R.V.R & J.C College of Engineering (Autonomous)

Regd.No: L23CD216 **CSE(Data Science)**

Experiment-1: Demonstration of Jira Tool

Aim: To demonstrate Agile Scrum activities using the Jira tool.

Jira is the #1 agile project management tool used by teams to plan, track, release and support world-class software with confidence.

Jira software is an agile project management tool developed by Atlassian for tracking and managing software development projects, bugs, issues, and tasks

Jira is a software product developed by Atlassian that allows bug tracking, issue tracking and agile project management. Jira is used by a large number of clients and users globally for project, time, requirements, task, bug, change, code, test, release, sprint management.

Before Zira

No centralized tracking – teams used scattered tools like emails and spreadsheets.

Poor collaboration – hard to share updates and assign responsibilities.

Lack of agile support – no built-in Scrum, Kanban, or sprint management

After Zira

Centralized tracking – All tasks, bugs, and projects are managed in one place.

Flexible workflows – Teams can customize, automate, and adapt processes easily.

Agile support & reporting – Built-in Scrum/Kanban boards, sprint tracking, and real-time dashboards give full visibility.

Uses & Benfits of Zira

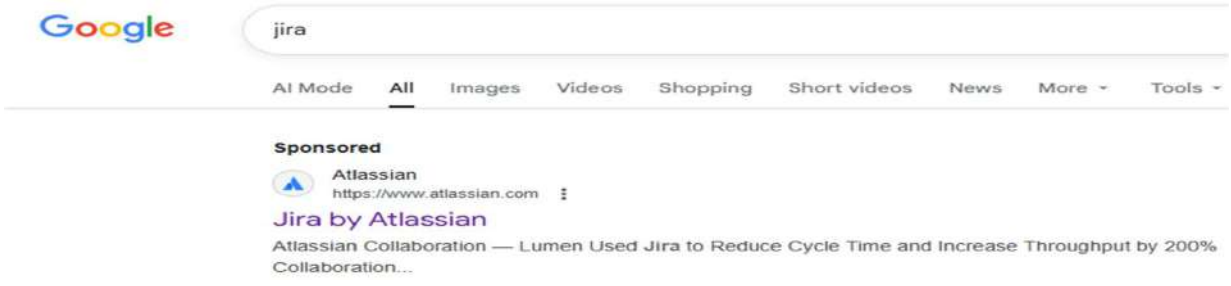
Improved collaboration – Team members can comment, assign tasks, and share updates seamlessly.

Prioritization & transparency – Helps prioritize tasks and gives stakeholders clear visibility of progress.

Integration with other tools – Connects easily with Confluence, Bitbucket, Slack, Git, and more.

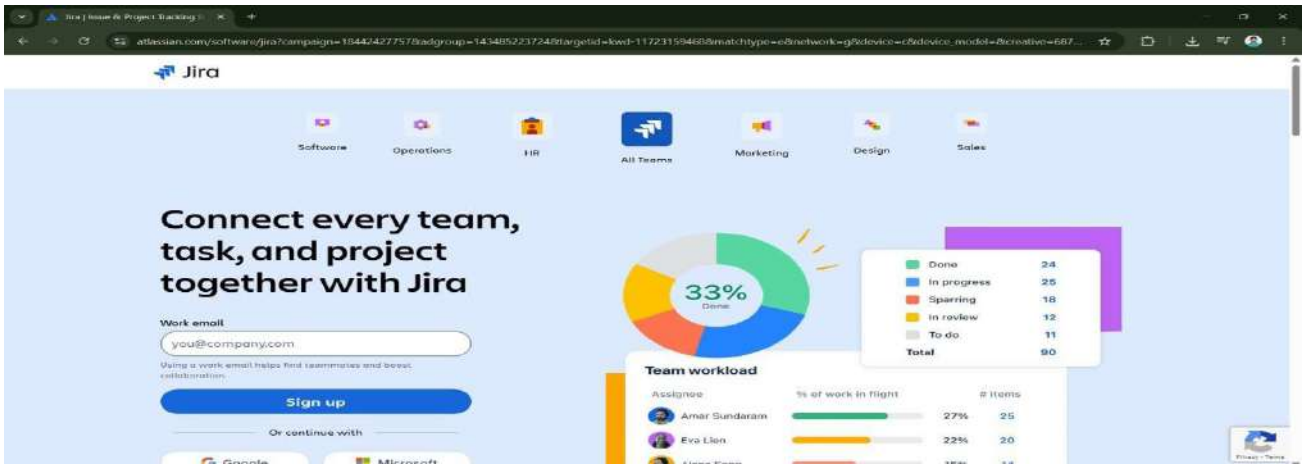
Create your free Jira Account

Step by Step Process:

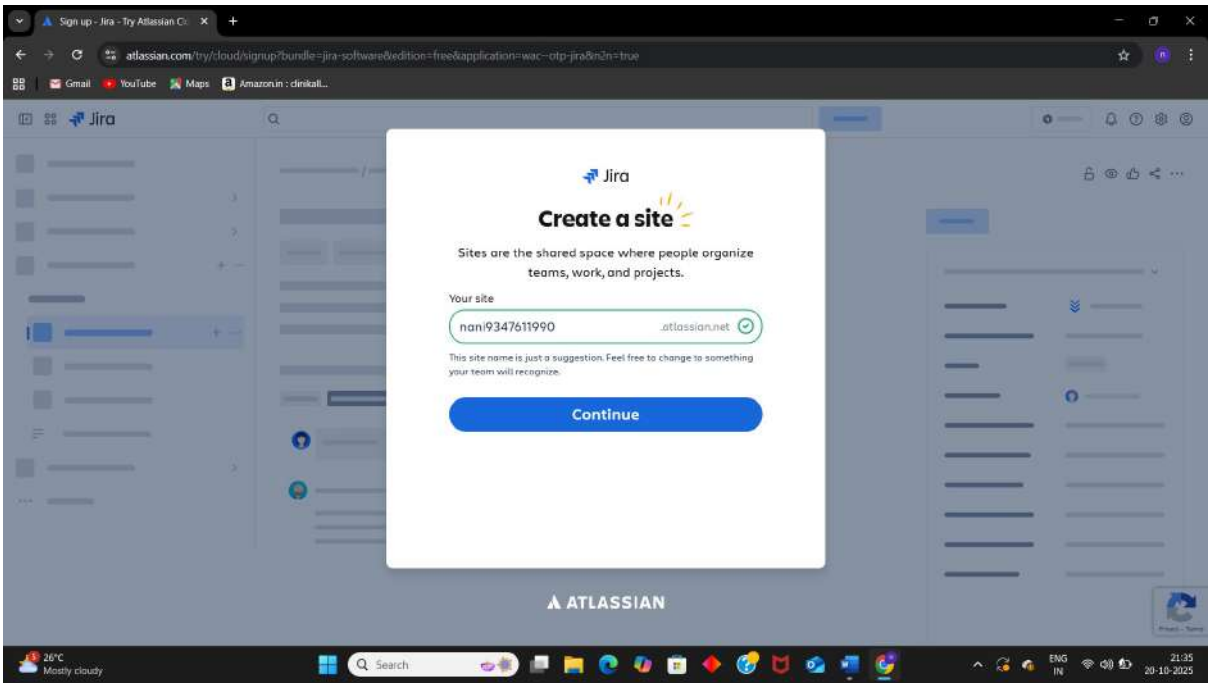


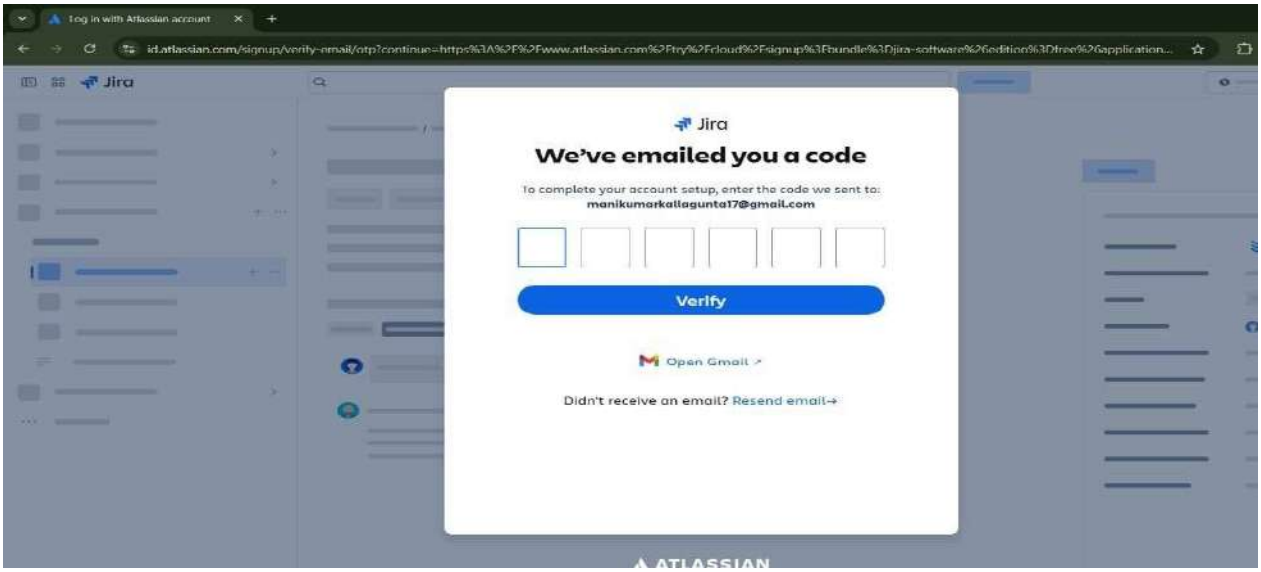
https://www.atlassian.com/software/jira

1)Sign Up: Click on Sign Up

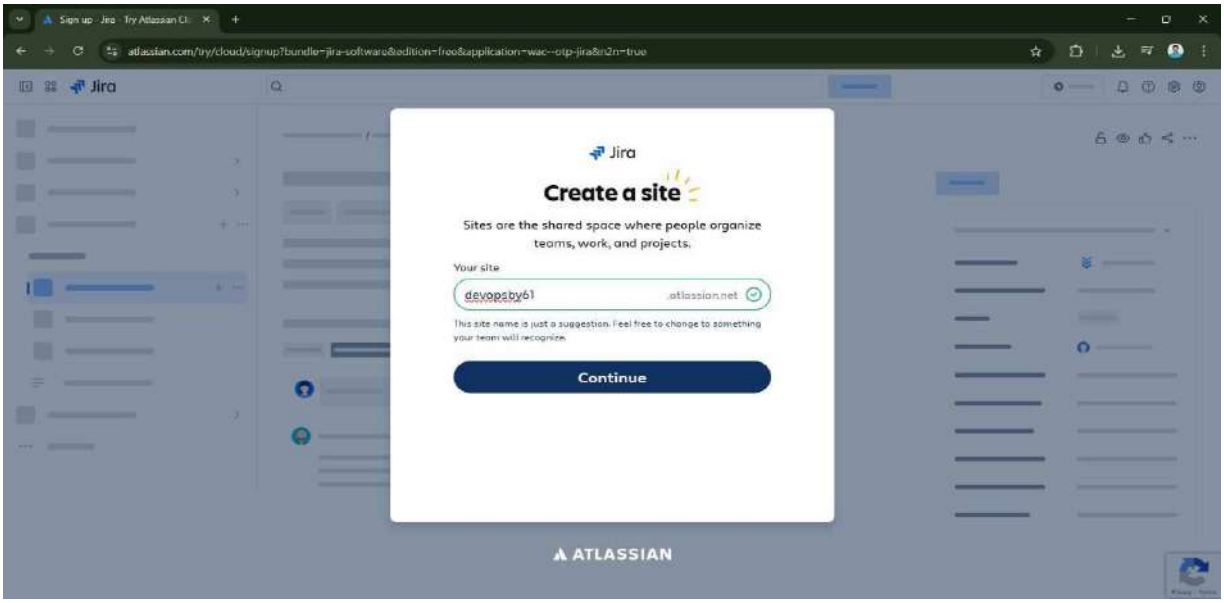


Enter The Mail id :





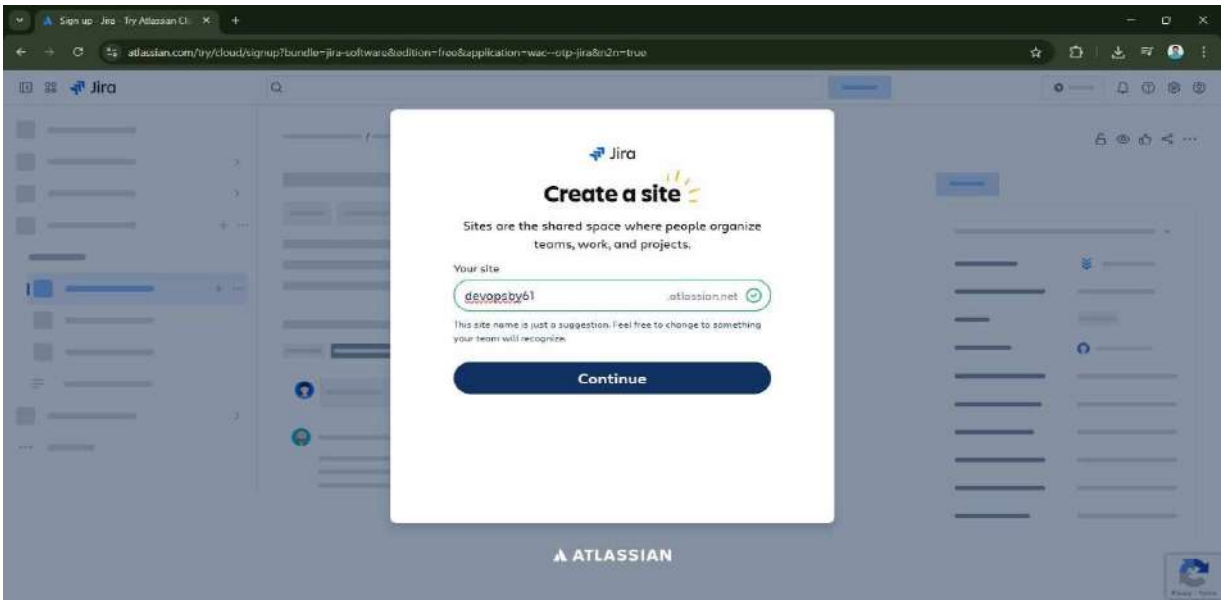
Set The new Password & Click on Continue

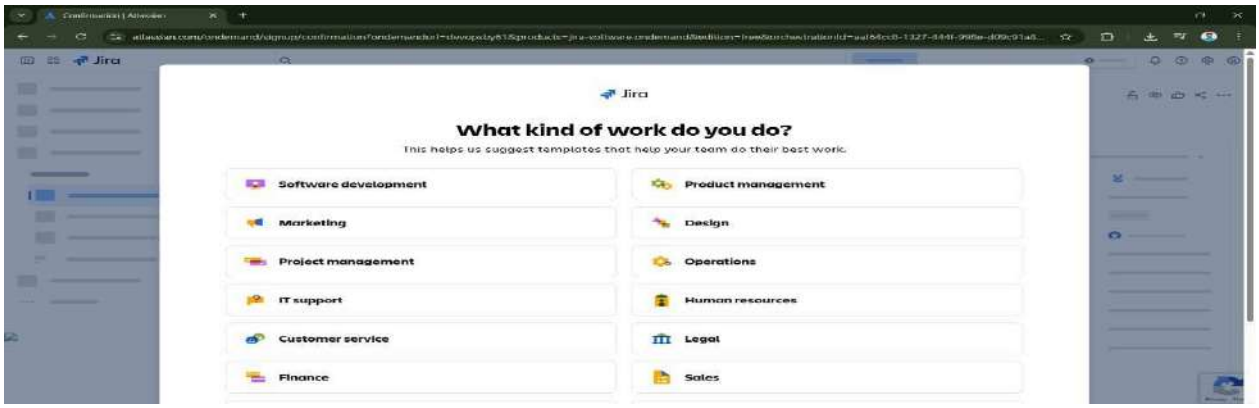


Creating Site :

