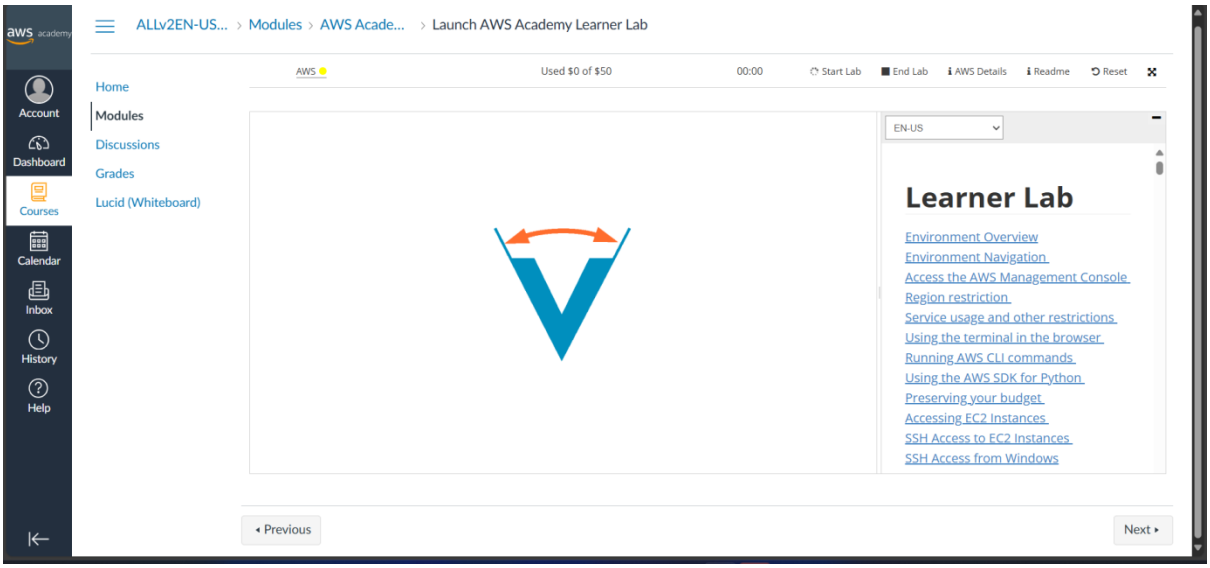
It helps visualize the organization and dependencies among modules or subsystems.



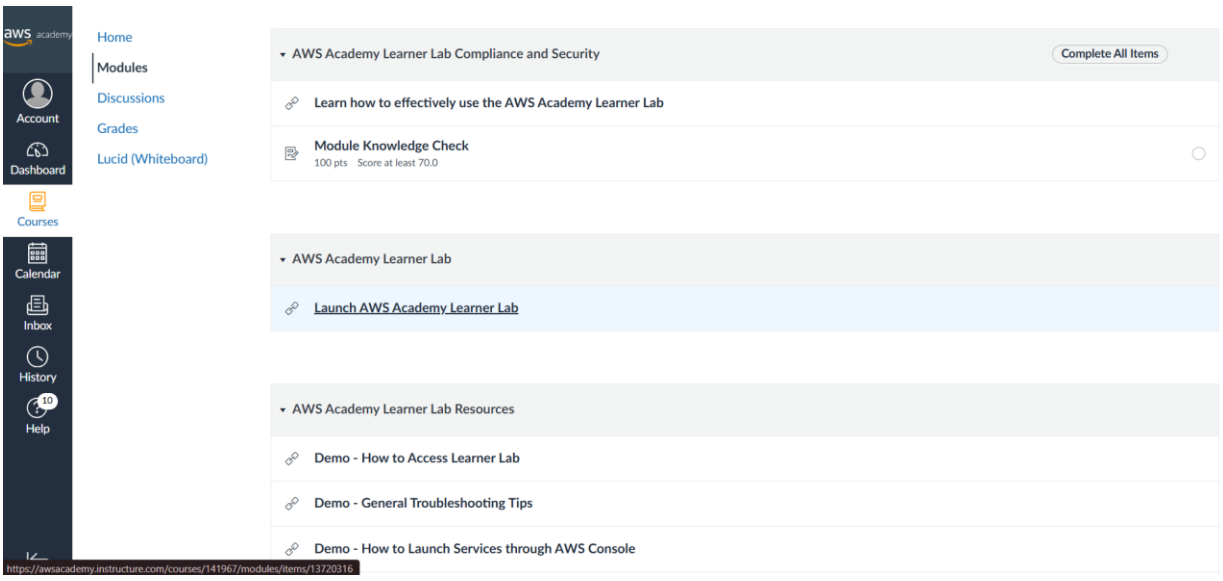
12.Creation of virtual machine for Ubuntu OS and Deploying the web application

DEPLOYMENT OF INDEX.HTML USING EC2 INSTANCE in AWS

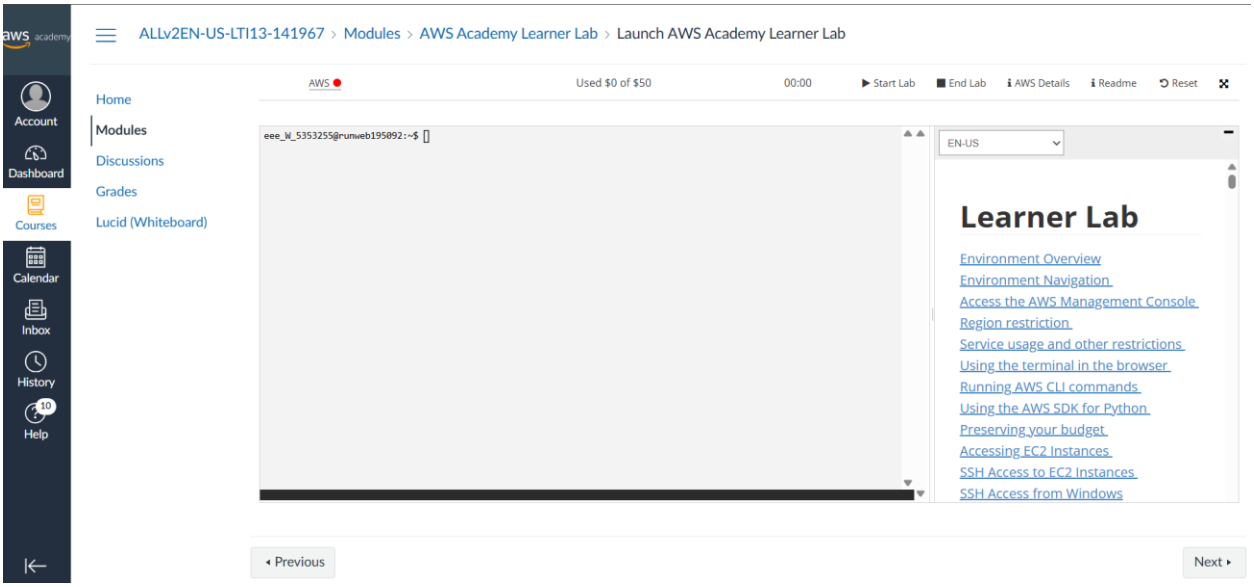
Step 1: Click on Modules.



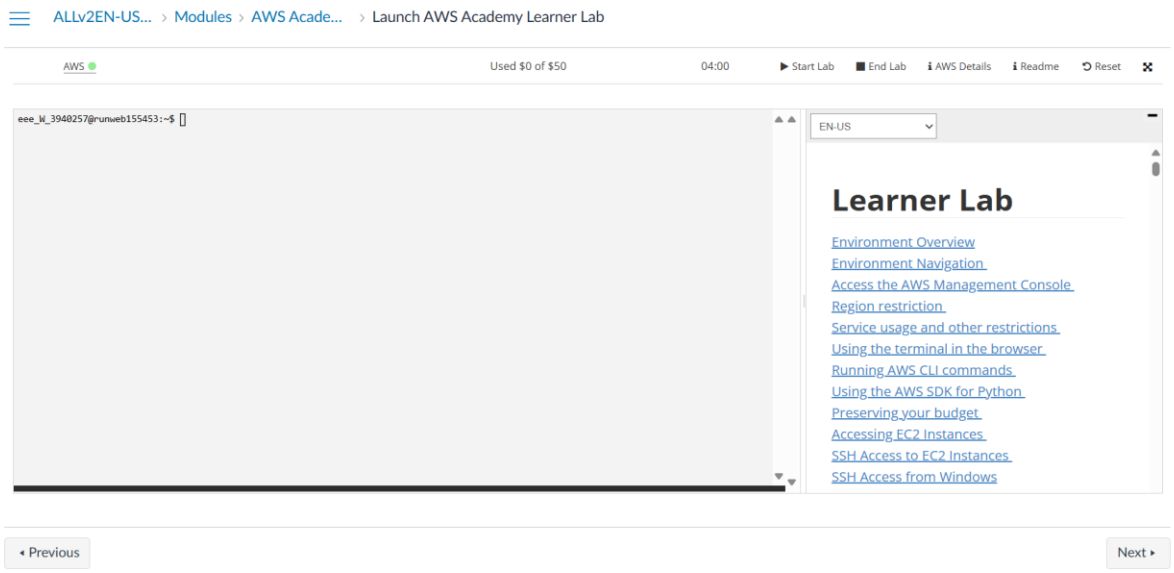
Step 2: Scroll down and select Lunch AWS Academy Lab