Enter The Site Name &

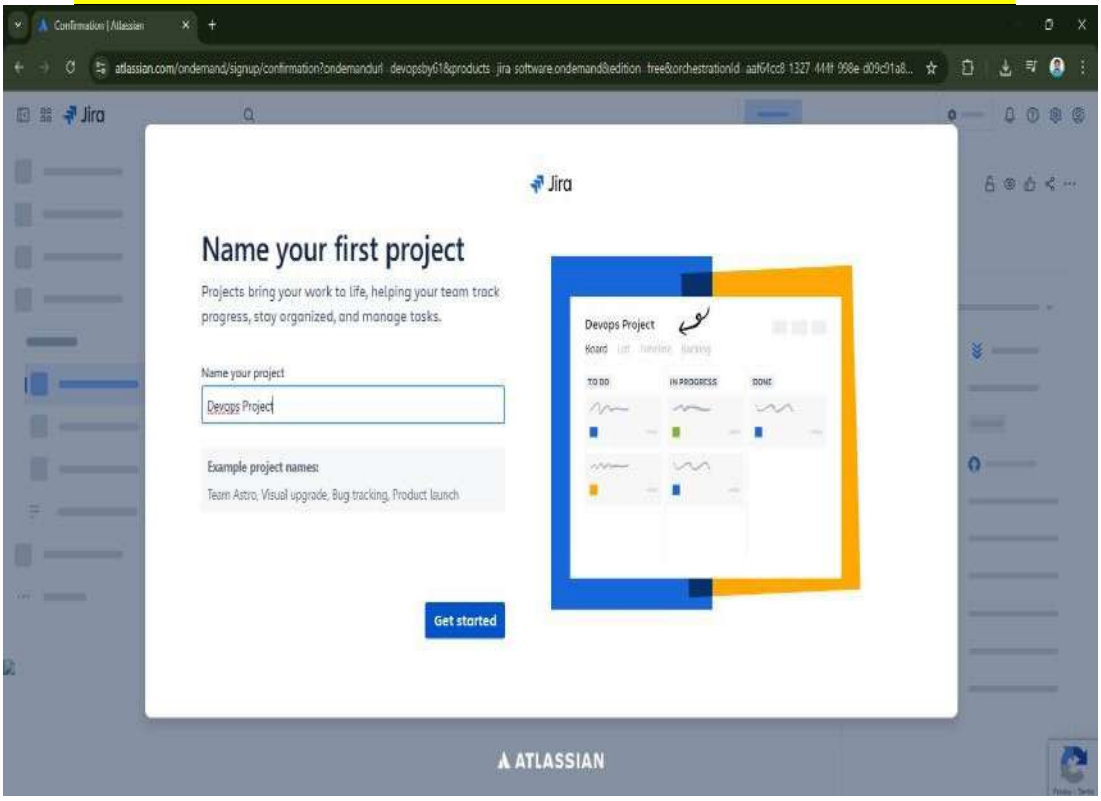
Continue

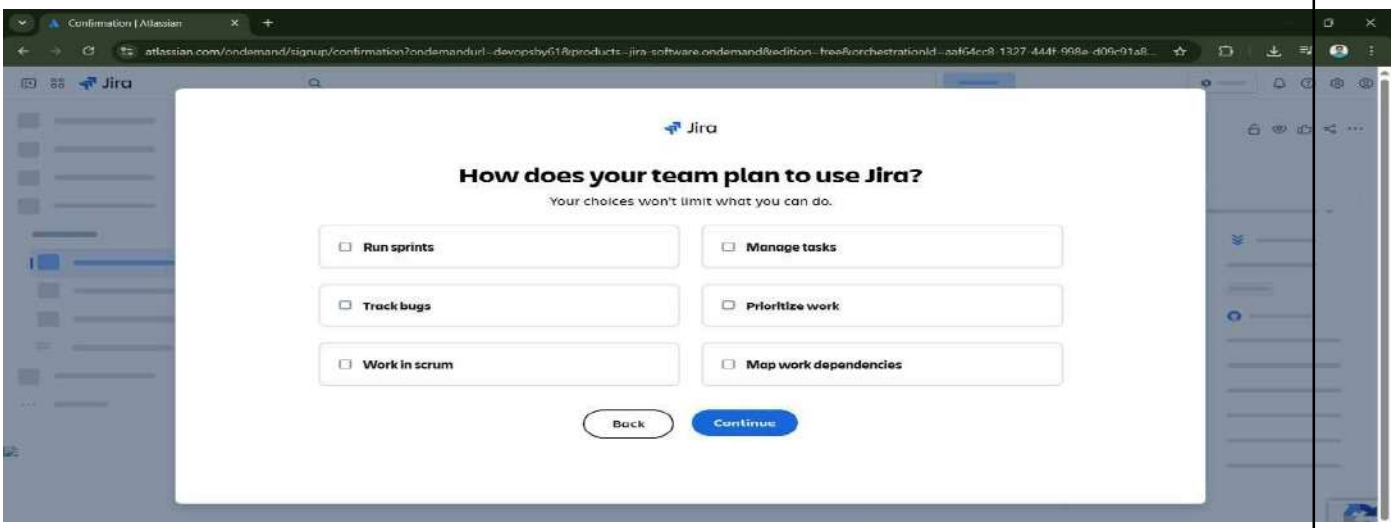


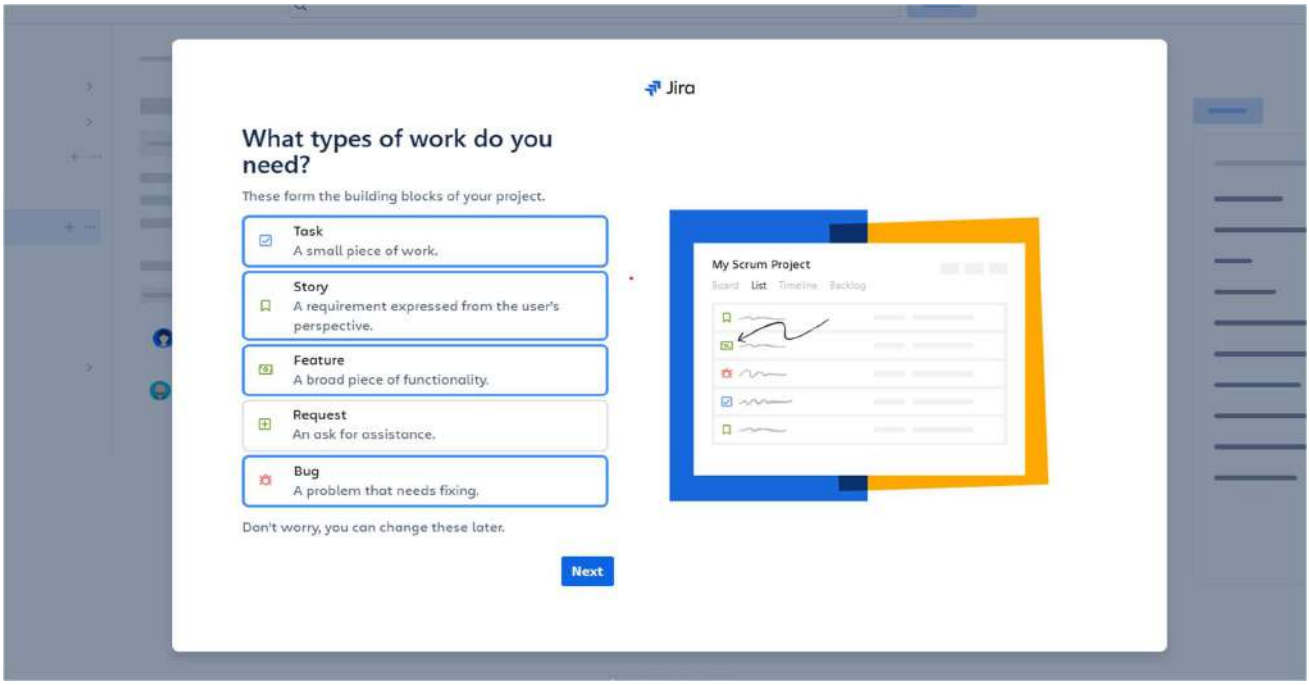


Select the Software Development Option

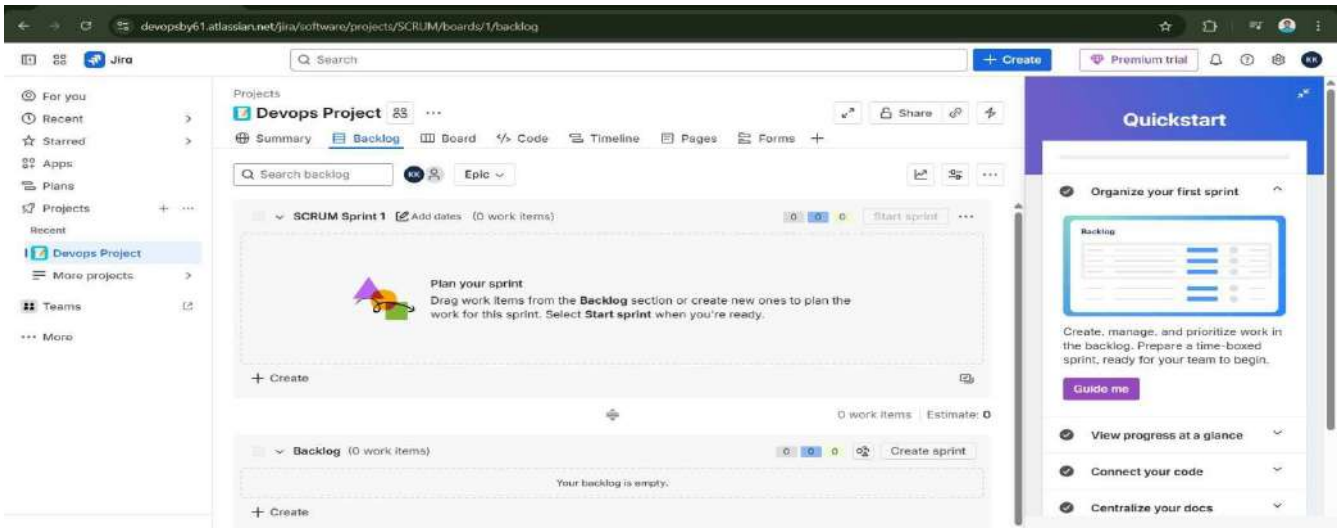
Create your first Project: Enter The Project Name & Then



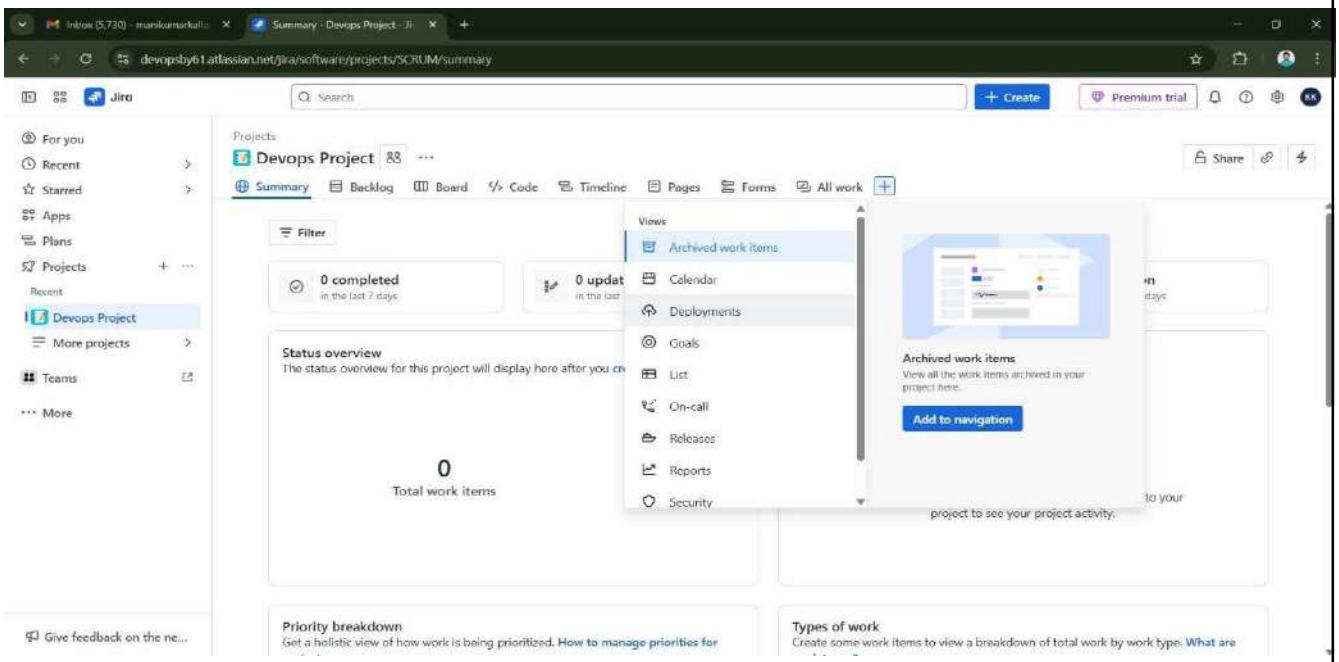




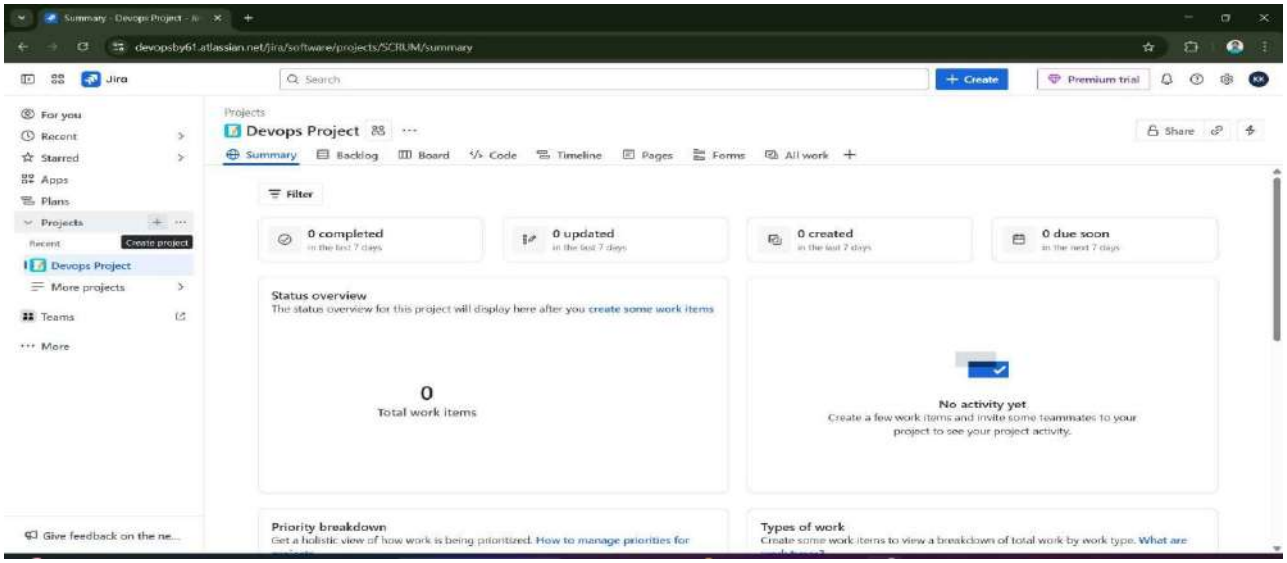
Select The Task,Story,Feature,Bug and Click on Next



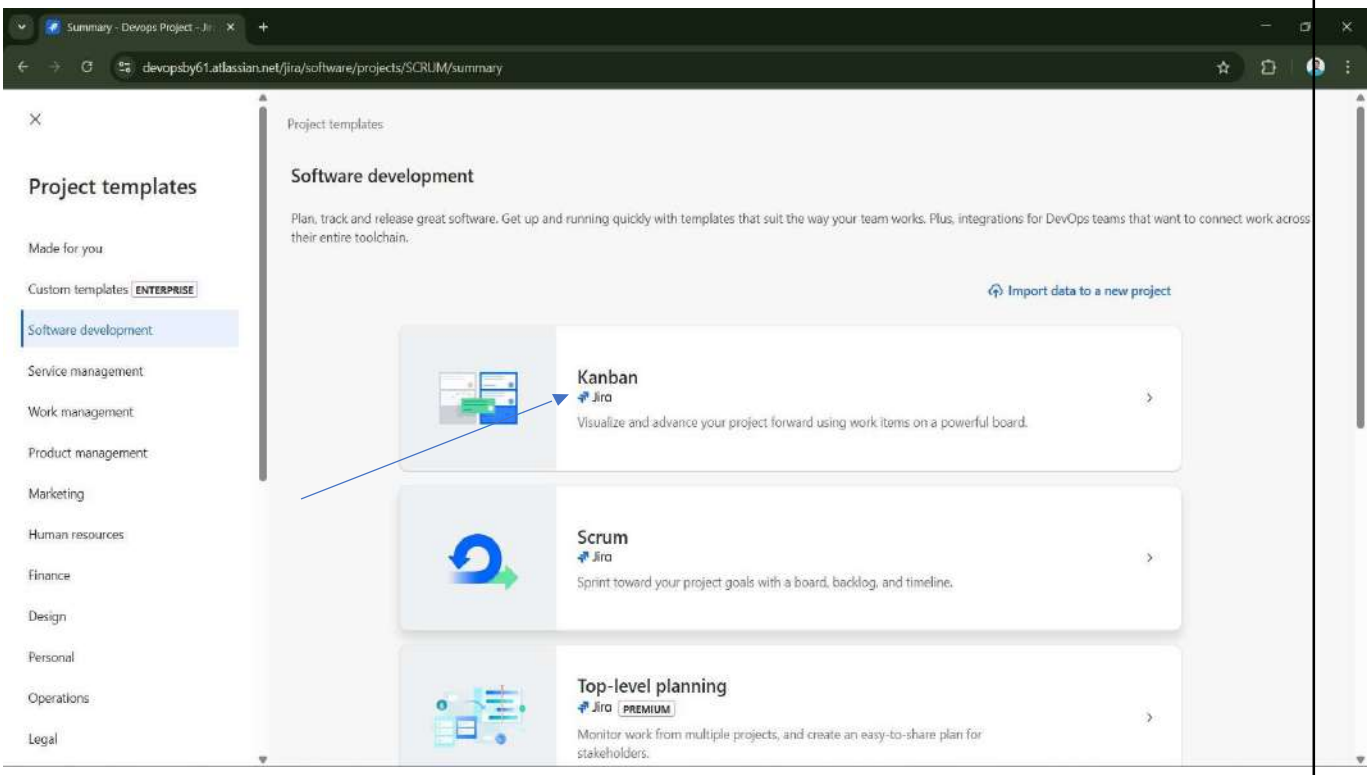
This is the Dashboard of Zira a default project done by us ex: Devops Project

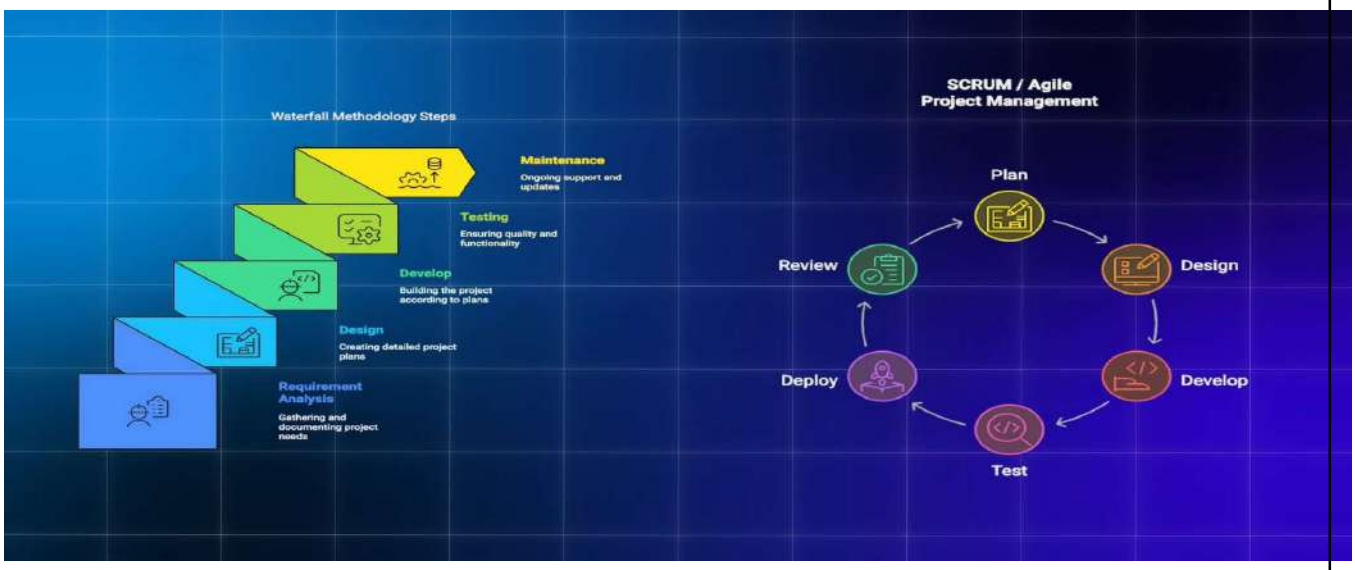


Here We are going to See what the list of items we are going to Work Ex: Backlog,Board, Calender,List,Reports etc



Here is the Overview of the Jira Dashboard that Shows Everything to user where all the work that is visible on the main Dashboard





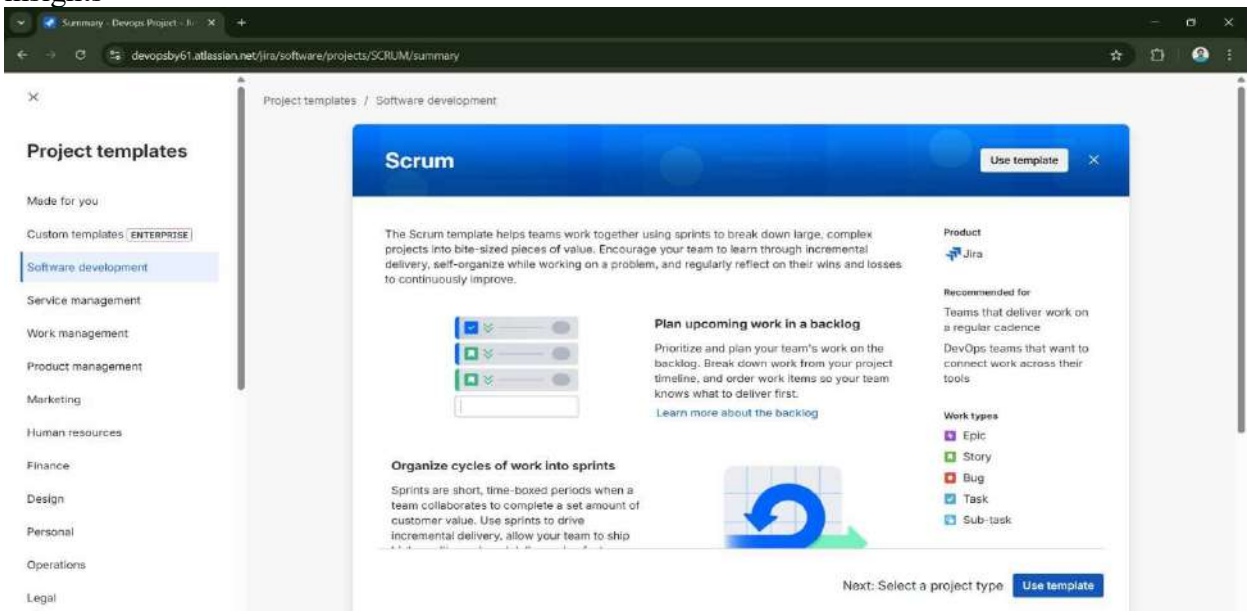
choose a template (Scrum, Kanban, or basic) and set visibility.
Select a project template (Scrum/Kanban/Team-managed), give it a name and key, then set visibility and defaults so your team can start adding issues.

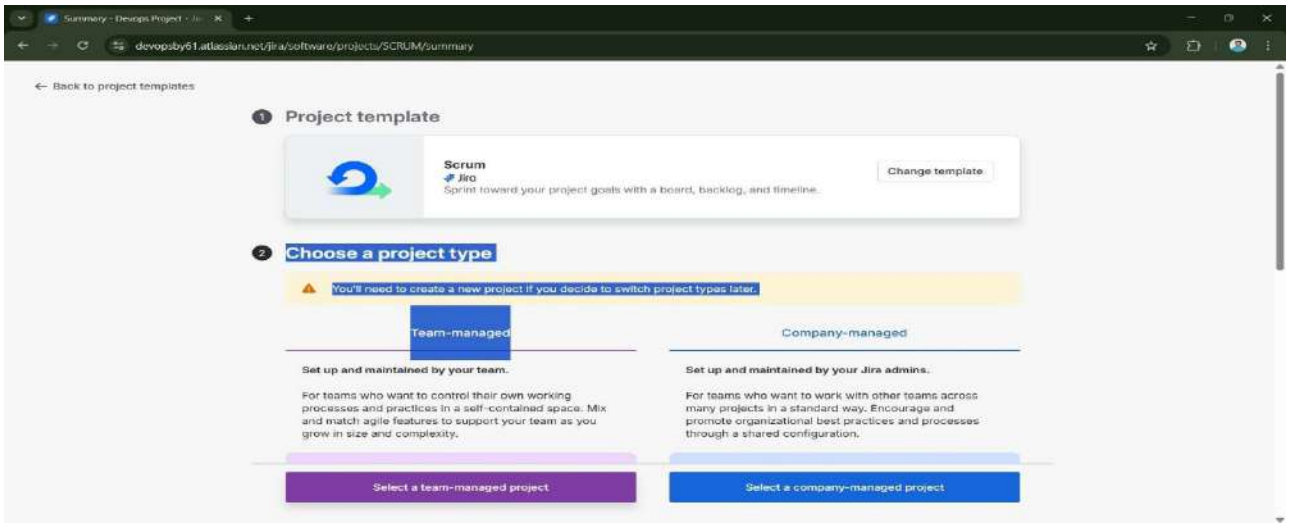
Here we Choose Scrum :

Scrum Dashboard — provides a real-time overview of sprint progress, assigned issues, and team performance metrics.

In a Scrum project, the Jira dashboard displays widgets that summarize sprint activity — such as

assigned issues, burndown trends, and team velocity.
These gadgets help Scrum Masters and team members track sprint health, identify blockers, and ensure the team remains on schedule to meet sprint goals.
It acts as a central hub for daily stand-ups and sprint review insights

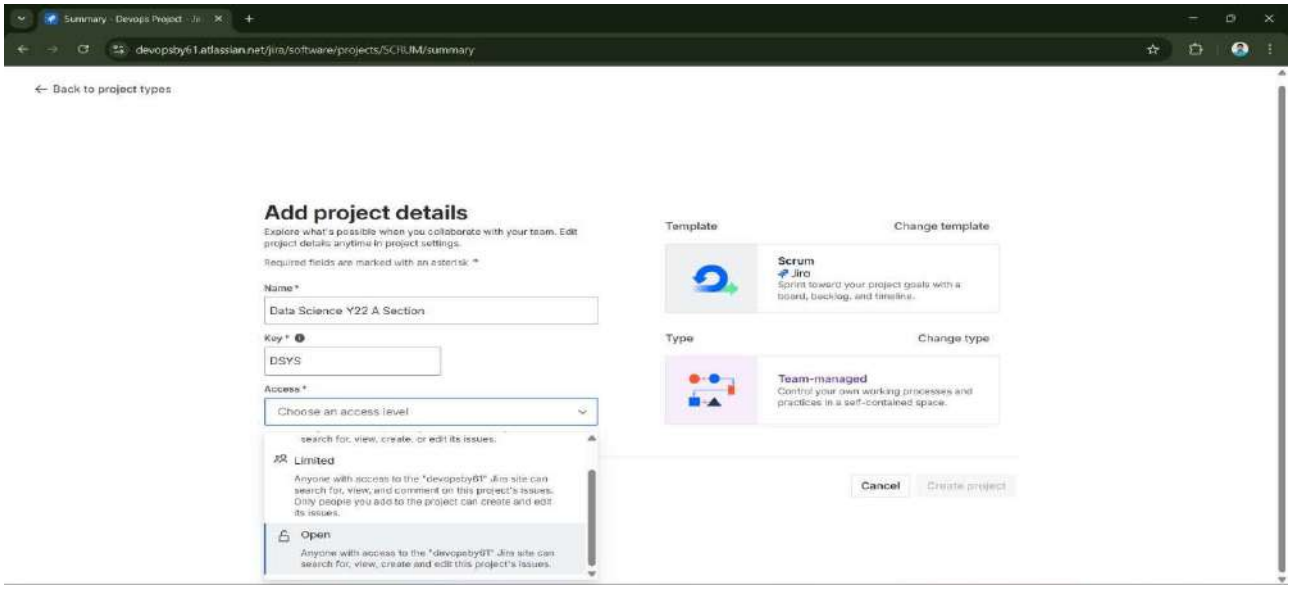




Selecting a project type in Jira Scrum — choose between Team-managed and Company-managed projects.