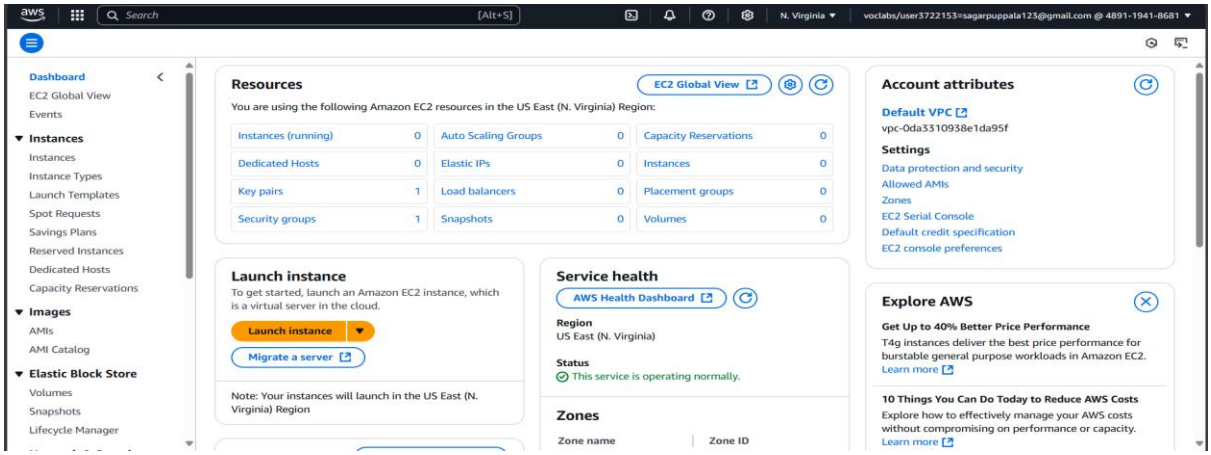
Step 3: click on start lab



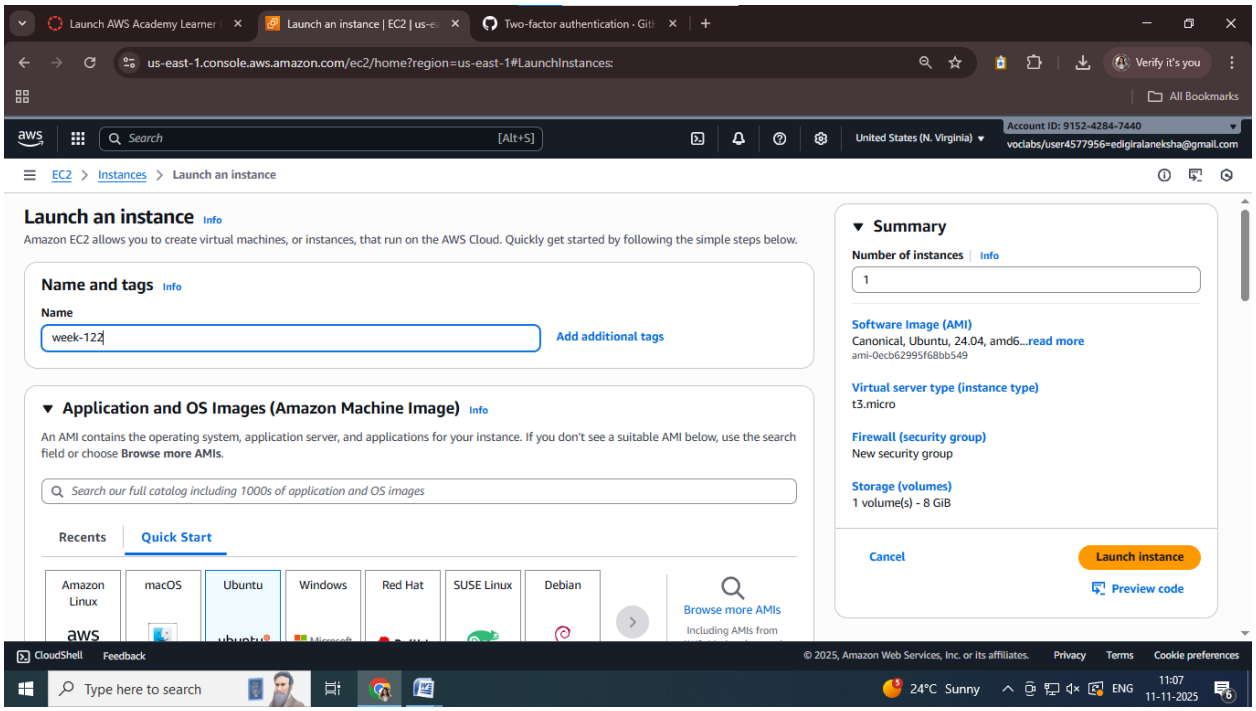
Step 4: click on AWS and in the services select EC2



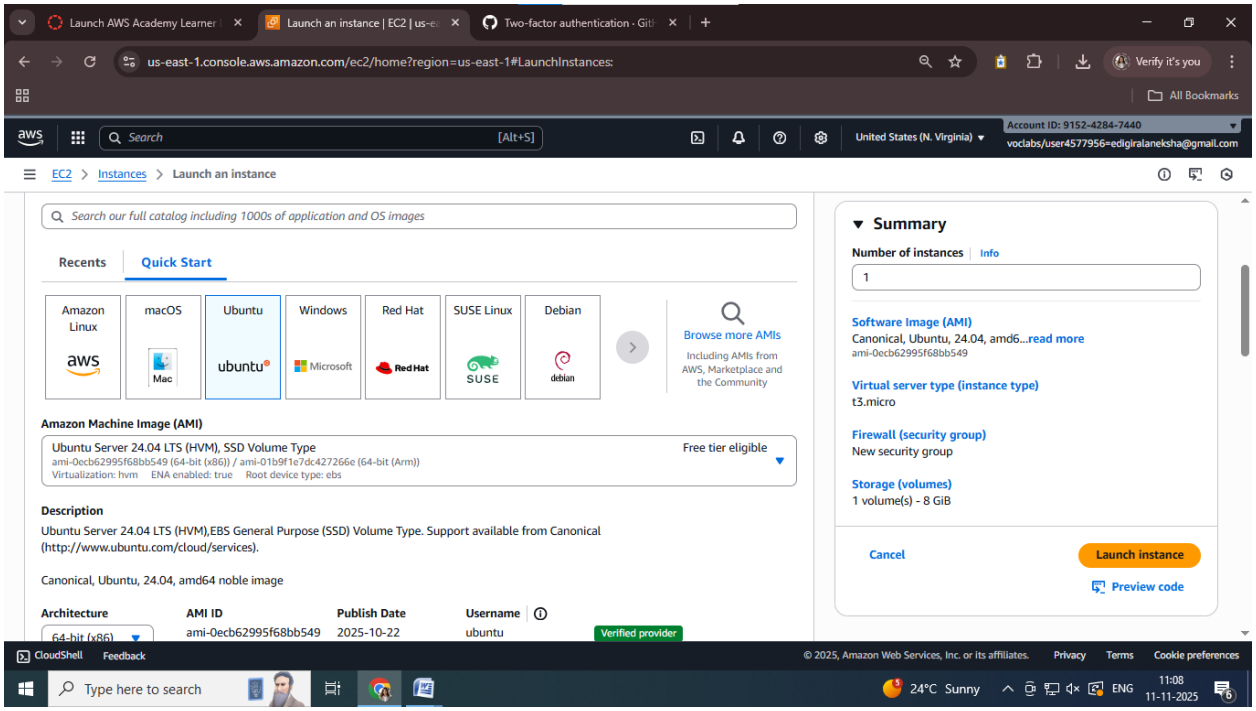
Step 5: select instances and select instance click on launch instance



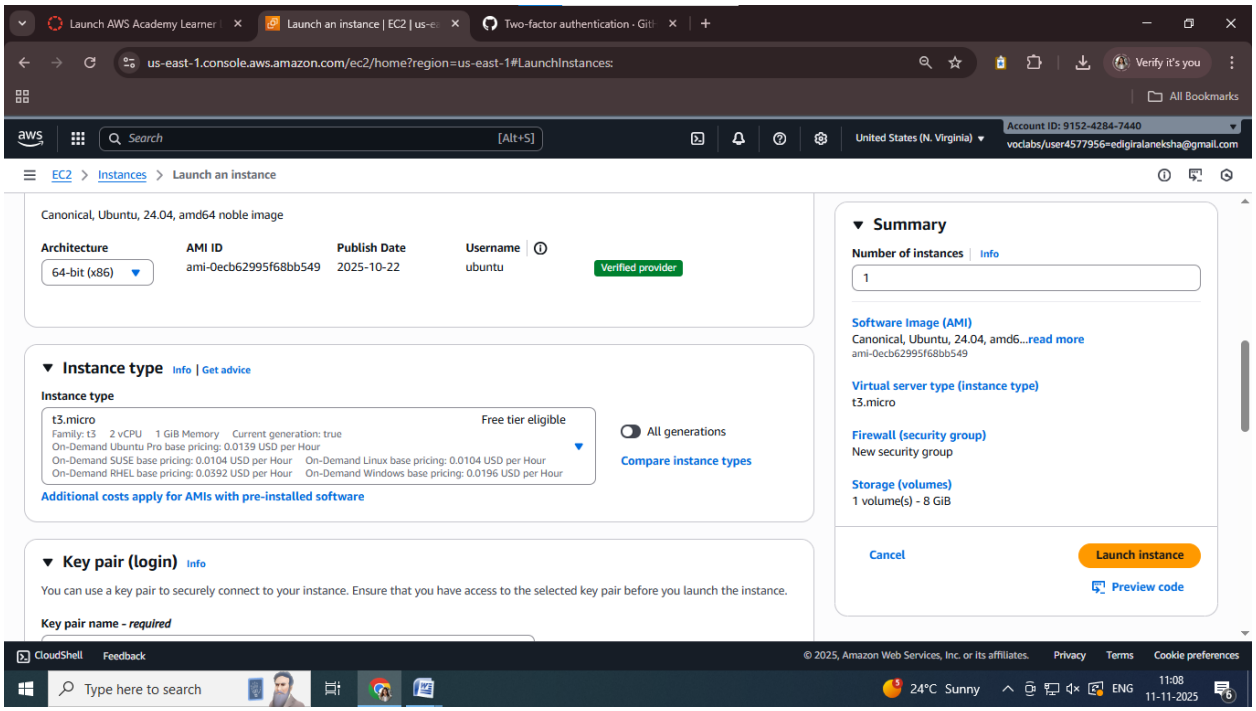
Step 6: Give the name of the machine “week-122”