Team-managed projects are ideal for small, independent teams who want flexibility and control over their workflows. Each team can customize their boards, issue types, and permissions without affecting others.

Company-managed projects are better for larger organizations or multiple teams that need consistent processes and shared configurations set by Jira administrators

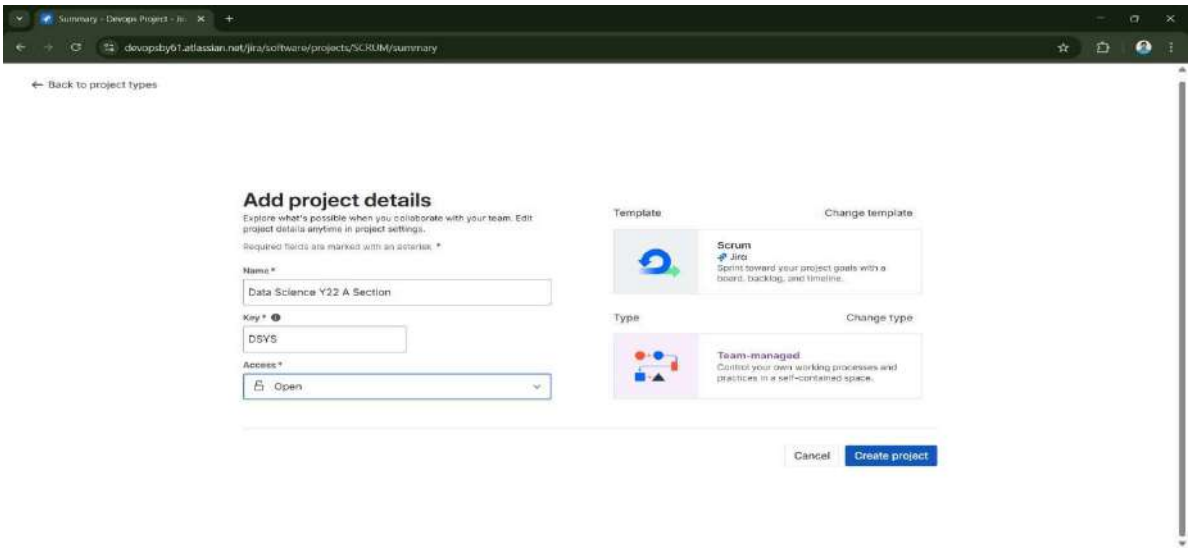


Name field: "Data Science Y22 A Section" — represents the project's title.

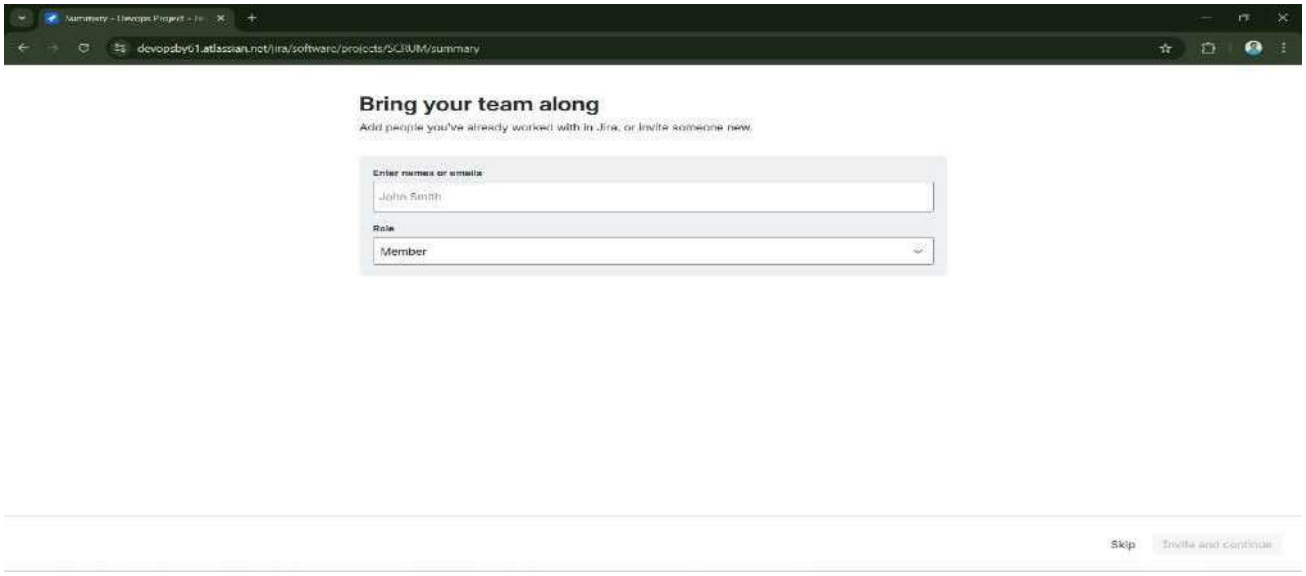
Key field: "DSYS" — automatically generated abbreviation used as a prefix for issue IDs (e.g., DSYS-1, DSYS-2).

Access level options: Dropdown showing "Limited" and "Open" — determines who can view or edit issues in this project.

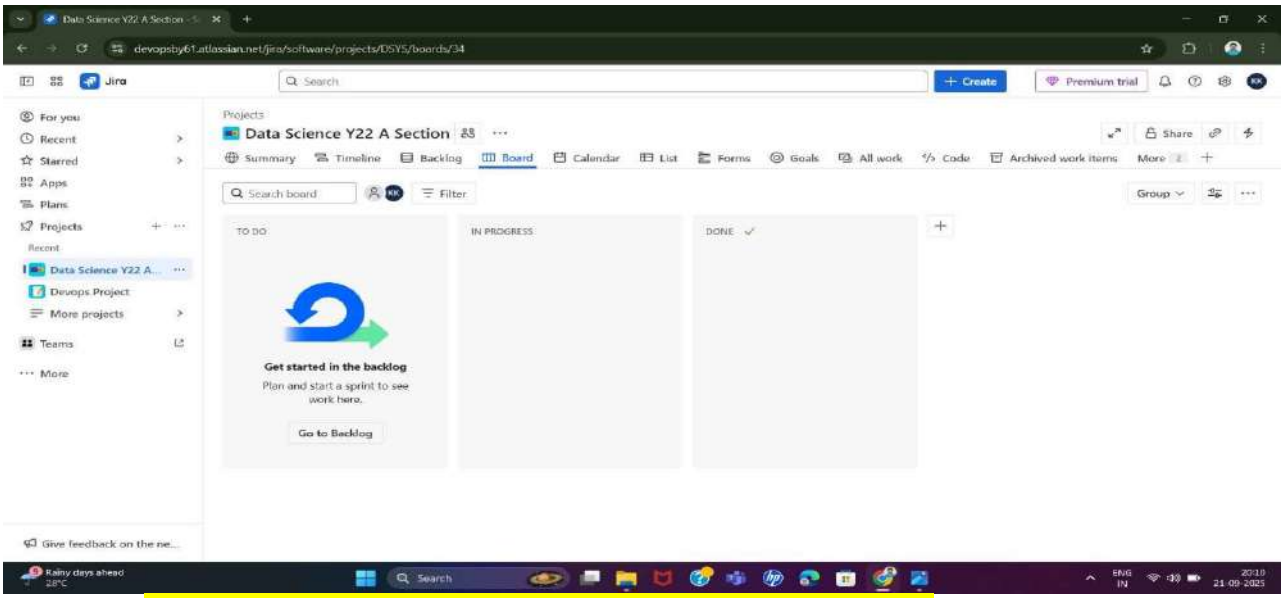
Template and Type sections: Confirm that the selected template is Scrum and the project type is Team-managed.



Invite Team members :



Invite team members — add users by email and set roles (Admin, Developer, Viewer).Add teammates by email and assign their role (project admin, developer, viewer). Invites let new users join the site and access projects.



Jira Dashboard explained (For You, Dashboards, Goals, etc.)

DashBoard — visual workflow to manage sprint tasks across “To Do,” “In Progress,” and “Done” columns.

Project Name: Data Science Y22 A Section (appears at the top).
Navigation Tabs: Summary, Timeline, Backlog, Board, Calendar, List, Forms, Goals, Code, etc.
Board Columns: Default Scrum workflow — To Do, In Progress, Done.
Message: “Get started in the backlog” — indicates that tasks must first be added in the backlog to appear here during sprints.
Goals tab (highlighted): Lets teams track sprint objectives and align work with broader goals.

This screen displays the Scrum Board for the project Data Science Y22 A Section.
The board provides a visual representation of the team's sprint workflow, divided into three main columns:

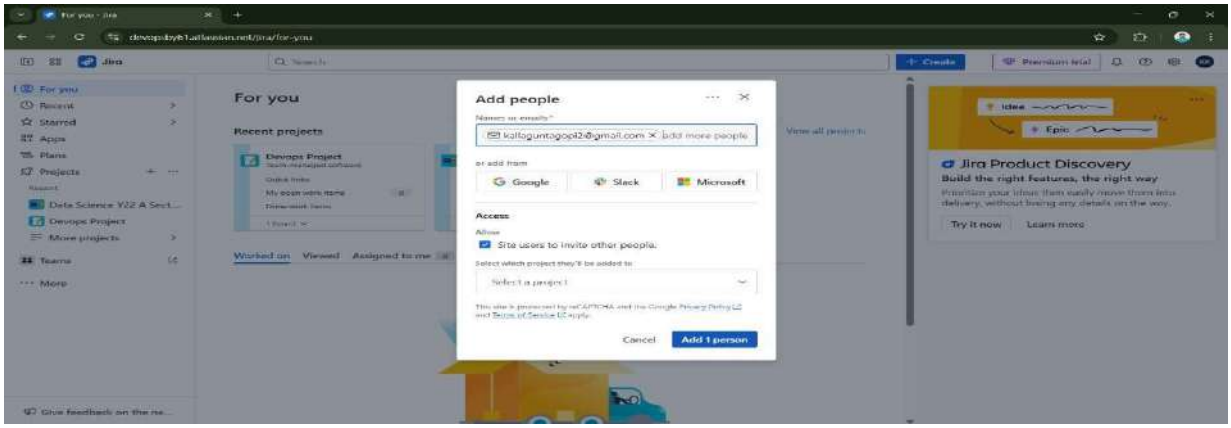
- **To Do:** Newly created tasks not yet started.
- **In Progress:** Tasks currently being worked on.
- **Done:** Completed tasks.

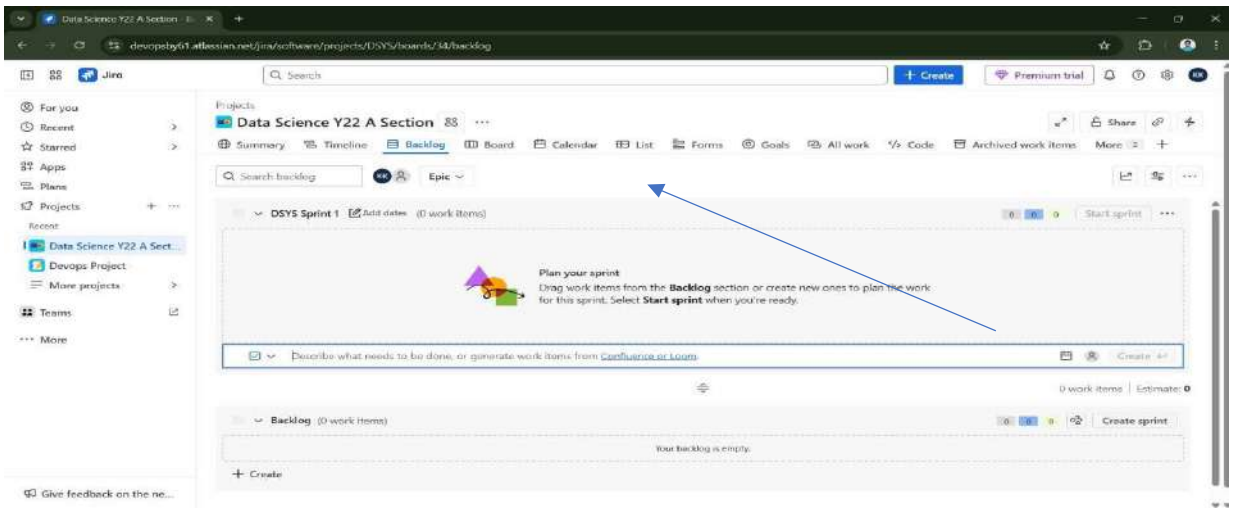
The "Get started in the backlog" message indicates that the sprint hasn't started yet — tasks must first be added in the backlog.

The navigation bar at the top allows quick access to features like Timeline (for sprint planning), Goals (for tracking sprint objectives), and Code (for linking repositories).

Once issues are added to the backlog and a sprint is started, tasks automatically appear on this board for tracking daily progress.

Add People

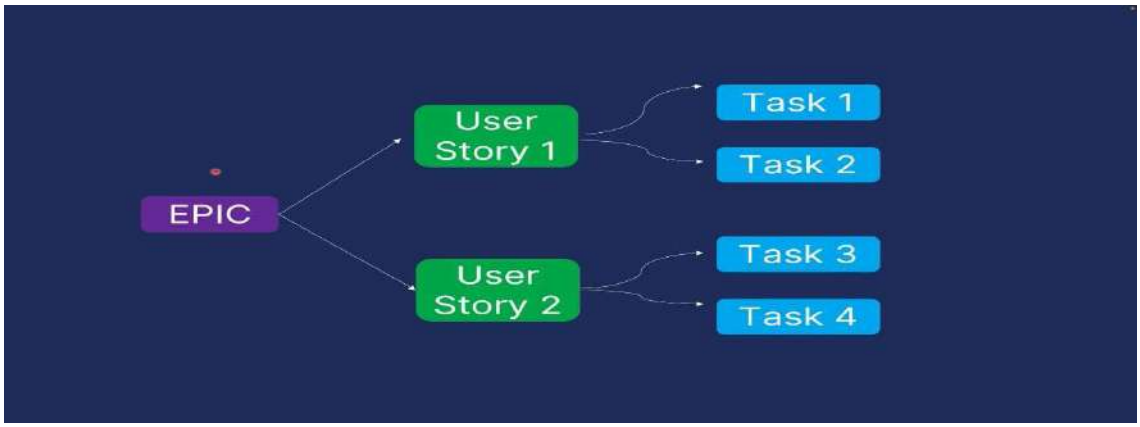


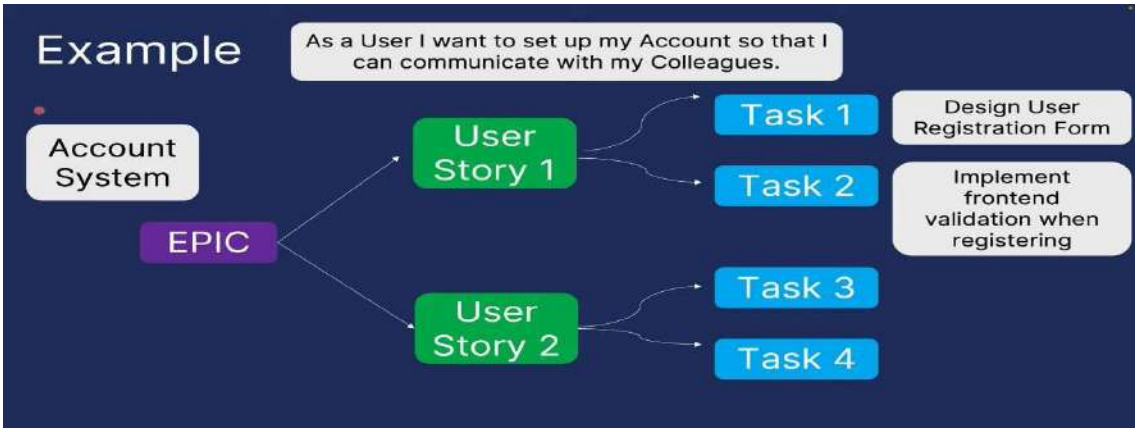


Issue Types explained

Zira has 3 different IssueTypes

Epic: Big Abstract View of Work which then gets broken down in several smaller tasks called User Stories

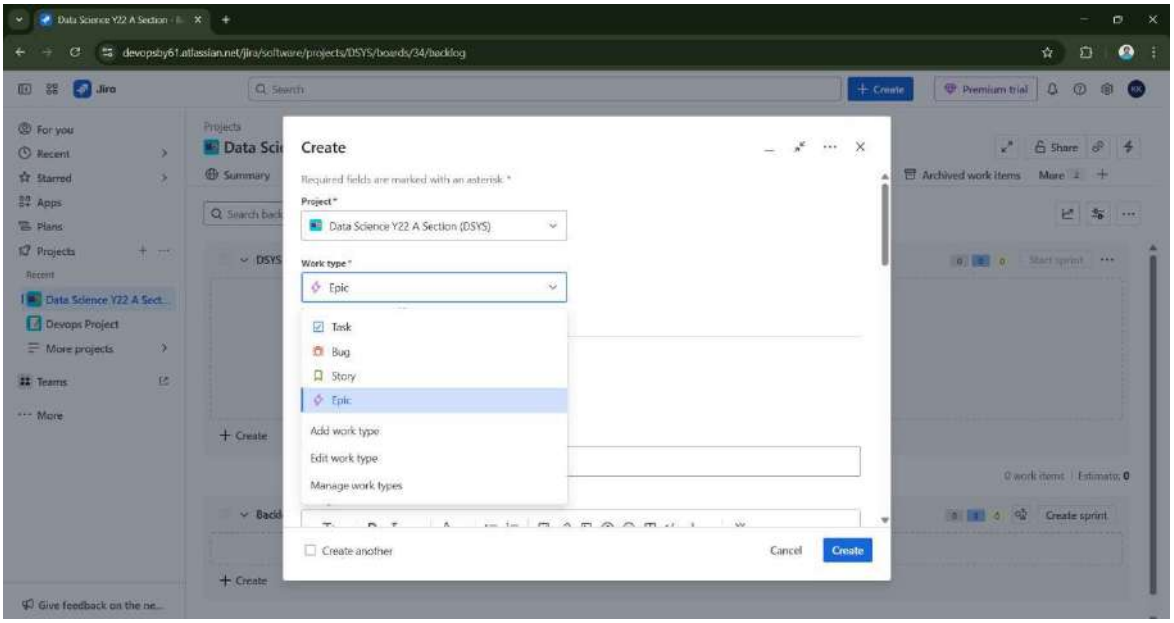


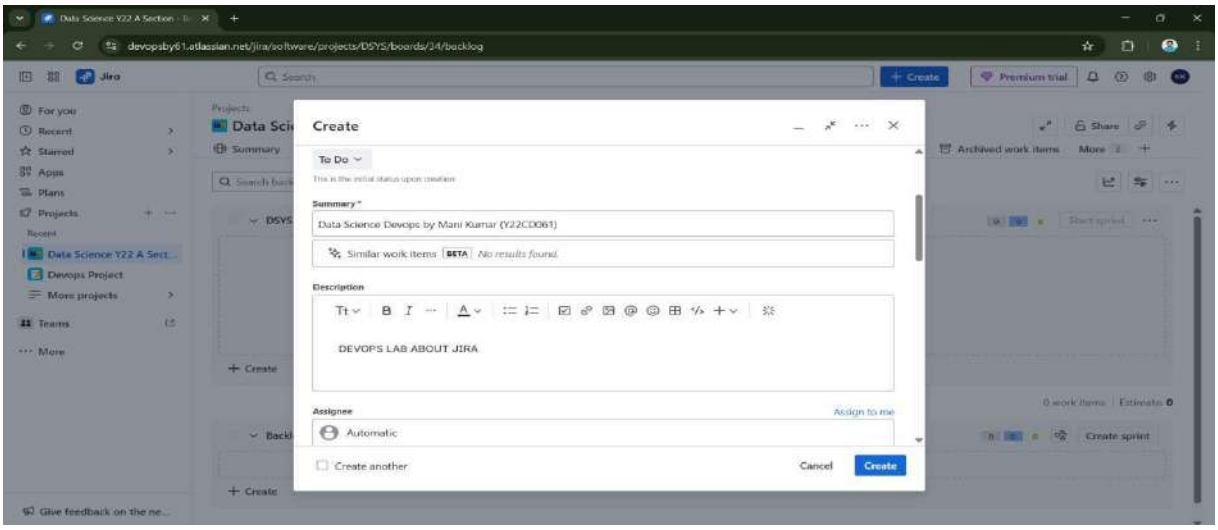


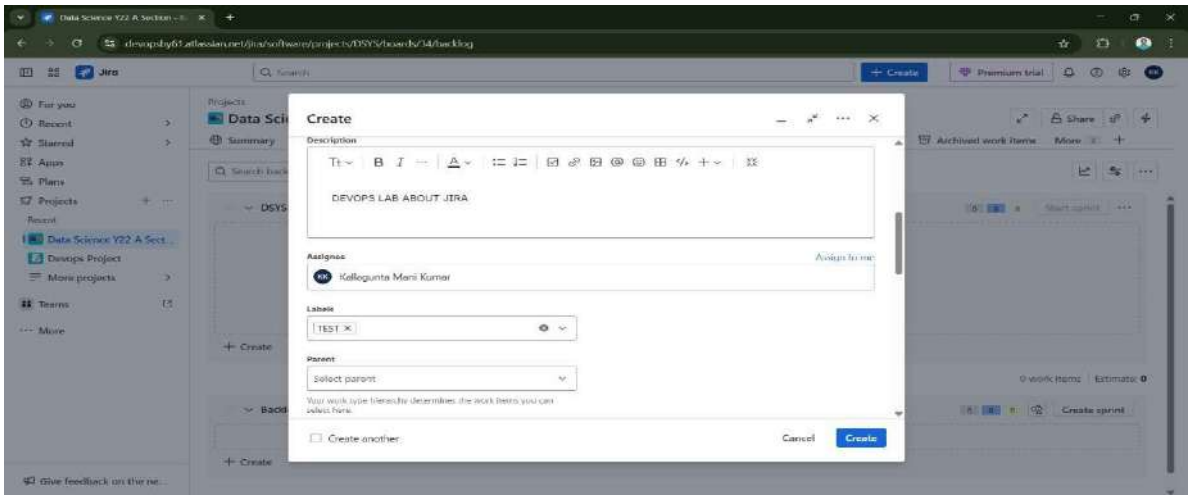
EPIC inside Jira

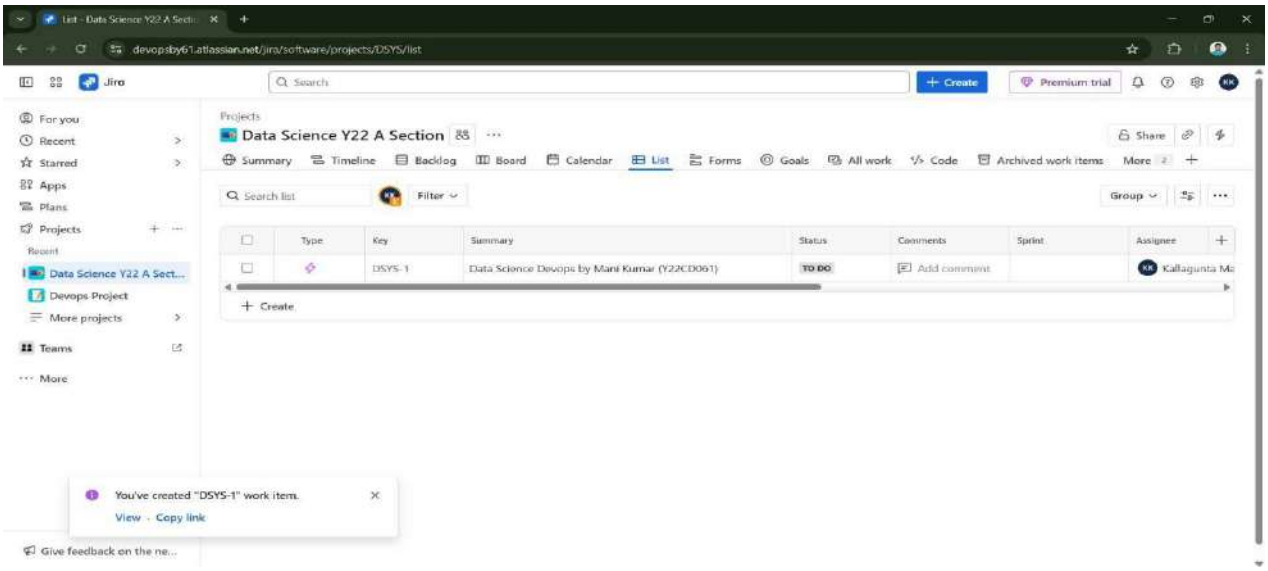
showing a list of available standard Issue Types in a typical Scrum/Agile Jira setup:

- Task
- Bug
- Story
- Epic (currently selected)
- An Epic is a large body of work that can be broken down into a number of smaller stories, tasks, and bugs.