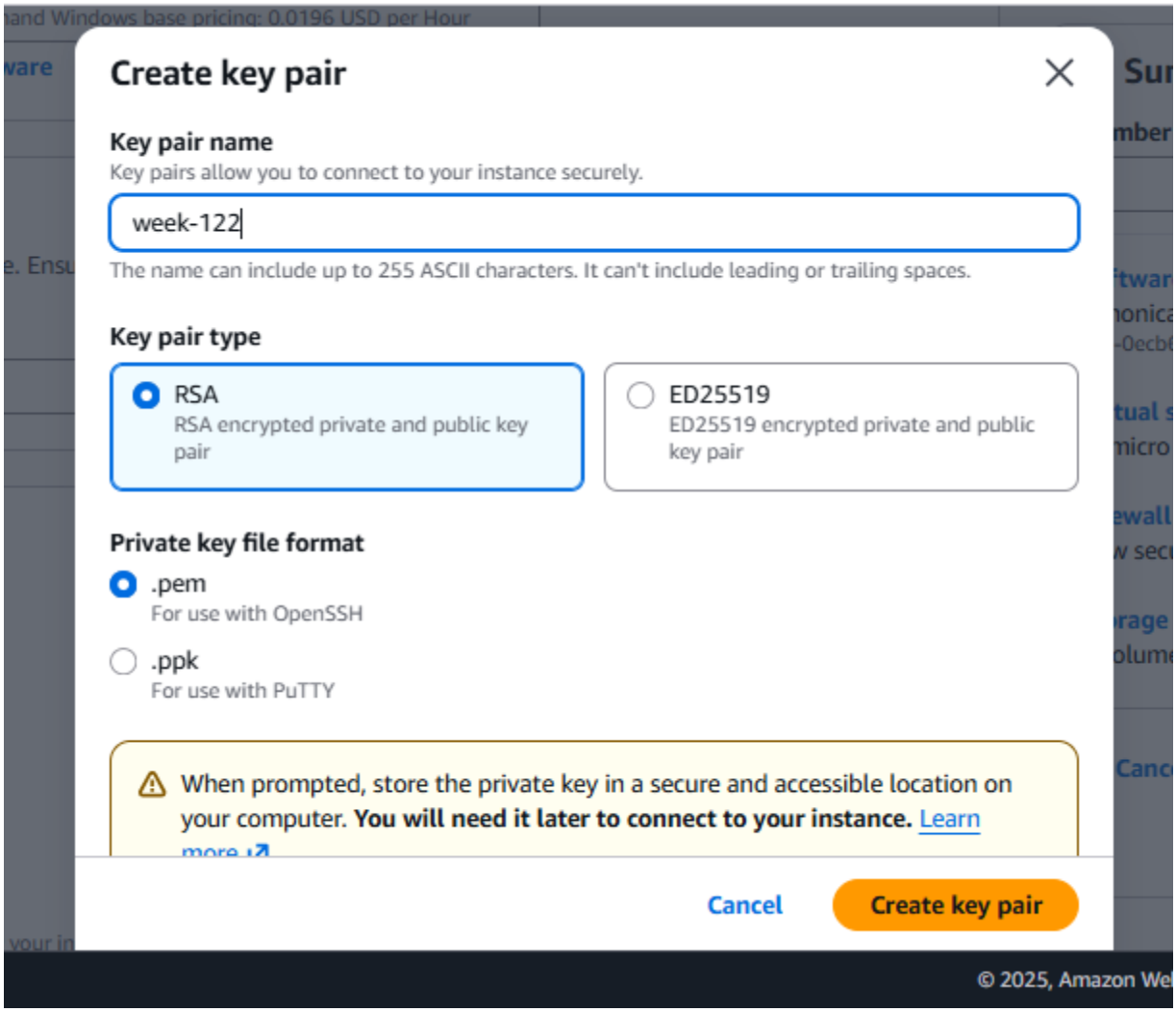
Step 6: Select the ubuntu server



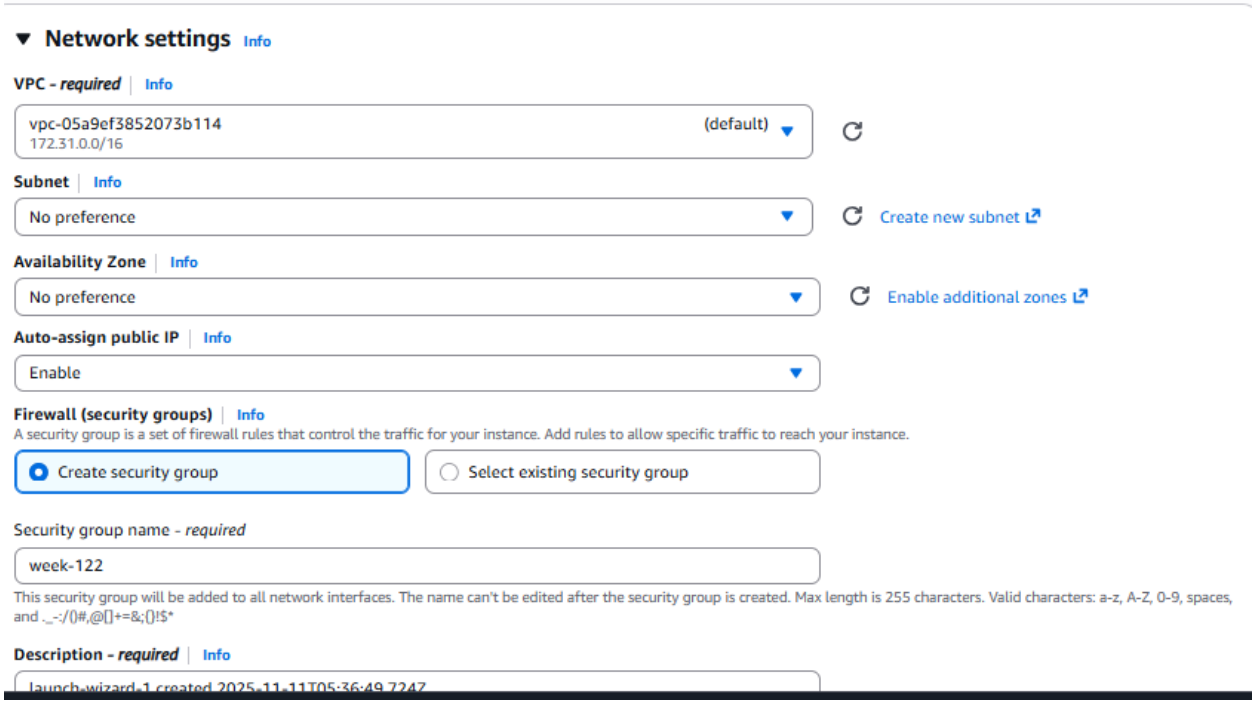
Step 7: select architecture as 64-bit and instance type as t3.micro(i.e., they are free)



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



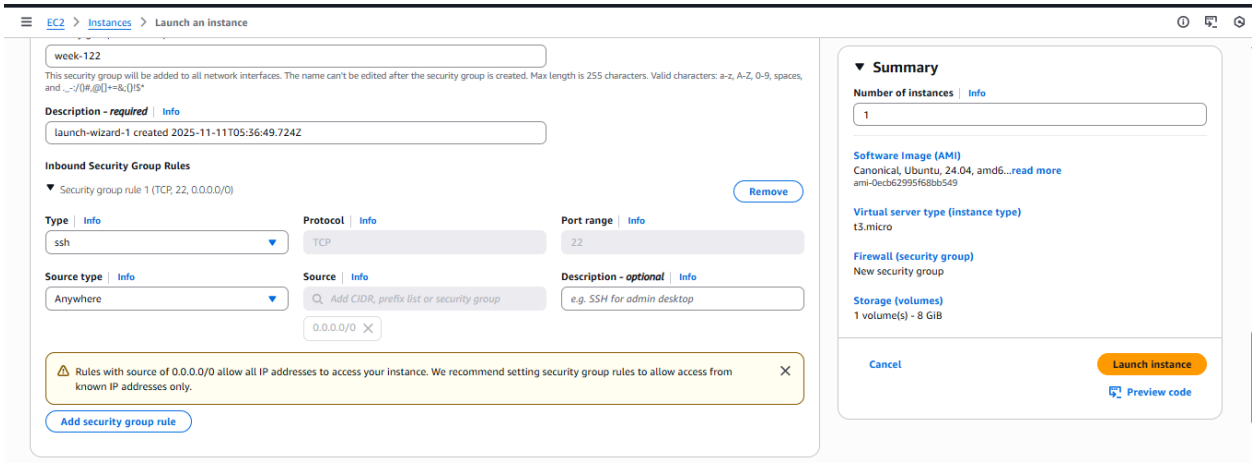
Step 9: In network settings select “create security group” and give the security group name



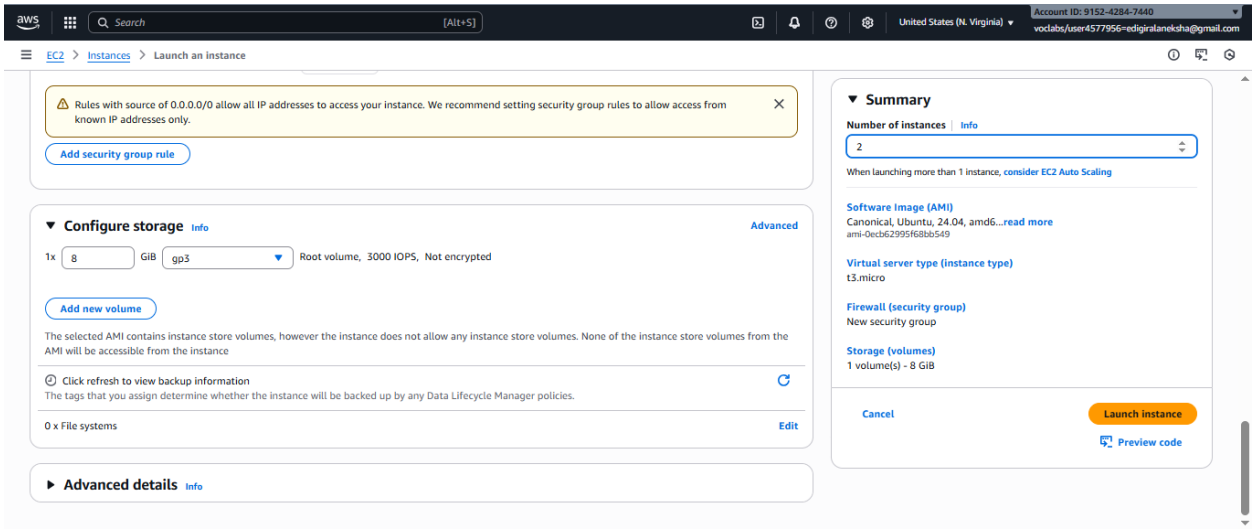
Step 10: Click on edit button on the top right corner and select