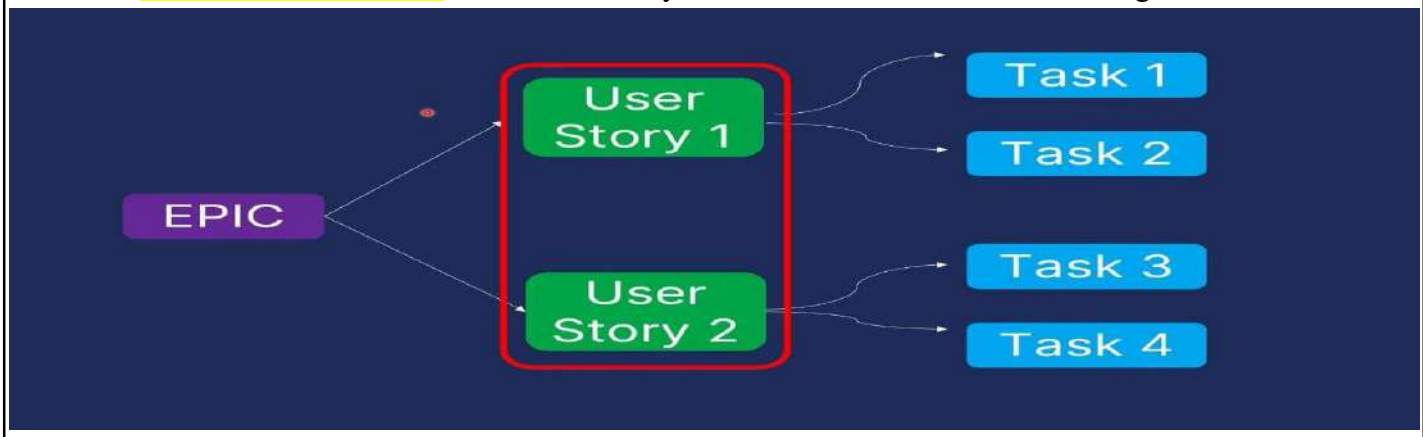

Create
Project *
Data Science Y22 A Section (DSYS)
Work type *
Epic
Task
Bug
Story
Epic
Add work type
Edit work type
Manage work types
Create another
Cancel
Create


Create
To Do
Summary *
Data Science Devops by Mani Kumar (Y22CD061)
Description
DEVOPS LAB ABOUT JIRA
Assignee
Automatic
Create another
Cancel
Create

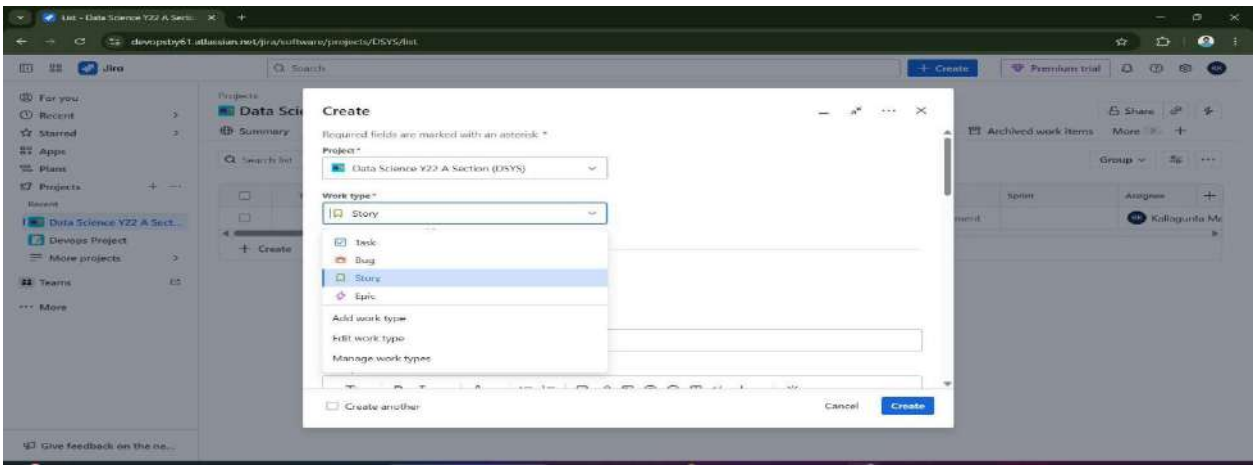

Create
Description
DEVOPS LAB ABOUT JIRA
Assignee
Kollegunta Mani Kumar
Labels
TEST
Parent
Select parent
Create another
Cancel
Create

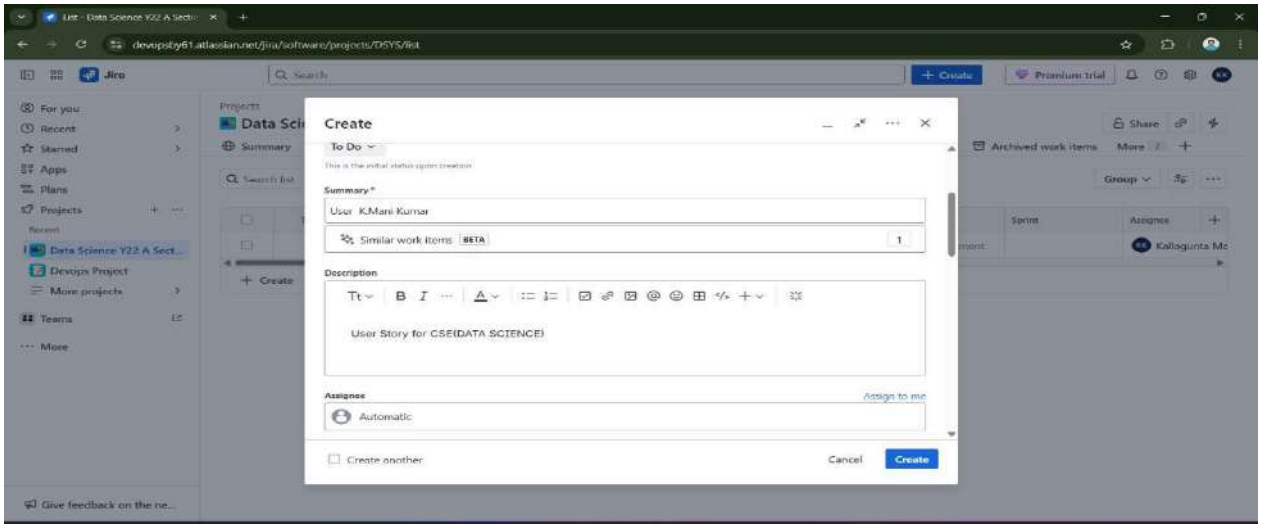


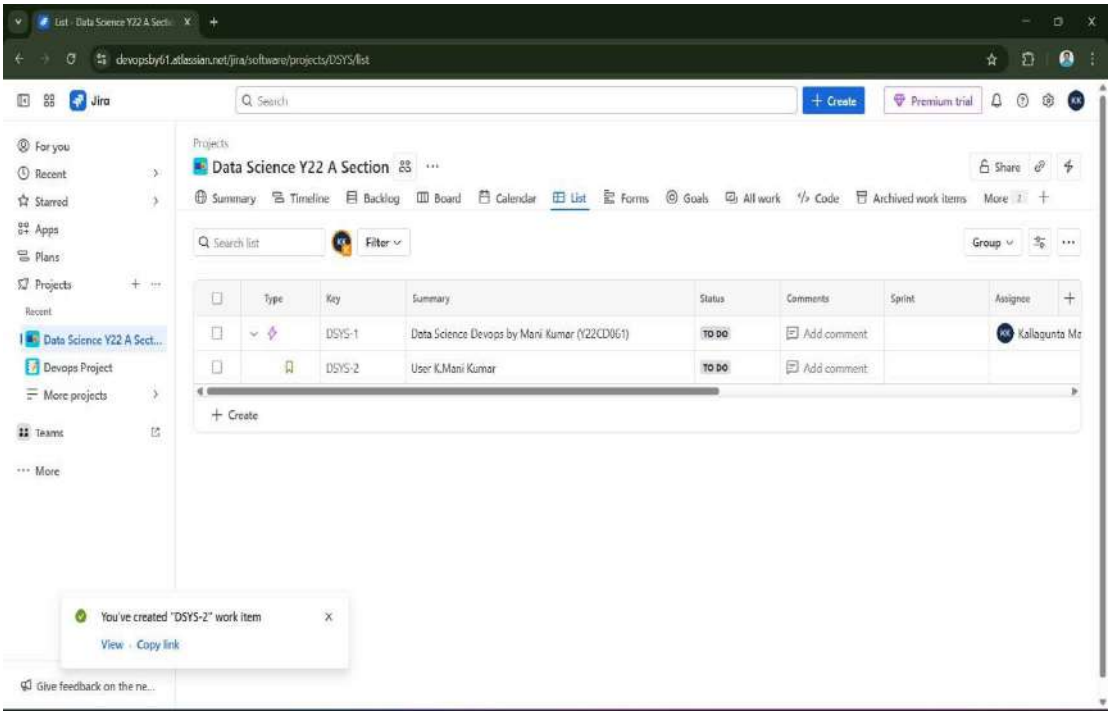
User Stories inside Jira **:** A user story is the smallest unit of work in an agile framework



Jira opening the "Create" issue dialog within the "Data Science Y22 A Section (DSYS)" project.
The user is currently interacting with the "Work type*" field, selecting a type of issue.
The selected work type is "Story", which defines a user-centric requirement for a software feature.
This action prepares the user to define and track a new feature requirement.

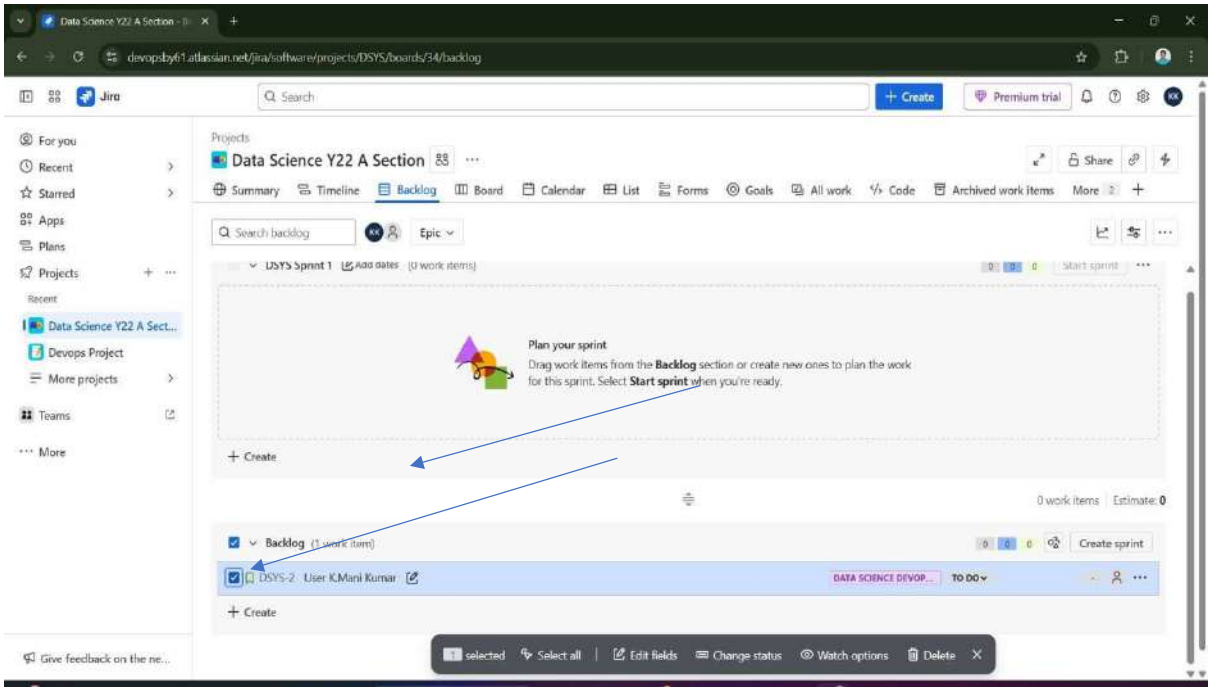




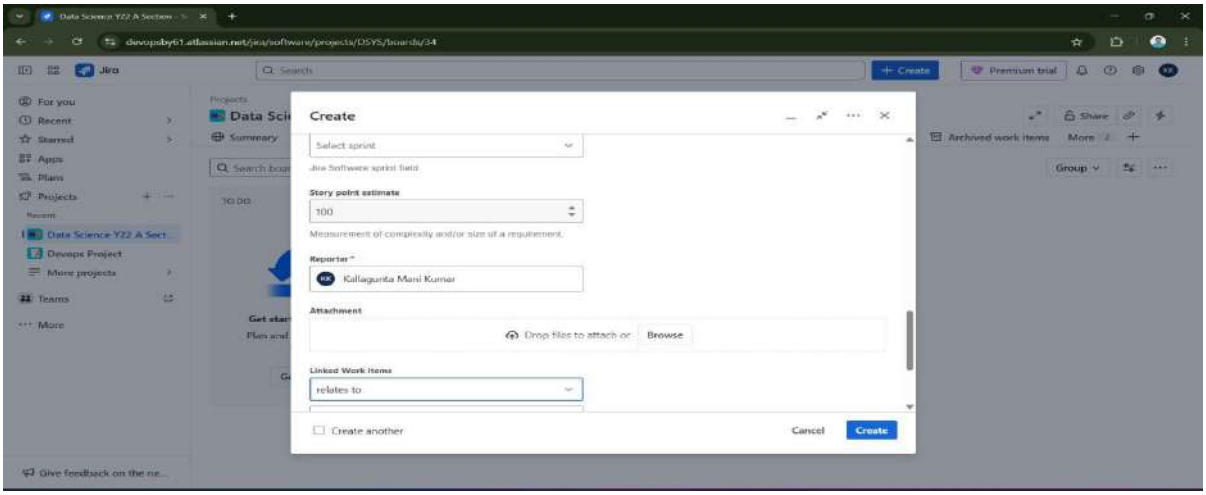


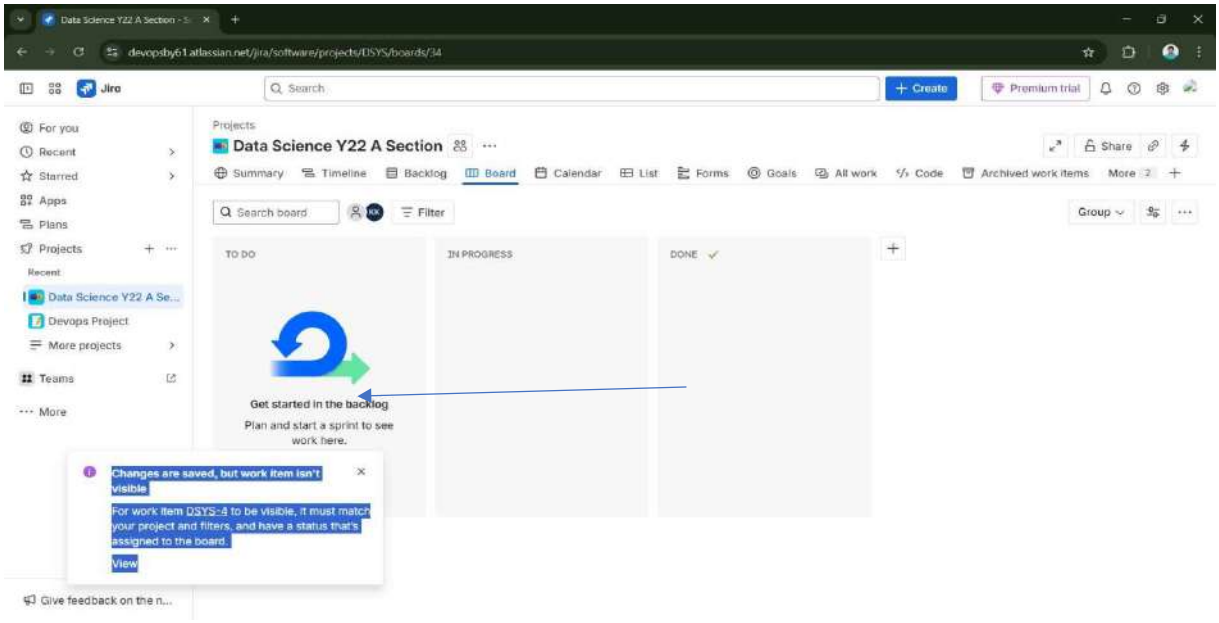
concise view of the progress within the "Data Science Y22 A Section" Jira project.

1. The Jira "List" view shows two successfully created work items in the project: DSYS-1 (an Epic) and DSYS-2 (a Story).
2. Both items are in the initial "TO DO" status, indicating they are ready for work.
3. A notification confirms the recent creation of the DSYS-2 Story.
4. This list now represents the immediate backlog for the project.