Type: ssh

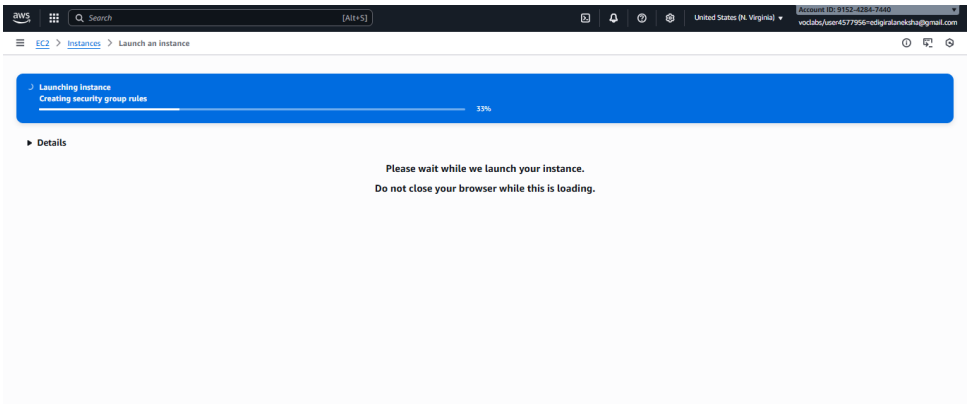
Source: anywhere

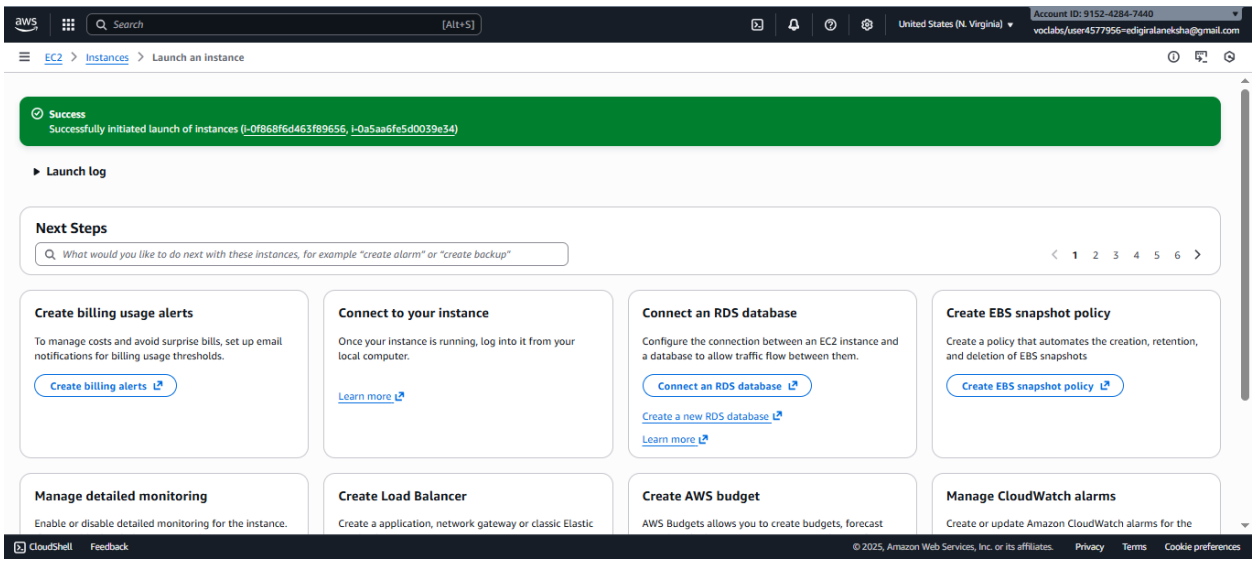


Step 11: in configure storage give 8GB and give number of instances as 2 and click on launch instance

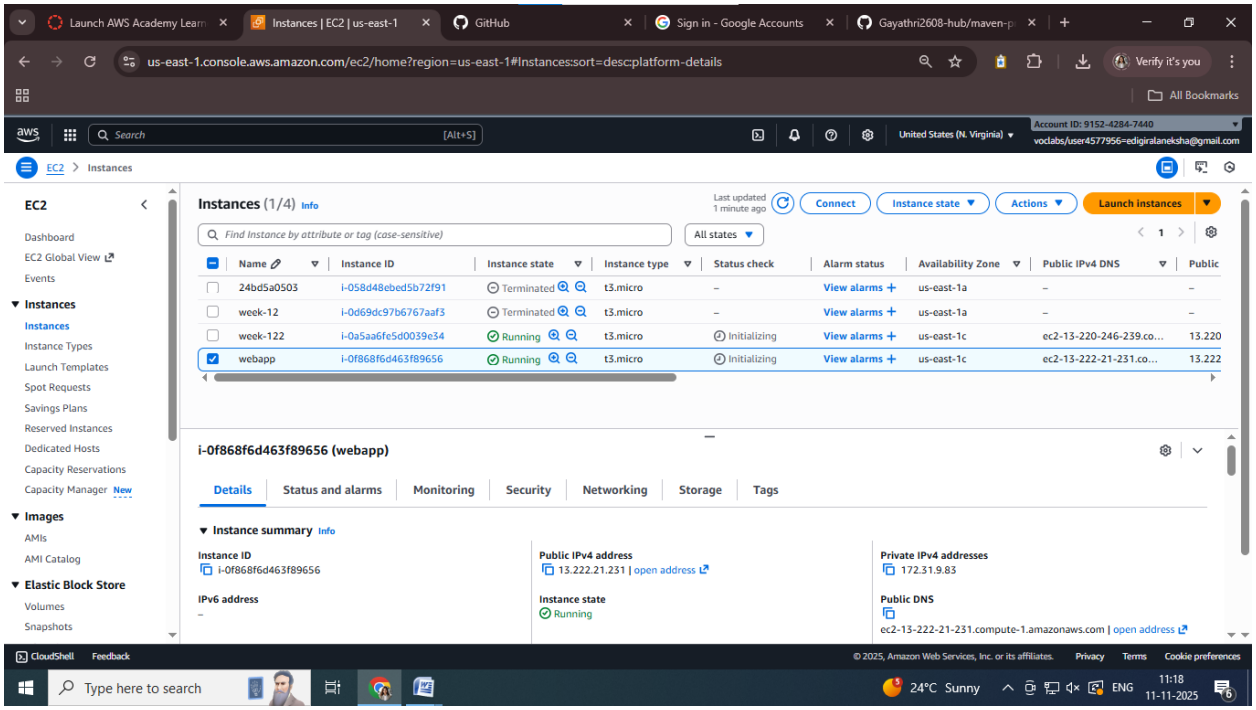


Step 12: The launching of instance will start and successful message will be shown