Tasks inside Jira: A task is a unit of work the team performs to deliver a backlog item that is estimated in hours

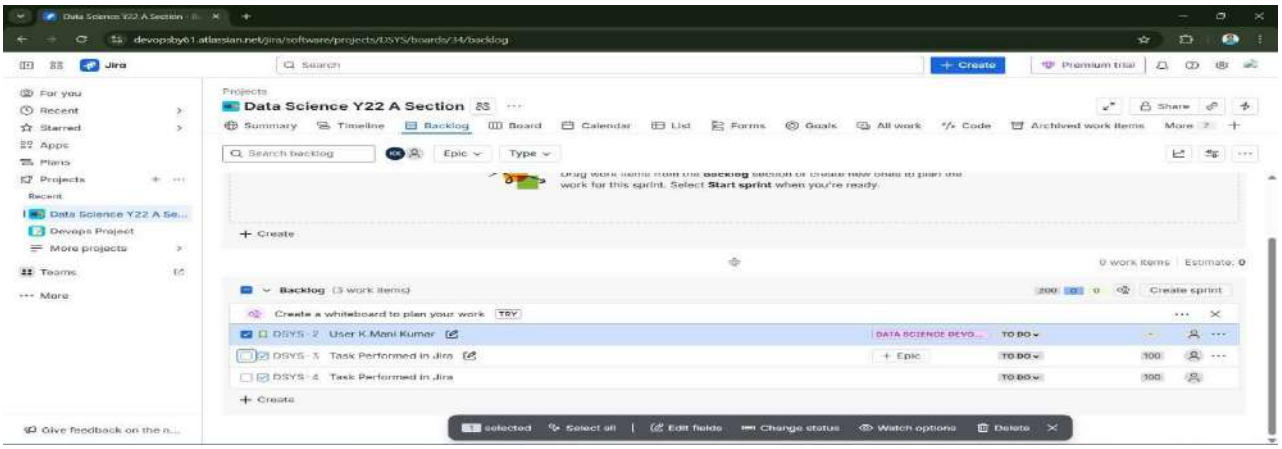




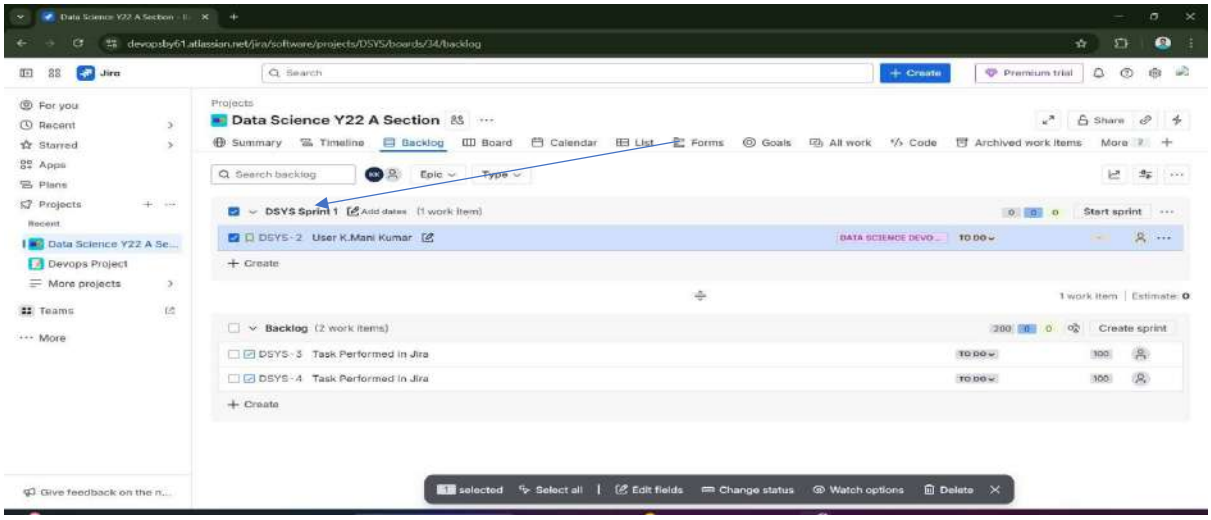
SCRUM Explained:

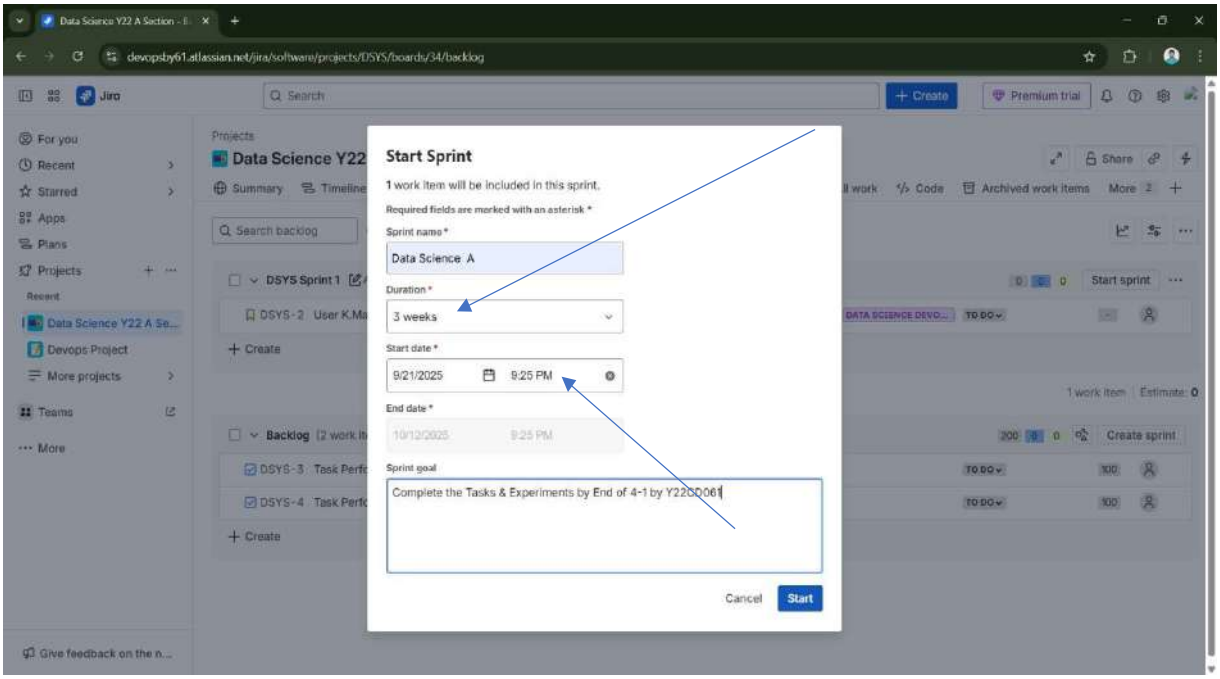
Jira Scrum is a powerful tool for teams that helps them work together with more efficiency. Jira Scrum is a mixture of two elements such as Jira and Scrum. where Jira is a popular product management tool and Scrum is a popular agile methodology

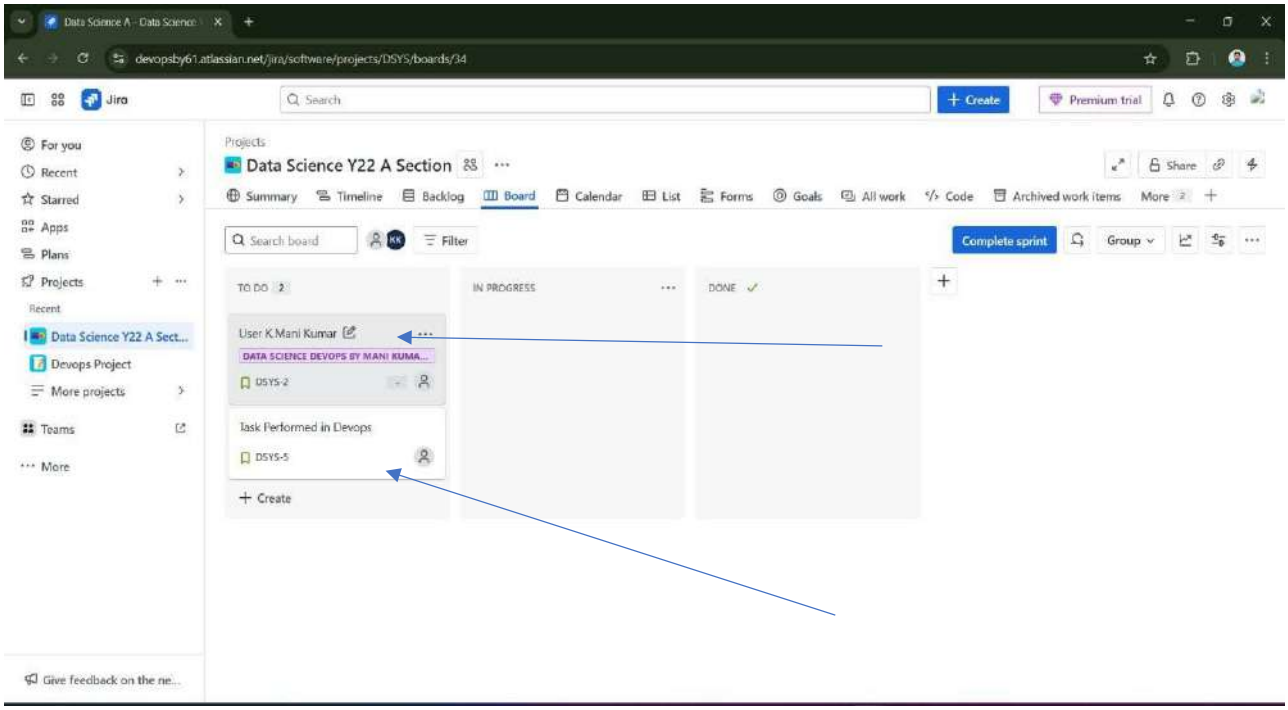
Backlog View og view in Jira is a dynamic, prioritized list of all work items (such as epics, stories, bugs, and tasks) that a team needs to complete, allowing for planning, ranking, and assignment for future sprints or iterations



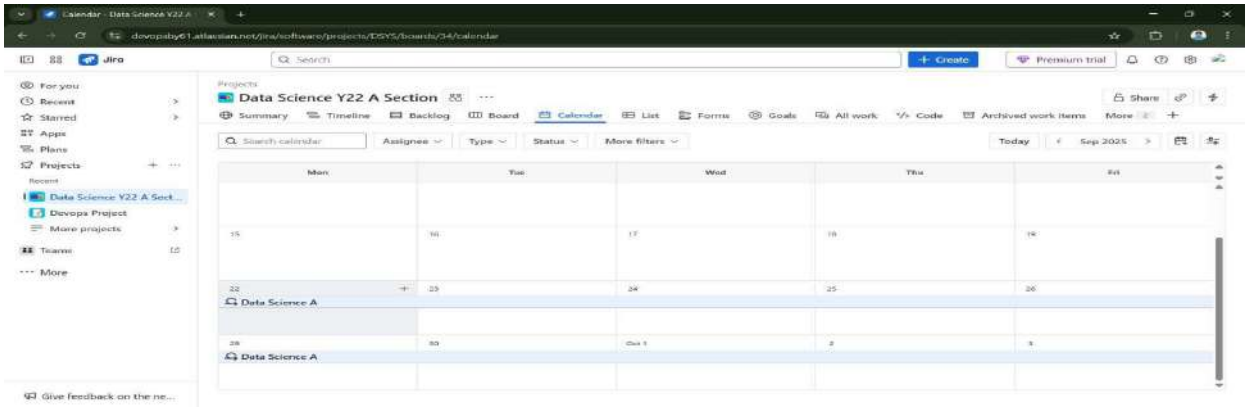
SPRINTs inside Jira

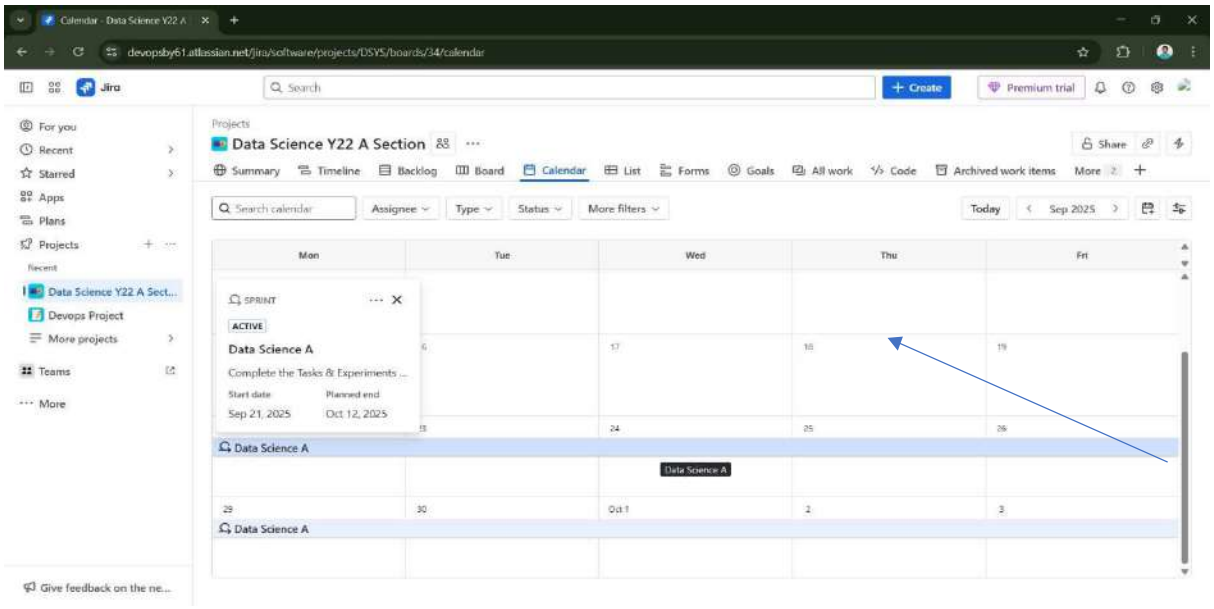


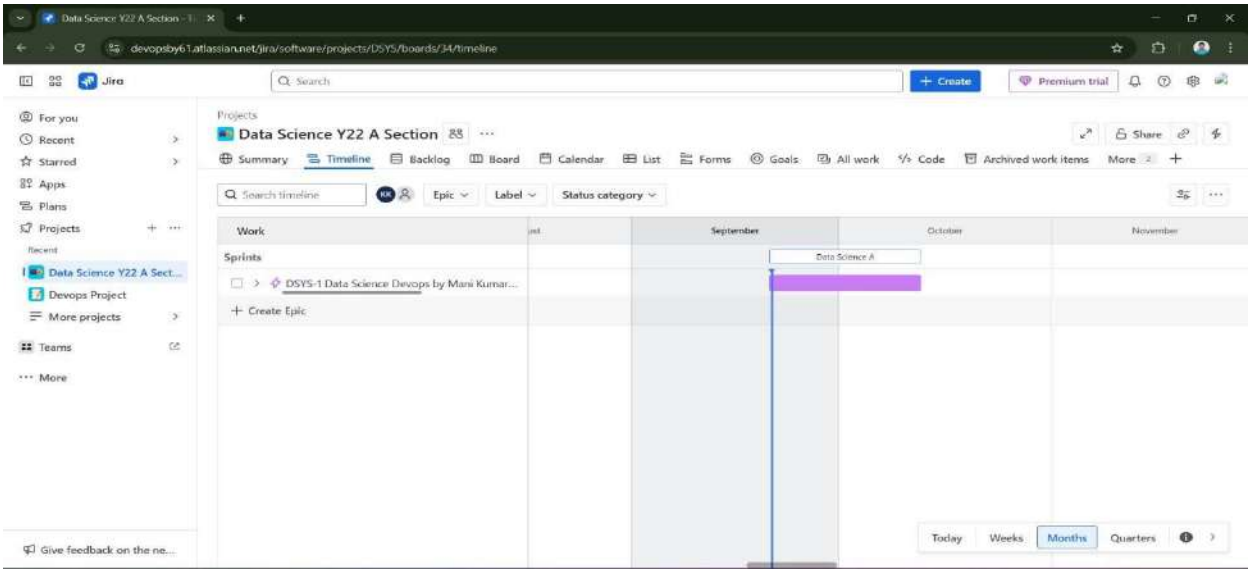




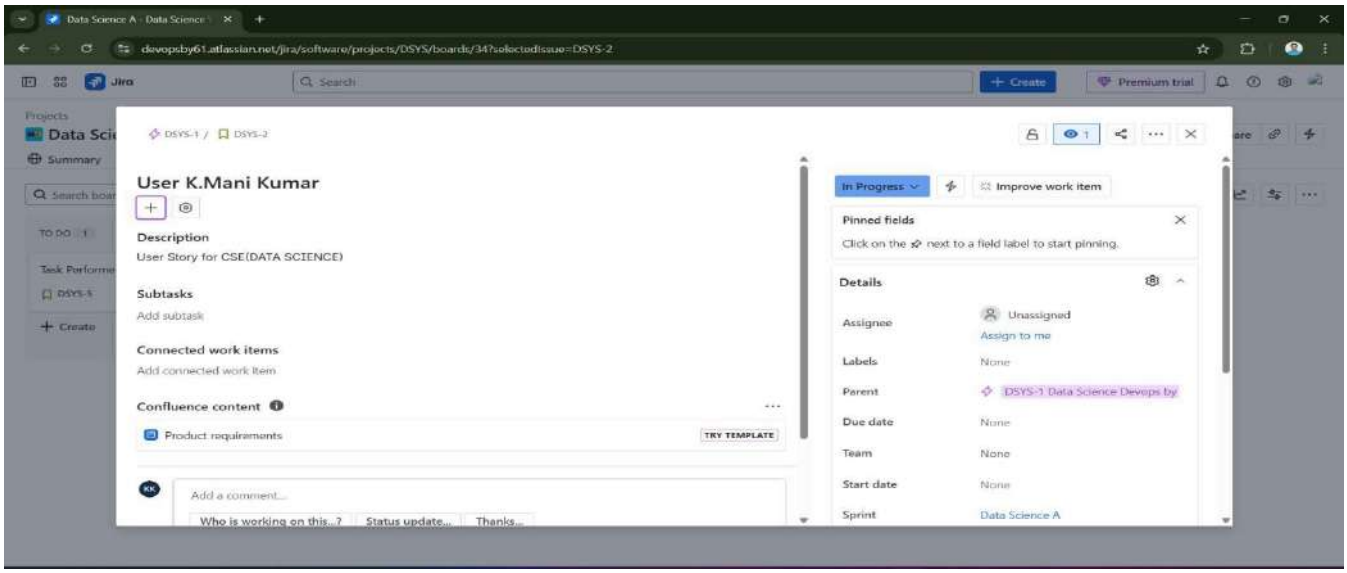
Calendar View in Jira : Jira's calendar view displays your space's work in a calendar format. You can use the calendar to schedule your team's work, see sprints and releases, and keep track of due dates and deadlines



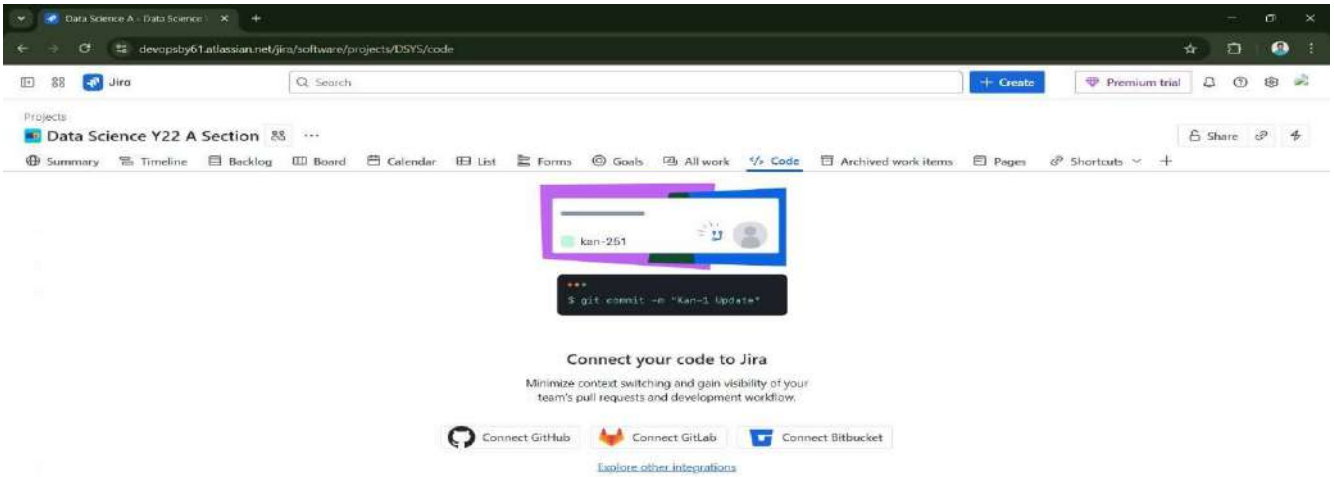




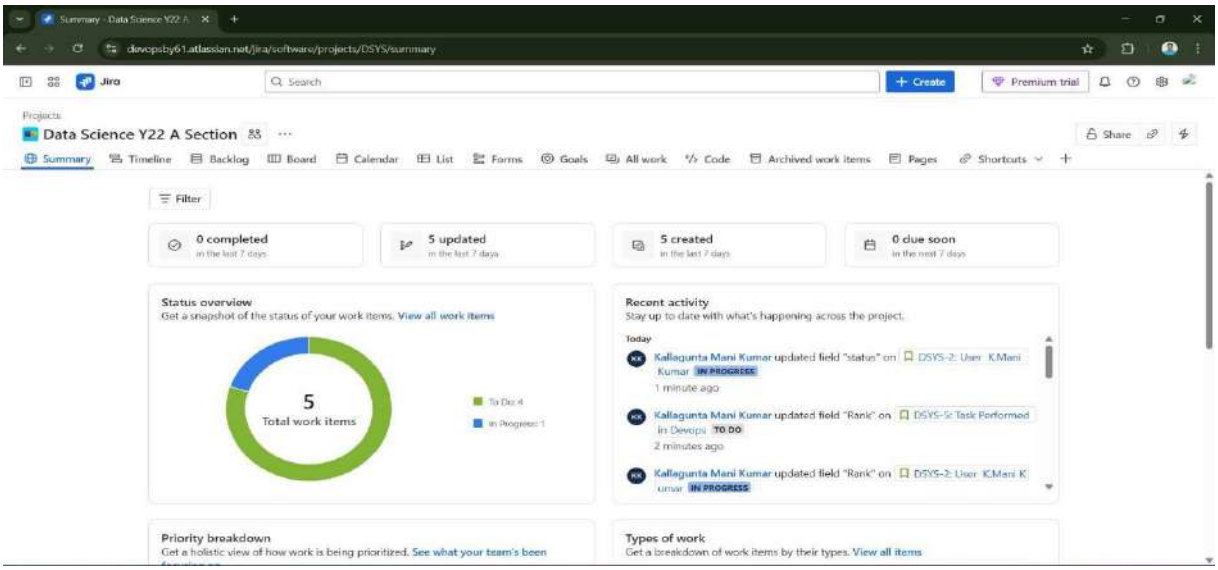
Item View: shows the detailed information of a single work item (like a task or a request) to help users track, update, and resolve it



Code Tab : The Code tab in Jira Software projects provides a centralized view of development activity linked to your Jira issues. This tab becomes accessible after integrating Jira with a source code management tool like Bitbucket or GitHub



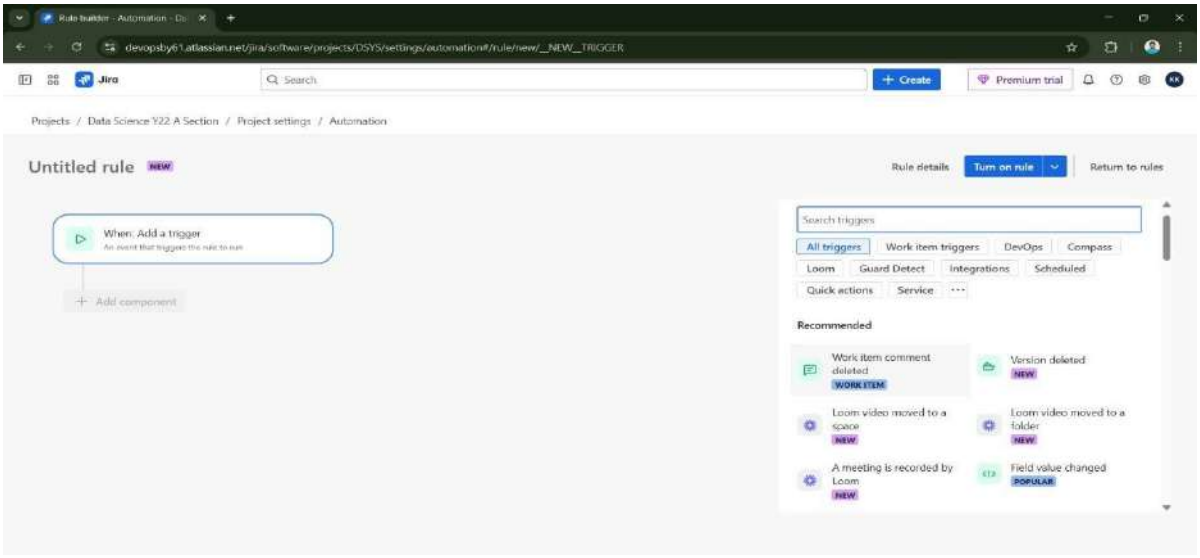
Summary : Summary shows key insights to help monitor progress and health of your plans. Work items that have either a starting date or a due date within the selected date range will show on this scree



Summary Dashboard for the "Data Science Y22 A Section" Jira project.

1. **Project Overview:** The dashboard summarizes recent activity, showing 5 updated and 5 created work items in the last 7 days.
2. **Status Overview:** A donut chart indicates there are 5 Total work Items in the project, with 4 in "To Do" status and 1 "In Progress."
3. **Recent Activity:** The "Recent activity" feed shows that a user named "Kallagunta Mari Kumar" has recently updated several items.
4. **Key Changes:** Specifically, the Story DSYS-2 was moved "IN PROGRESS", indicating work has started on that item.
5. **Conclusion:** The project is active with 5 total issues, one of which is currently being worked on, reflecting the initial stages of execution.

Jira Automations: Automation allows you to focus on the work that matters, removing the need to perform manual, repetitive tasks by allowing your teams to automate their tasks, processes, and workflows. With our simple no-code rule builder, you can create automation rules to take care of everything from the most simple repetitive tasks to the most complex scenarios - all in a few clicks



Jira Automation Rule Builder, where a user is starting to create a new automation rule for the "Data Science Y22 A Section" project. The user is currently at the first step, prompted to "Add a trigger," which defines the event that will start the automation process. The right panel shows various categories and recommended triggers, such as "Work item comment deleted."

Story points in Scrum

Story points are a unit of measure used in agile estimation to evaluate how much effort a user story requires. In scrum teams, this replaces traditional time estimates and focuses instead on relative estimation.

Teams typically assign story point values by considering three key factors:

- **Complexity –** How technically challenging is the work item?
  Example: Integrating with a third-party API that lacks clear documentation.
- **Amount of work –** How many tasks or subtasks are involved?
  Example: Renaming 200 image files and updating their metadata across environments.

- **Uncertainty –** How familiar is the development team with the problem or solution?
  Example: Debugging a rare issue that only happens under specific user roles.

In scrum, story points aren't tied to a specific amount of time. Instead, they're used to compare the relative size of stories. A story assigned 5 points should require more effort than a 2-point story, but less than an 8-point one

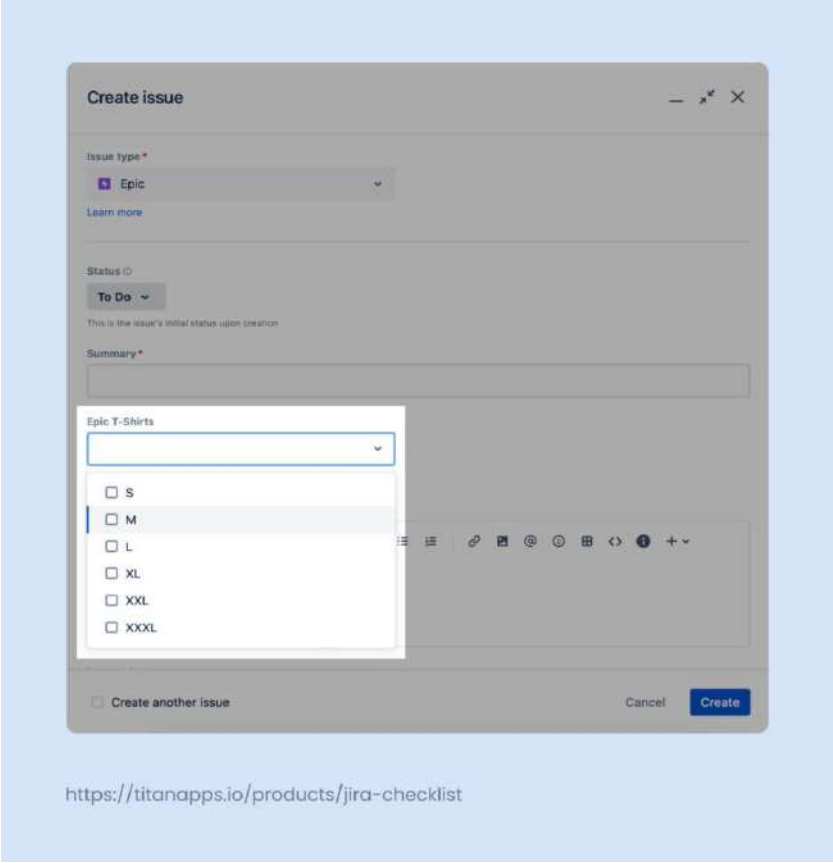
How to estimate a user story with story points?

- Fibonacci sequence (1, 2, 3, 5, 8, 13, 21…) – The most popular option for agile estimation, especially in Jira. It limits choices and encourages teams to think in meaningful increments.

Best for development-heavy teams and sprint execution.

- T-shirt sizes (XS, S, M, L, XL) – A relative scale often used during project planning or high-level backlog grooming. Works well for Epics and roadmap-level planning.
- Powers of 2 (1, 2, 4, 8, 16…) – A simplified approach that escalates faster than Fibonacci. Useful when dealing with larger initiatives or kanban-style workflows.

T-shirt size	Story Point	Time to deliver work
XS	1	Minutes to 1-2 hours
S	2	Half a day
M	3	1-2 days
L	5	Half a week
XL	8	Around 1 week
XXL	13	More than 1 week
XXXL	21	Full Sprint



Experiment-2: Jenkins Integration in AWS

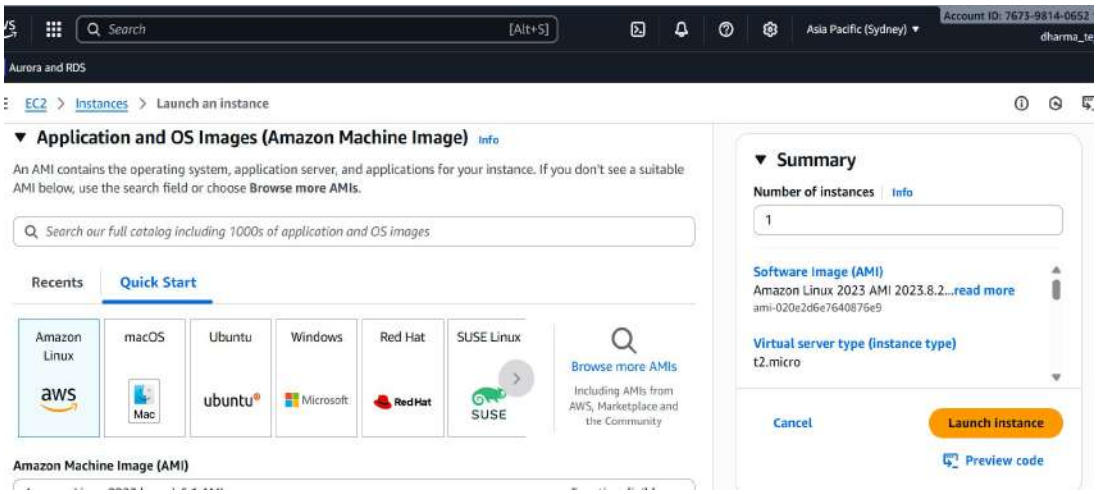
Aim: To demonstrate integration of Jenkins with AWS for continuous integration and Deployment.

1 . Launch the EC2 Instance

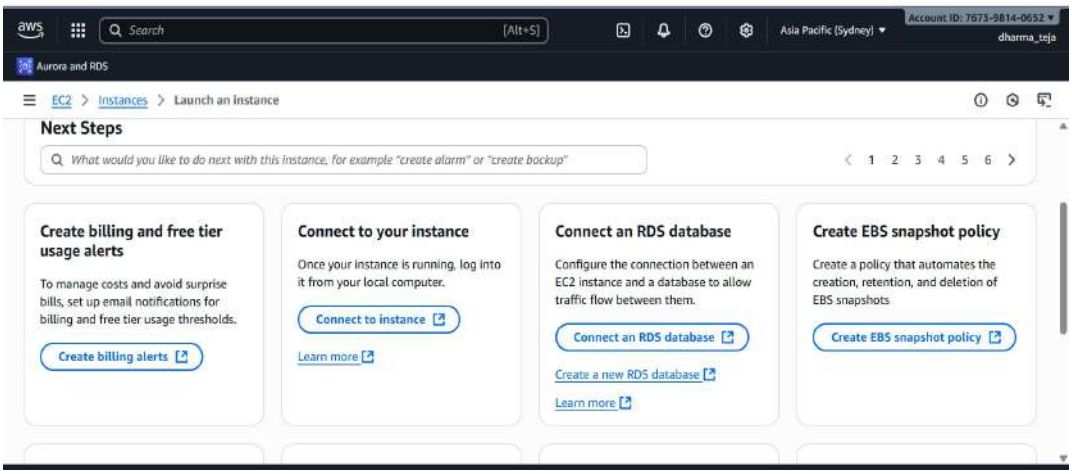
1.1 Navigate to the EC2 dashboard in the AWS Management Console.

1.2 Click on Launch an instance and Amazon Machine Image (AMI), such as Amazon Linux 2, Amazon Linux 2023, or Ubuntu LTS.

1.3 Select the desired instance type, t2.micro (eligible for the AWS Free Tier).



1.4 Complete the launch process and wait for the instance to enter the Running state.



2. Configure Security Groups

2.1 Go to the Security Groups section associated with your instance.

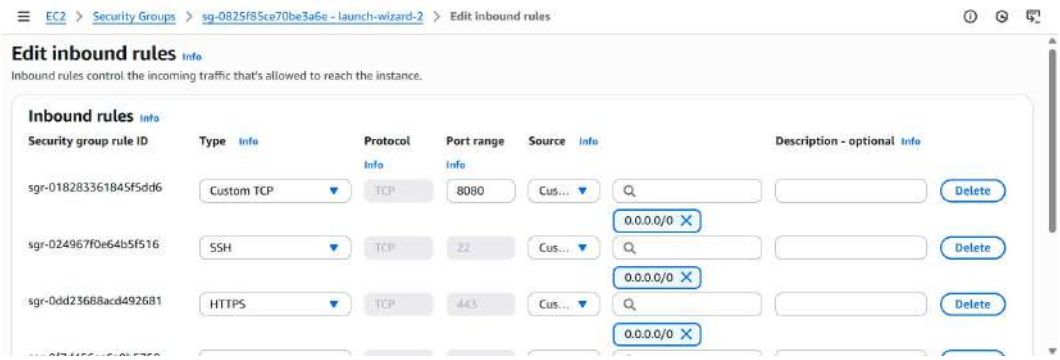
2.2 Edit the inbound rules

2.3 Add a rule for Jenkins:

- Type: Select Custom TCP.
- Port range: Enter 8080.
- Source: Set this to 0.0.0.0/0 (or your specific IP range for better security) to allow access from anywhere.

2.4 Ensure a rule for SSH on port 22 is already present to allow you to connect to the instance.

2.5 Save the inbound rules.



3. Connect and Install Software

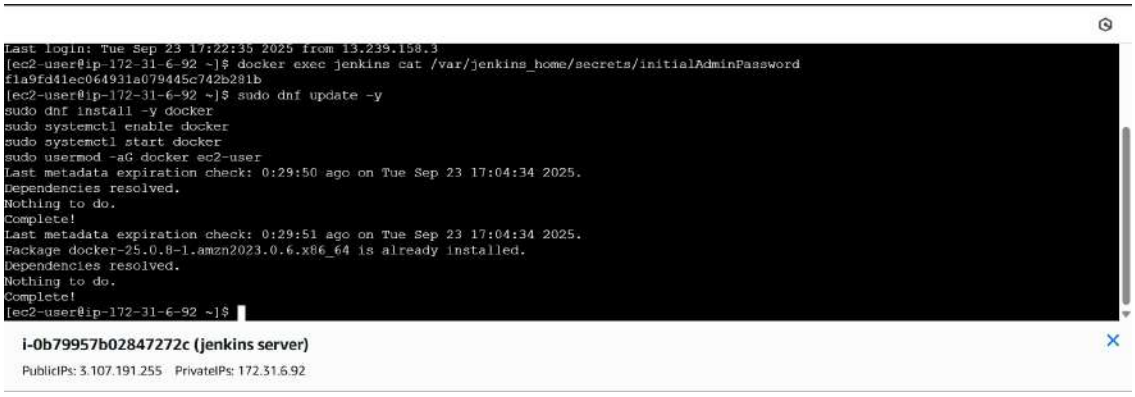
3.1 Connect to your instance using SSH or the AWS EC2 Instance Connect tool.

3.2 Install Docker: Run the following commands to update the system and install Docker:

sudo dnf update -y

sudo dnf install -y docker

3.3 Start and Enable Docker:



4. Deploy Jenkins using Docker

4.1 Pull the Jenkins LTS image: docker pull jenkins/jenkins:lts

4.2 Run the Jenkins container: This command runs Jenkins in the background (-d), names it jenkins, maps the host port 8080 to the container port 8080, and mounts a volume (-v) to persist data in /var/jenkins_home.

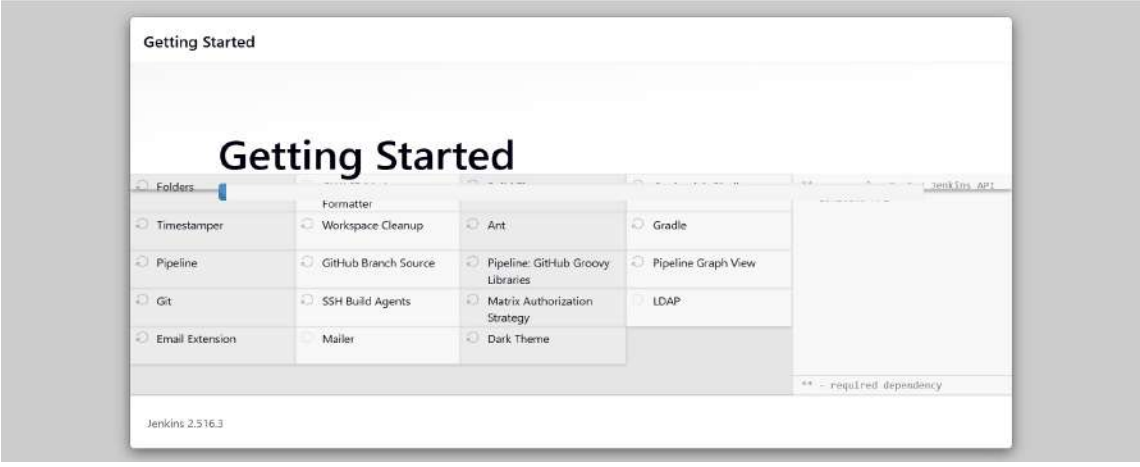
5. Access and Configure Jenkins

51. Access the Jenkins Web UI: Open your web browser and navigate to http://<EC2-public-ip>:8080. Replace <EC2-public-ip> with the Public IP address of your EC2 instance

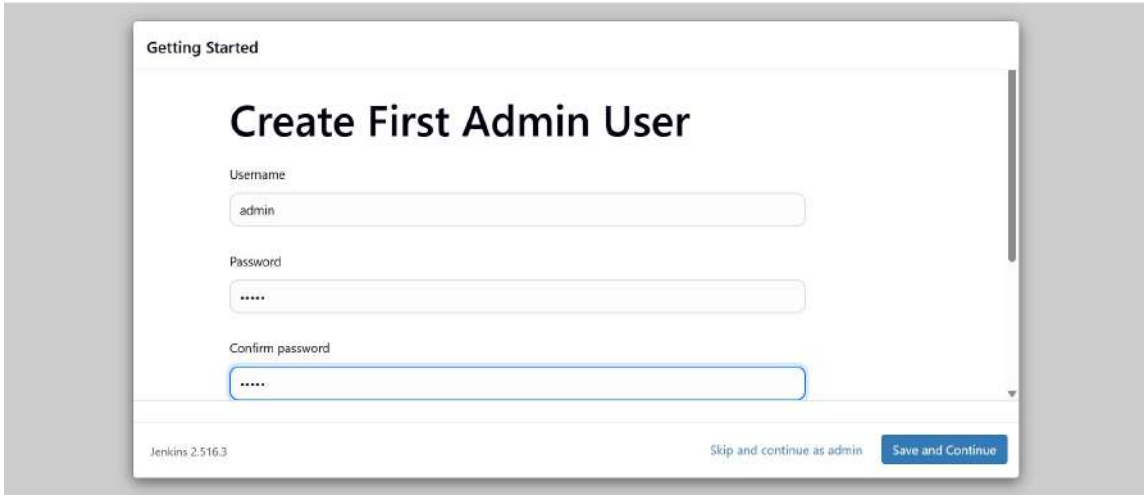
5.2 Retrieve the Initial Admin Password: Run the following command on your EC2 instance's command line to get the initial security password needed to unlock Jenkins:

5.3 **Complete Setup**:

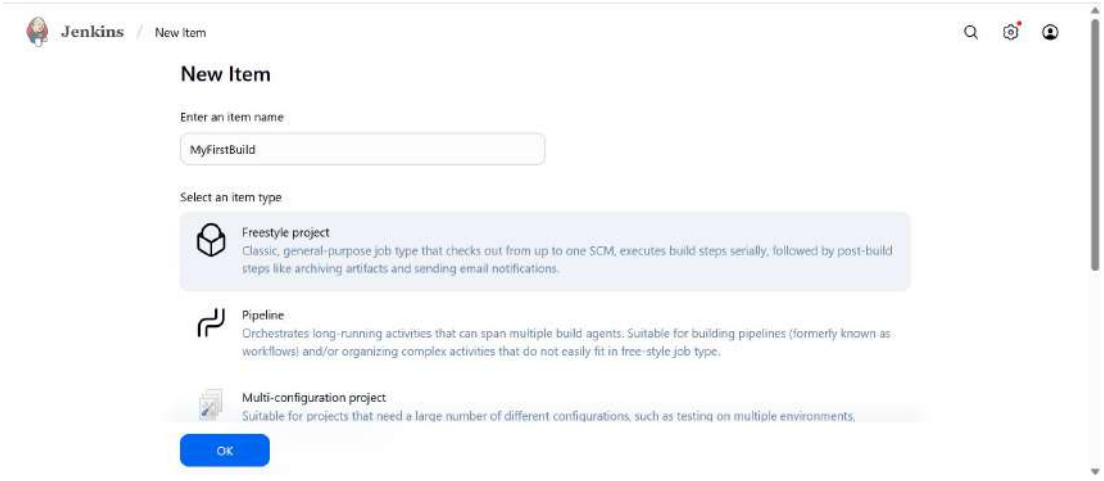
- Enter the password retrieved in the previous step.
- Follow the on-screen instructions, typically by installing suggested plugins.



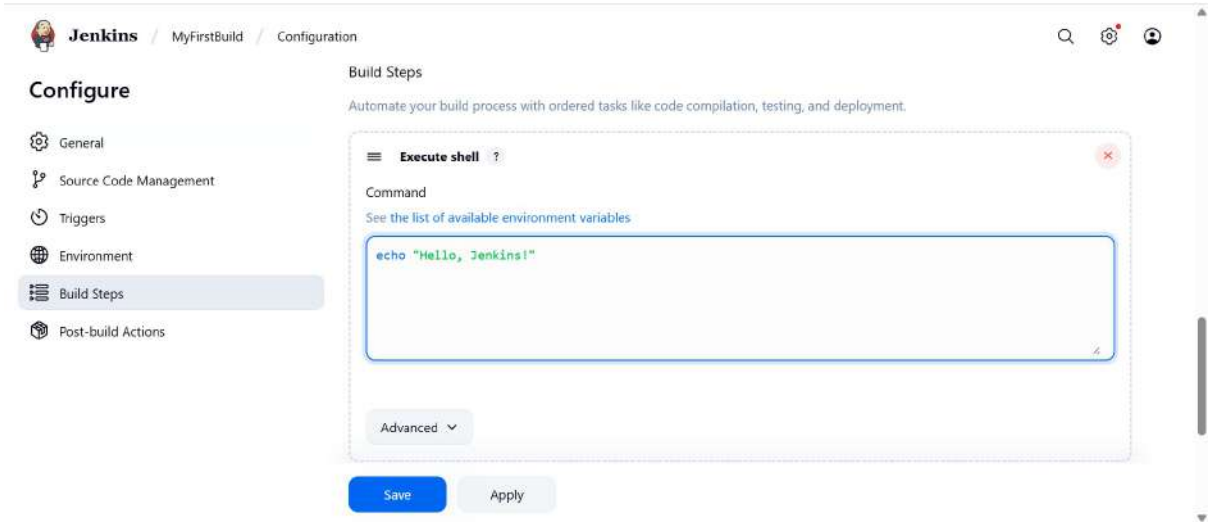
6. Create and Run a Sample Job



6.1 Create a New Item



6.2 Add Build step

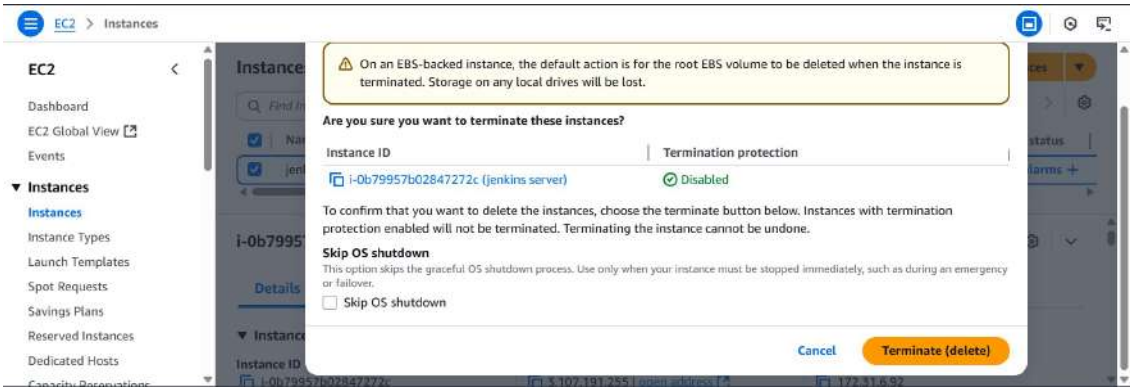


6.3 Run the Build and check the output in Console

7. Cleanup and Termination

7.1 Stop Jenkins

7.2 Terminate the EC2 instance and exit



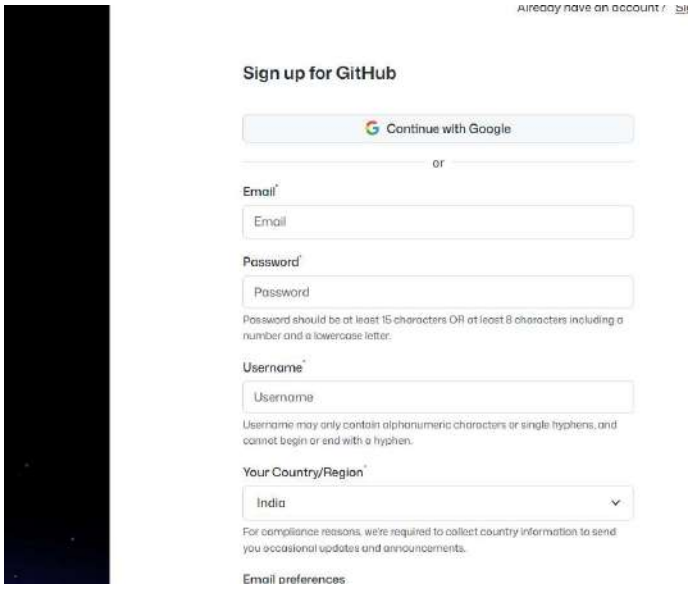
Experiment-3: Demonstration of Git Version Control