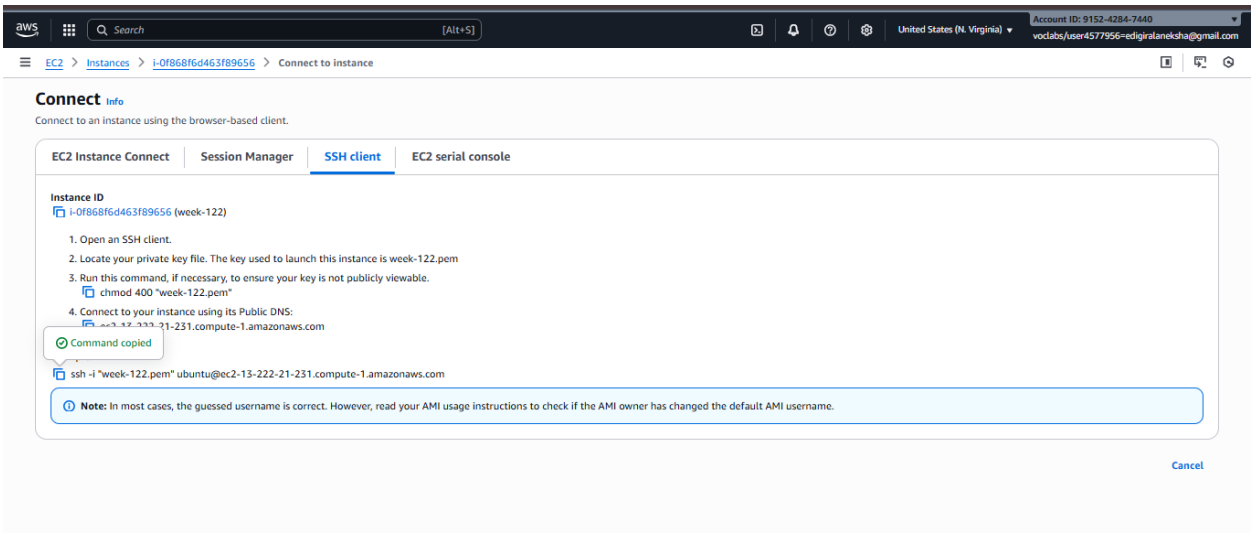




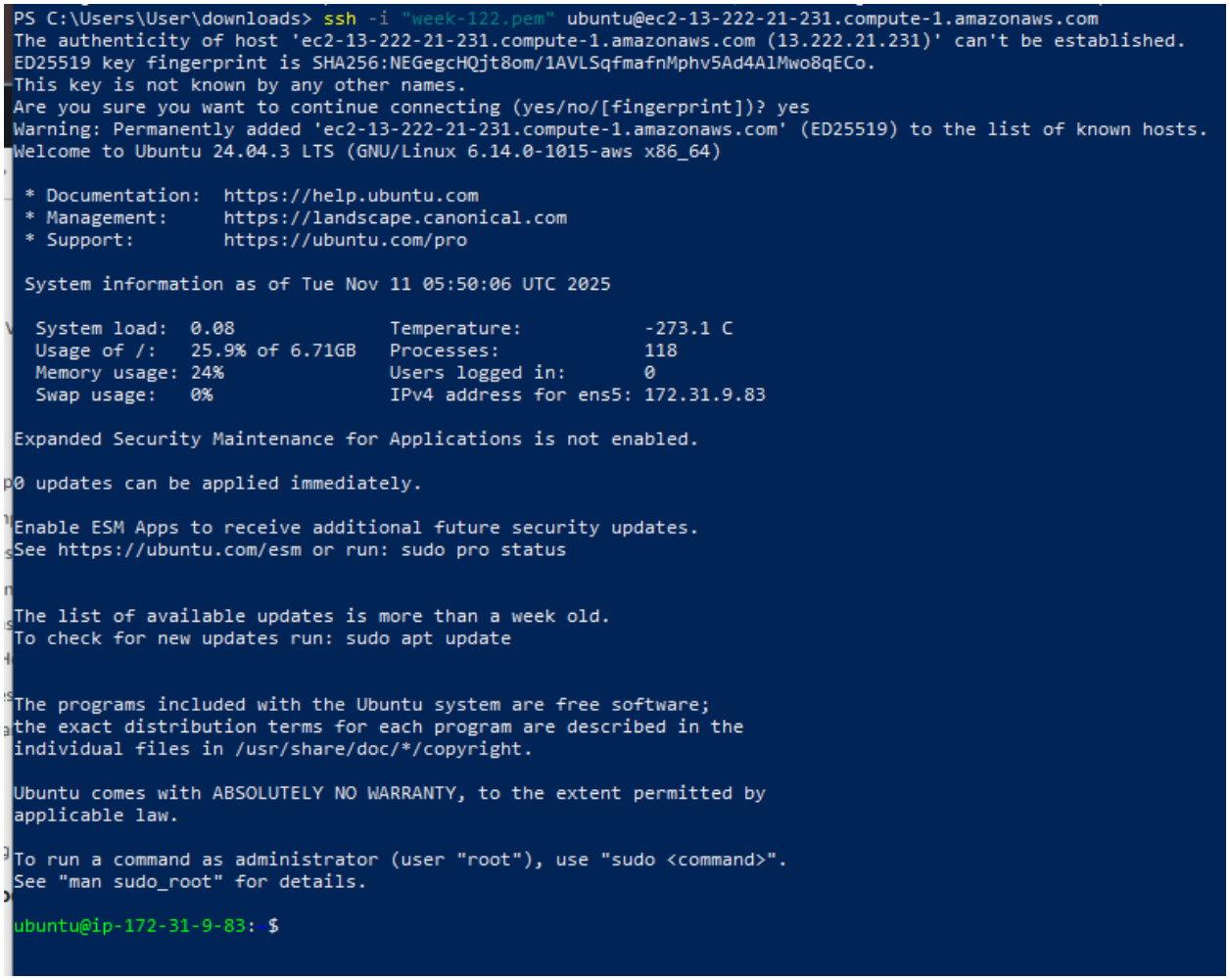
Step 13: In the instances the created ones will be shown, you can also rename the instance , changed week-1222 to “webapp”



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



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.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base pigz 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

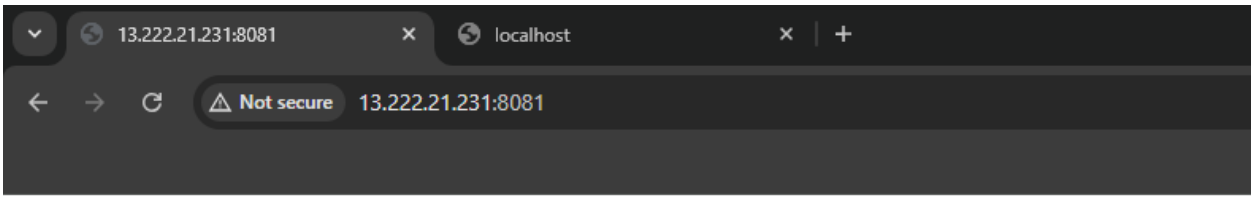
```
root@ip-172-31-9-83:/home/ubuntu/ar/maven-practice# sudo docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
img1         latest    a67a112ce8ac   2 minutes ago    413MB
dep1         latest    28efbe56e633   29 minutes ago   413MB
tomcat       9.0       2e4887a16e43   12 hours 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