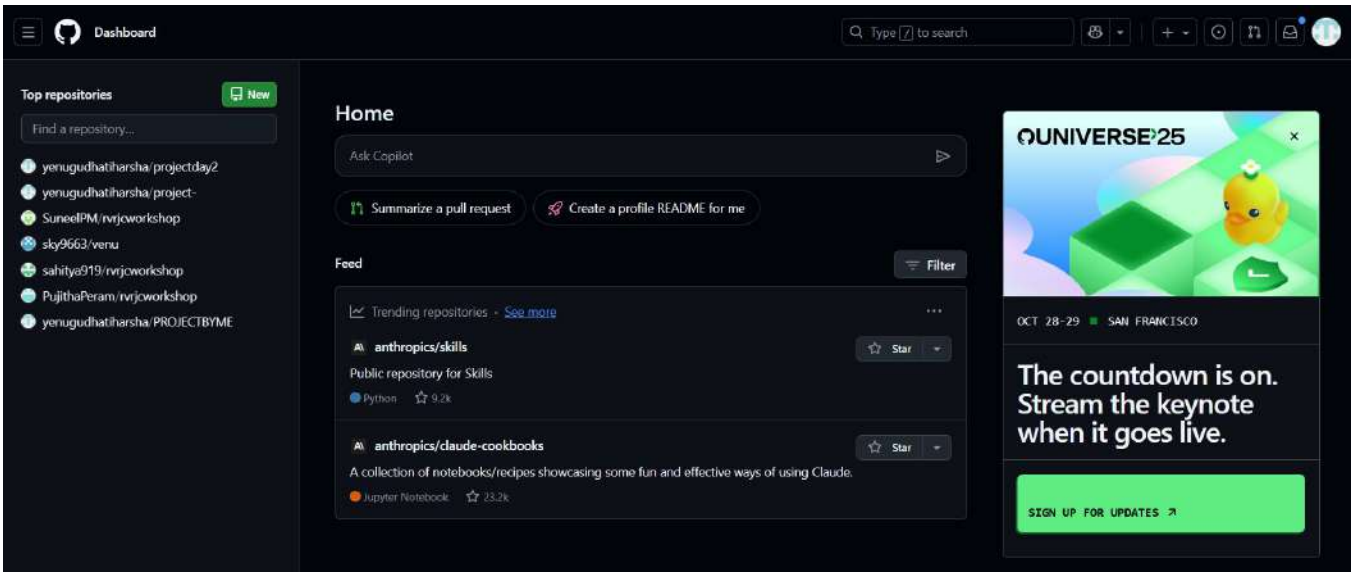
Aim: To demonstrate basic version control operations using Git.

1. **Creating a GitHub account**: **Open the Official Website then** go to: https://github.com/
   This is the official homepage of GitHub.

Click on "Sign up". On the top-right corner of the page, click the **"Sign up"** button.

Enter the details and then a GitHub account will be created.



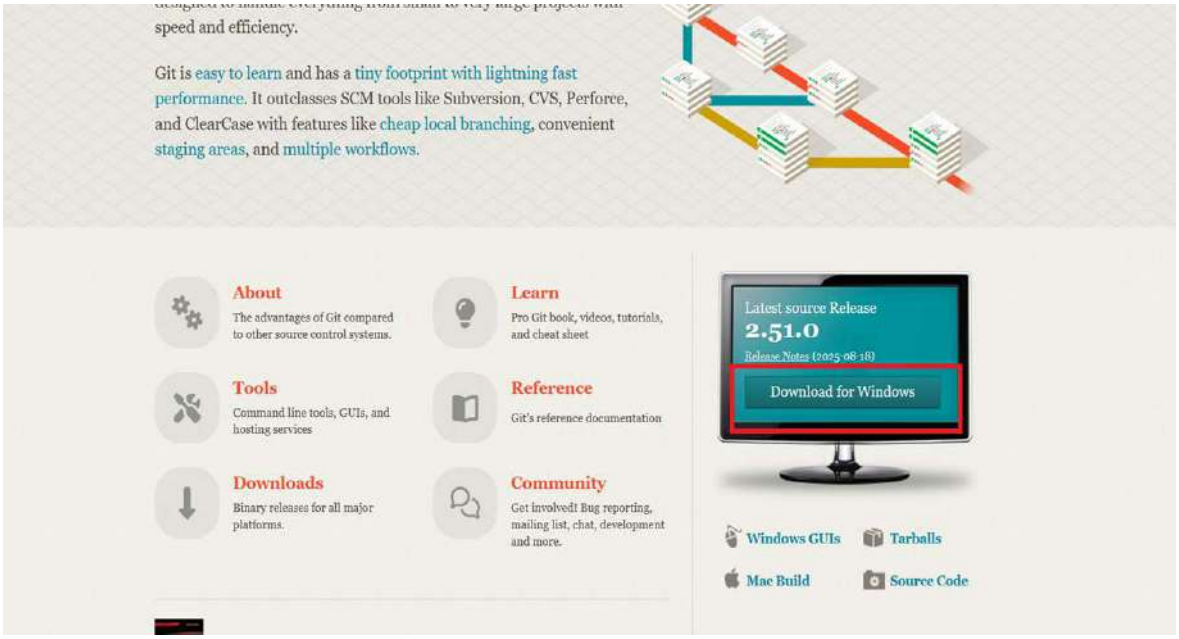


Demonstrate Working with Git bash using github.

1. **Git installation:** In order to ensure whether git is installed in your laptop or not use the command which is mentioned below:

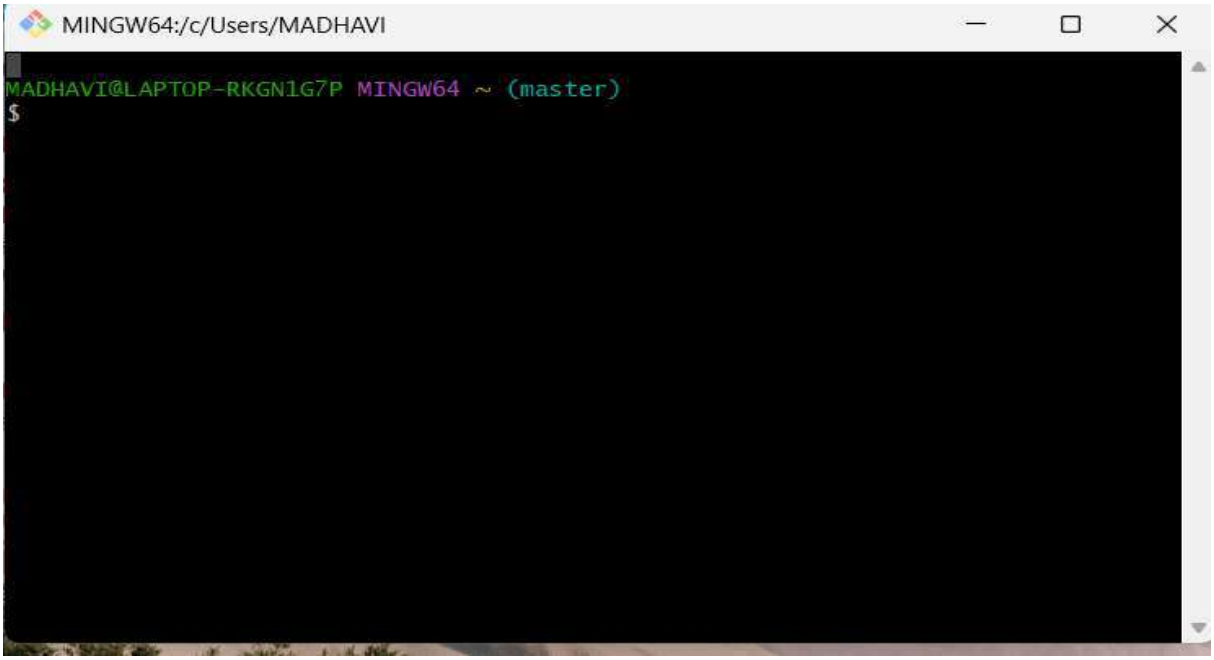
As in the above image we can see that version of git is shown, so we can say that git is installed in your laptop. If not installed, download and install from git-scm.com.

After clicking in link provided below in order to download git a browser will open and then click on download for windows which is shown in below figure.

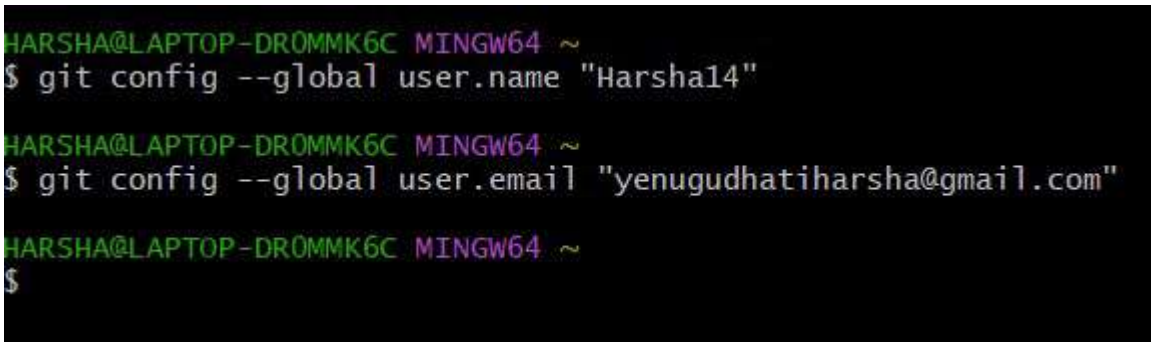


Now after downloading git what should we open to execute the commands? At this point many will confuse and will open command prompt but we have to open git bash terminal

). After clicking on the symbol of git bash(make sure that you **run it as an administrator**) the terminal opens and now, we can execute any git commands.



2. **Git configuration:** Set up your identity



Check current configurations which show global or local Git settings such as:

i. Line ending and file conversion settings

ii. Git LFS (Large File Storage) filters

iii. Credential helper

iv. Default branch name

v. User identity configuration

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 ~
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=schannel
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.name=Harsha14
user.email=yenugudhatiharsha@gmail.com

HARSHA@LAPTOP-DR0MMK6C MINGW64 ~
$
```

3. **Create a directory:** Use the **mkdir** command to create a new directory and then navigate into it.

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 ~
$ cd d:

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d
$ mkdir l23cd216

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d
$ cd l23cd216

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216
$
```

4. **Initialize a Github repository:** Now that we are inside the directory, now initialize the github repository.

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216
$ git init
Initialized empty Git repository in D:/l23cd216/.git/

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (master)
$
```

5. **Check repository status:** After initializing the Git repository with **git init**, you can check the current status of your working directory using:

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (master)
$ git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (master)
$
```

As in above image we can see that the files in **y22cd168** folder are untracked files

6. **Add files to staging area:** In order to convert untracked files to tracked files we use the command mentioned in the image below.

With the help of **git add .** command mentioned in image the untracked files will be converted to tracked files.

With the help of git status command, we can be able to see which files are converted from untracked to tracked files in the current working directory.

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        123.html
        example.html

nothing added to commit but untracked files present (use "git add" to track)

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (master)
$
```

7. **Commit changes:** Record the staged files in the repository.

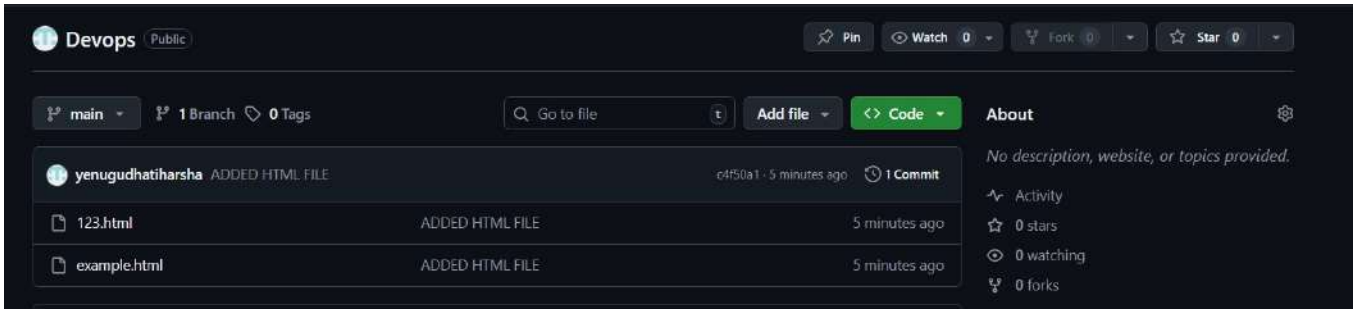
```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (master)
$ git commit -m "ADDED HTML FILE"
[master (root-commit) c4f50a1] ADDED HTML FILE
 2 files changed, 2 insertions(+)
 create mode 100644 123.html
 create mode 100644 example.html

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (master)
$
```

8. **Connect to a remote repository:** Link the local repository with a remote repository. First open your github account and create a new repository.

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (master)
$ git remote add origin https://github.com/yenugudhatiharsha/Devops.git

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (master)
$ git remote -v
origin  https://github.com/yenugudhatiharsha/Devops.git (fetch)
origin  https://github.com/yenugudhatiharsha/Devops.git (push)

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (master)
$
```

9. **Push changes to remote:** Upload local commits to the remote repository. Before pushing commits to repository make sure you are in main branch.

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (master)
$ git branch -M main

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ git push -u origin main
info: please complete authentication in your browser...
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 305 bytes | 305.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/yenugudhatiharsha/Devops.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$
```



Experiment-4: Working with Git Branches

Aim: To demonstrate branching and merging in Git.

1. **Creating a branch:** Creates a new branch in your Git repository with the specified name. This creates a branch named feature-login but keeps you on your current branch **(e.g., main).**

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ git branch feature-login

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ git branch
  feature-login
* main

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$
```

2. **Create & switch to branch:** This creates a new branch and switches to the branch you have newly created.

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ git checkout -b feature
Switched to a new branch 'feature'

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (feature)
$
```

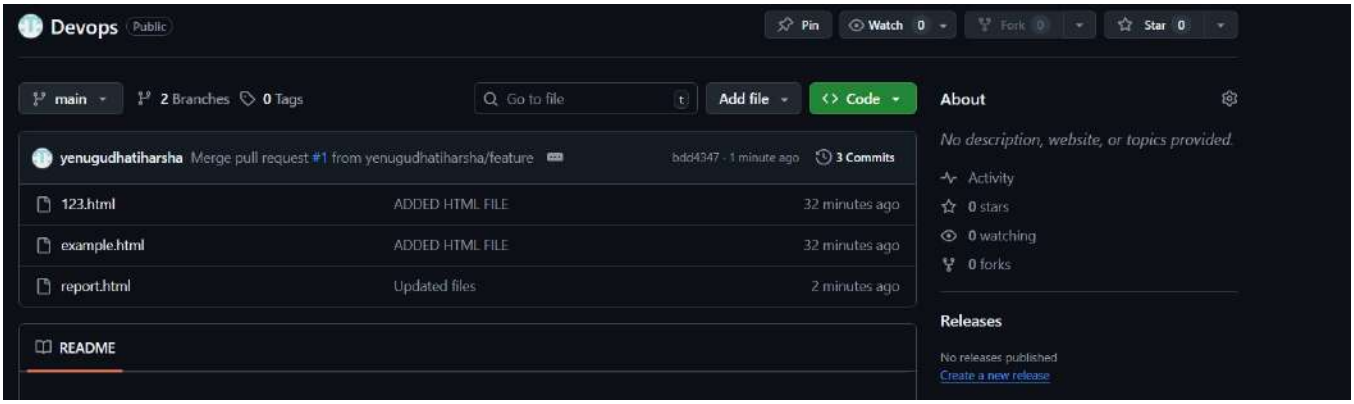
3. **Add & commit changes on new branch:** Make changes or add new files, then stage and commit.

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (feature)
$ git add .

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (feature)
$ git status
On branch feature
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   report.html


HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (feature)
$ git commit -m "Updated files"
[feature 3ac58bd] Updated files
 1 file changed, 1 insertion(+)
 create mode 100644 report.html

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (feature)
$
```

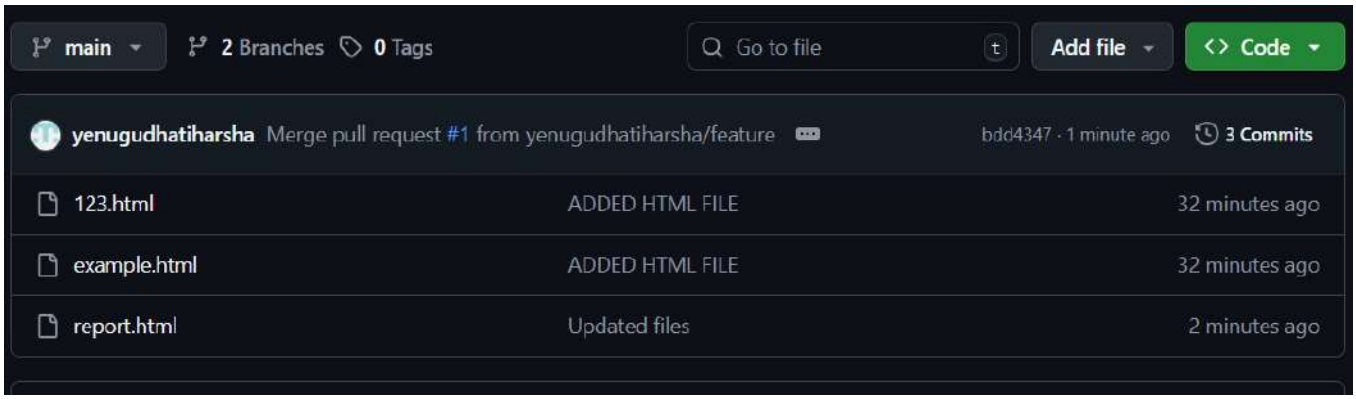
4. **Push the new branch from local to remote:** Uploads your local branch commits to the remote repository and sets up tracking.

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (feature)
$ git push -u origin feature
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 316 bytes | 316.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
remote:      https://github.com/yenugudhatiharsha/Devops/pull/new/feature
remote:
To https://github.com/yenugudhatiharsha/Devops.git
 * [new branch]      feature -> feature
branch 'feature' set up to track 'origin/feature'.

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (feature)
$
```

Here I changed **157.html** file & added the file to feature branch. Now go to your repo on GitHub. You'll see a message: **"Compare & pull request**". Click it.



Then add a description of what you did and then create a **pull request->click on merge pull request->confirm merge**. Then your changes are pulled into main branch.



5. **Merge conflict:** Create a file and make initial commit in main branch. Now checkout to another branch you created and modify file and then make initial commit. Now switch back to main branch and create conflicting change.

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ echo "Hi HARSHA" > conflict.txt

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ git add conflict.txt
warning: in the working copy of 'conflict.txt', LF will be replaced by CRLF the
next time Git touches it

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ git commit -m "Committed a text file"
[main e688732] Committed a text file
 1 file changed, 1 insertion(+)
 create mode 100644 conflict.txt

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ git checkout feature-login
Switched to branch 'feature-login'

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (feature-login)
$ echo "Hi Harsha from new-branch" > conflict.txt

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (feature-login)
```

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (feature-login)
$ git add conflict.txt
warning: in the working copy of 'conflict.txt', LF will be replaced by CRLF the
next time Git touches it

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (feature-login)
$ git commit -m "Added a text file"
[feature-login f540cca] Added a text file
 1 file changed, 1 insertion(+)
 create mode 100644 conflict.txt

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (feature-login)
$ git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ git merge feature-login
Auto-merging conflict.txt
CONFLICT (add/add): Merge conflict in conflict.txt
Automatic merge failed; fix conflicts and then commit the result.

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main|MERGING)
```

6. **Resolve merge conflict:** open the file **(conflict.txt)** and manually choose what content you want to keep. Now add the file, make initial commit and then try to merge the branch.

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ git merge feature-login
Auto-merging conflict.txt
CONFLICT (add/add): Merge conflict in conflict.txt
Automatic merge failed; fix conflicts and then commit the result.

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main|MERGING)
$ notepad conflict.txt

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main|MERGING)
$ git add conflict.txt

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main|MERGING)
$ git commit -m "Added a text file"
[main f11378c] Added a text file

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ git merge feature-login
Already up to date.

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$
```

7. **Delete a branch:** In order to delete the branch locally use the following command which is mentioned in below image.

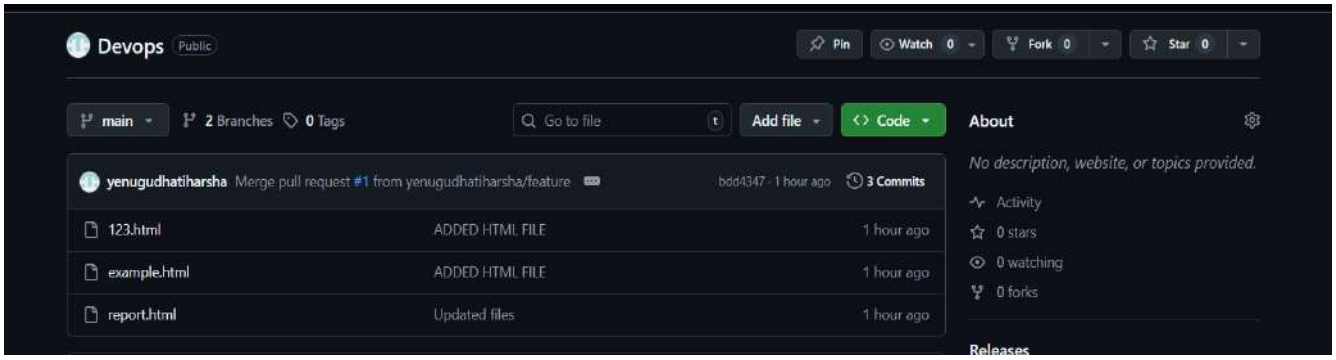
```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ git branch -d feature-login
Deleted branch feature-login (was f540cca).

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ git branch
  feature
* main
  origin

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$
```

The -d (short for --delete) flag is a **safe delete**: It will **refuse to delete** the branch if it hasn't been **fully merged** with your current branch to avoid losing unmerged work.

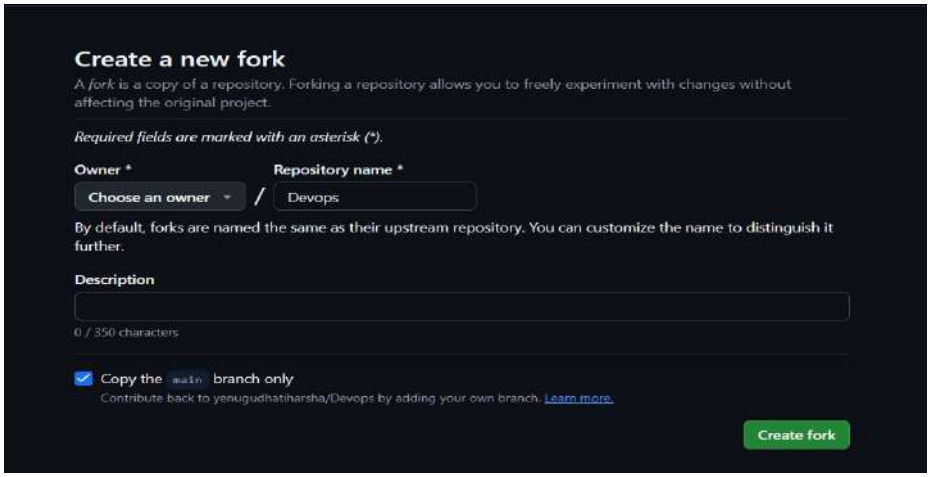
8. **Cloning repository:** Creating a local copy of a Git project that exists on a remote server (like GitHub, GitLab, or Bitbucket). This allows you to view, edit, and contribute to the project from your own laptop.

```
HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ git clone https://github.com/yenugudhatiharsha/Devops.git
Cloning into 'Devops'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 8 (delta 2), reused 5 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (8/8), done.
Resolving deltas: 100% (2/2), done.

HARSHA@LAPTOP-DR0MMK6C MINGW64 /d/l23cd216 (main)
$ ls
123.html  Devops/  conflict.txt  example.html
```

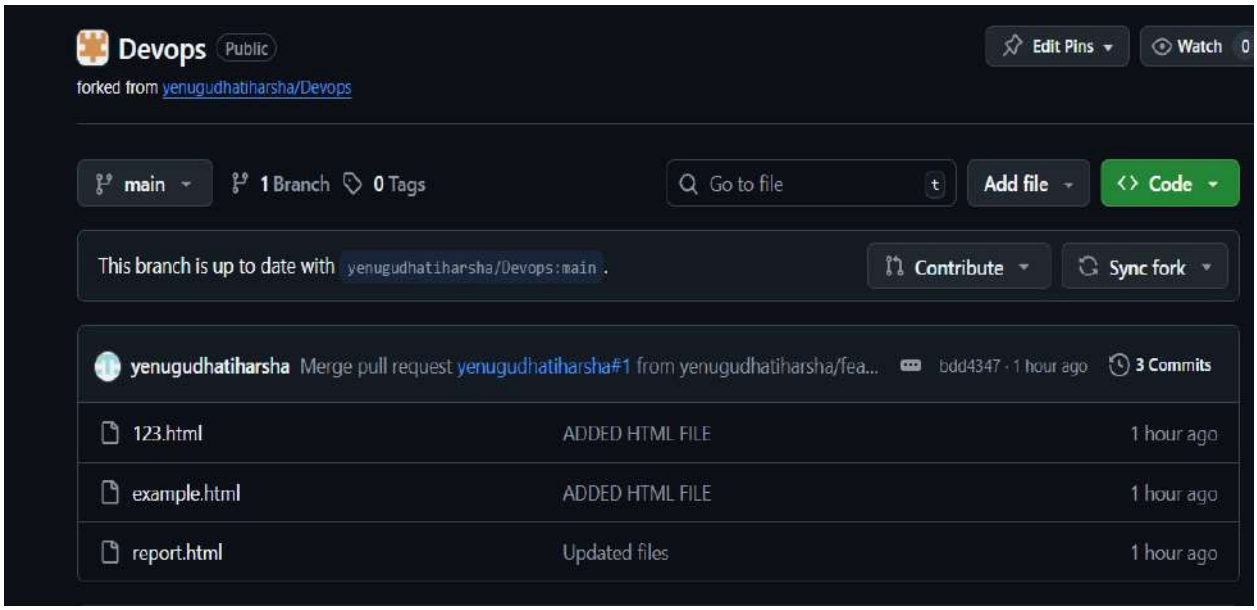
9. **Fork the repo**: Go to the GitHub repo page → click **Fork** (top-right) → choose your account.



After clicking on create new fork the browser looks like the below figure.



After forking, the browser will be like this.

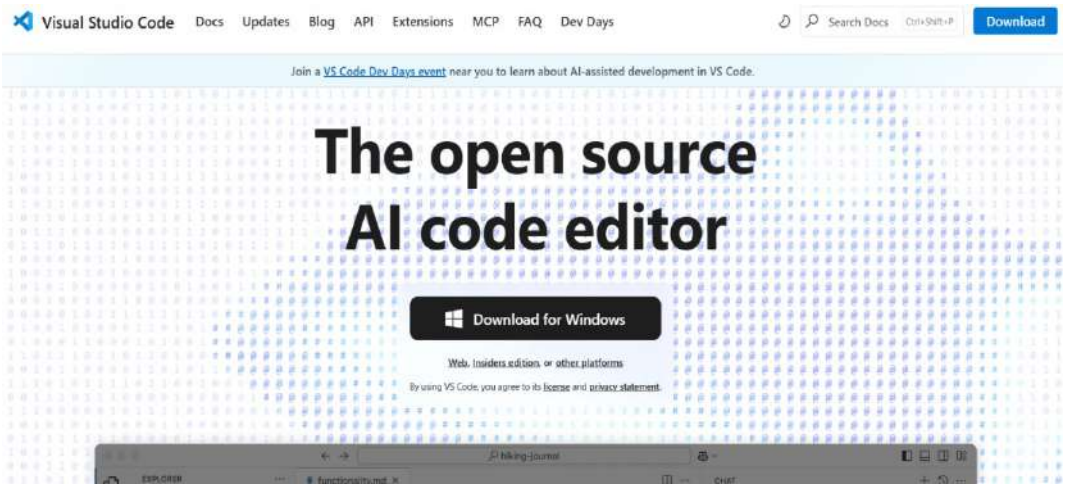


Now after forking now, we can clone this repository and can execute any commands in GitBash.

1. Integrating Git with Visual Studio Code

Aim: To demonstrate the integration of Git with VS Code.

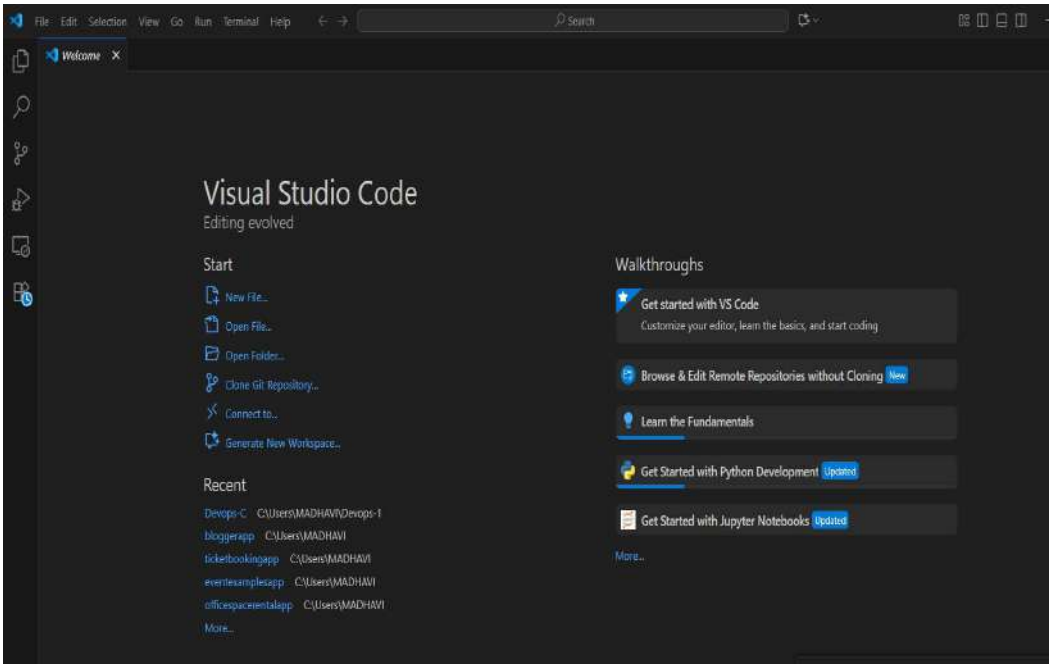
1. **Installing VS:** Open your web browser and visit: https://code.visualstudio.com/ **.Y**ou'll see a blue **"Download for Windows"** button (it automatically detects your operating system).



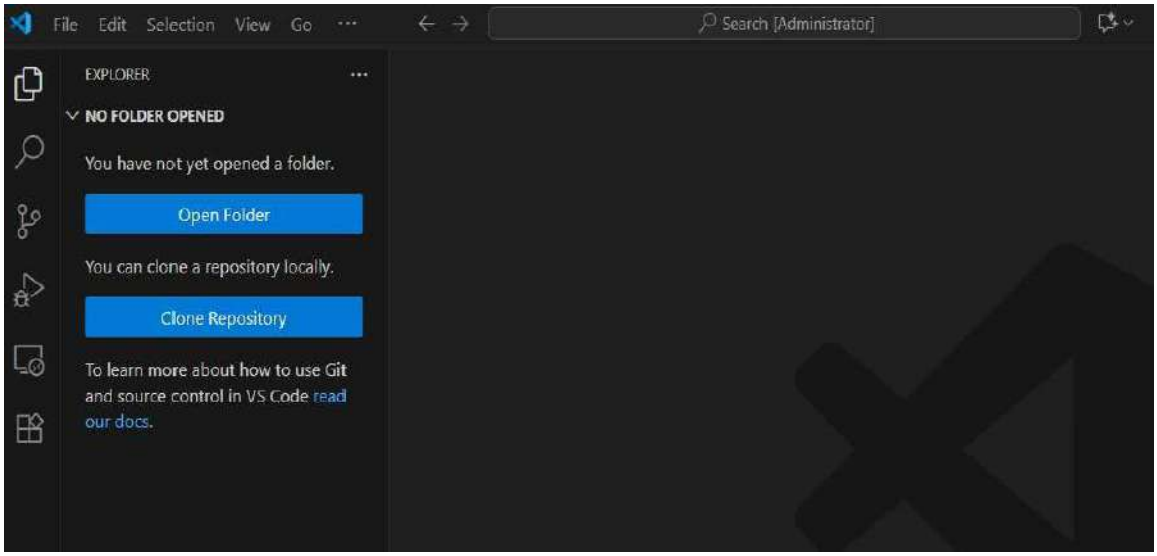
After downloading VS now, click on symbol(

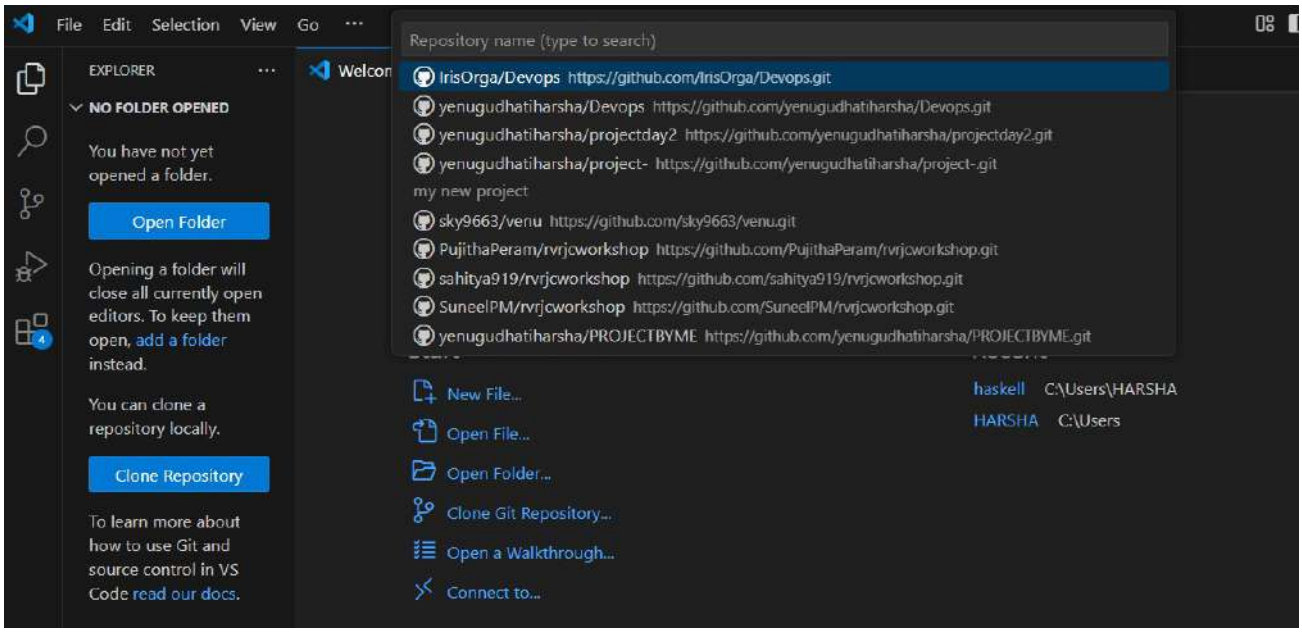
)

And then the window of VS will open and we can execute any commands we want.

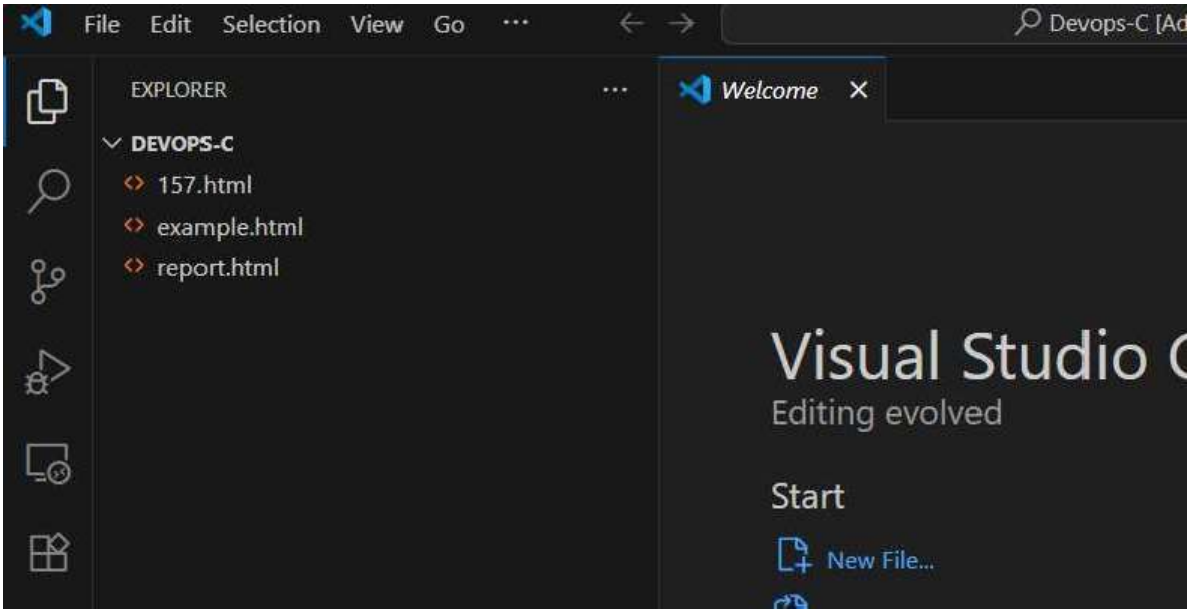


2. **Clone a Repository**: Open VS code and then go to file explorer. In **file Explorer-> Clone Repository.** Now at top in search bar click on github.

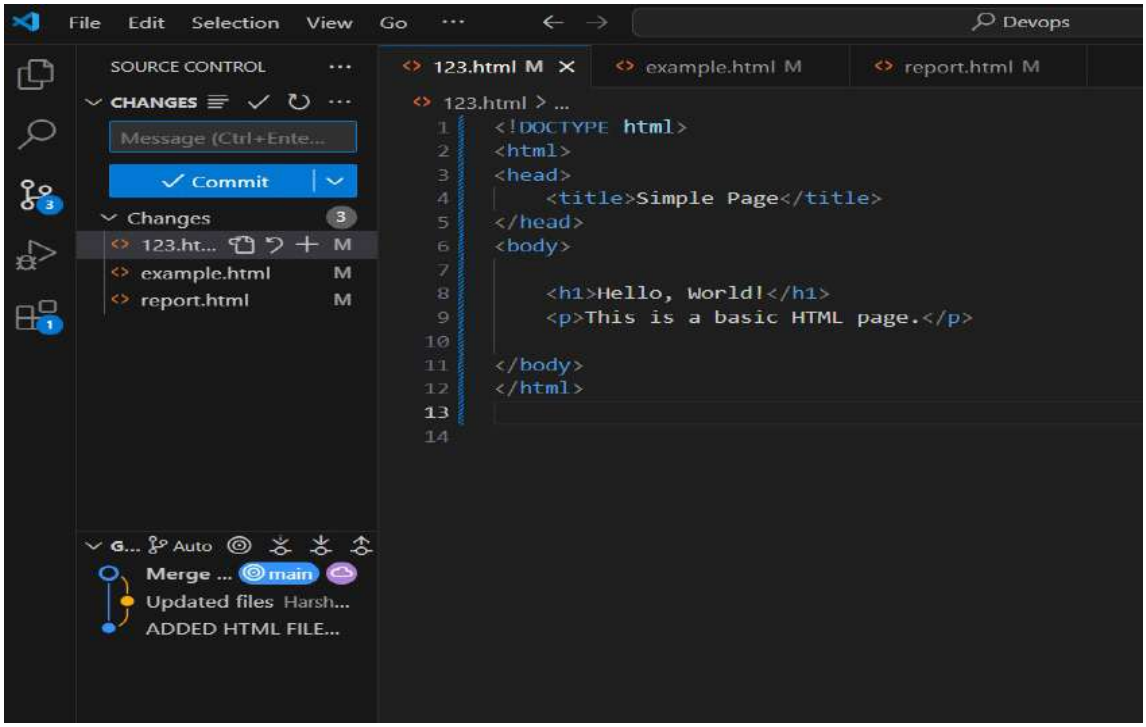




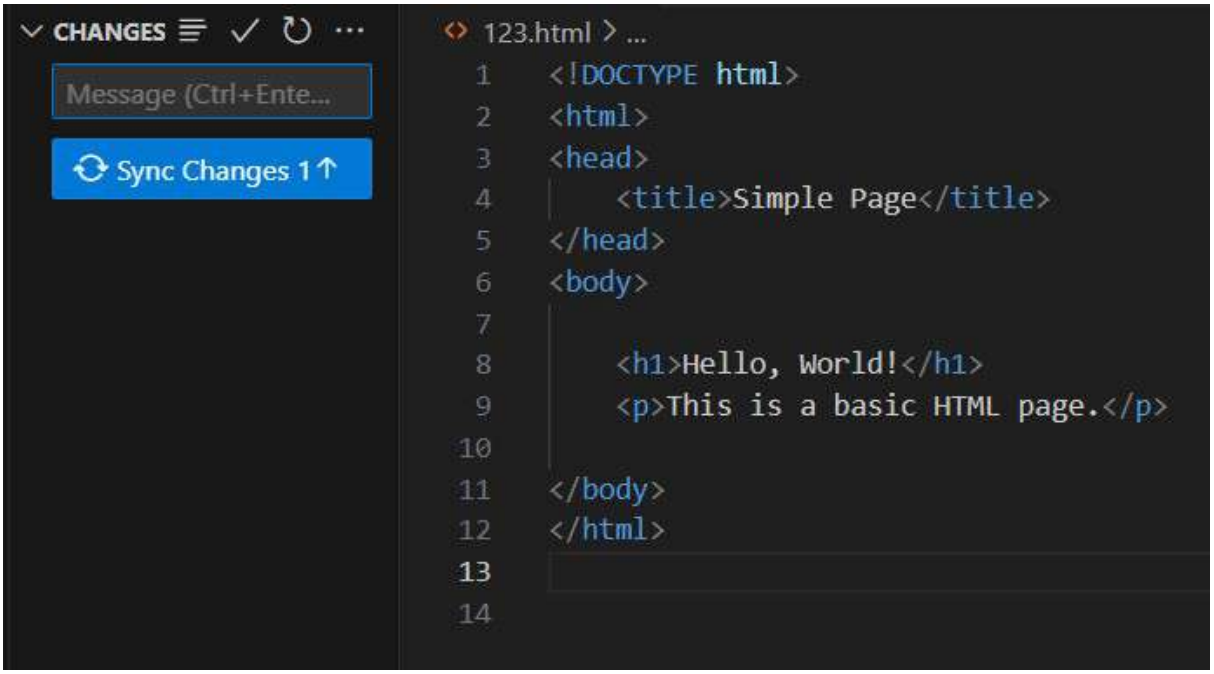
After clicking on any repository, you want then select the destination we want to place the cloned repository.



3. **Stage Changes:** Now click on any of file you want to change (for ex: 157.html) modify the file and then save the file. Now go to Source control and click on commit to save changes made to file.

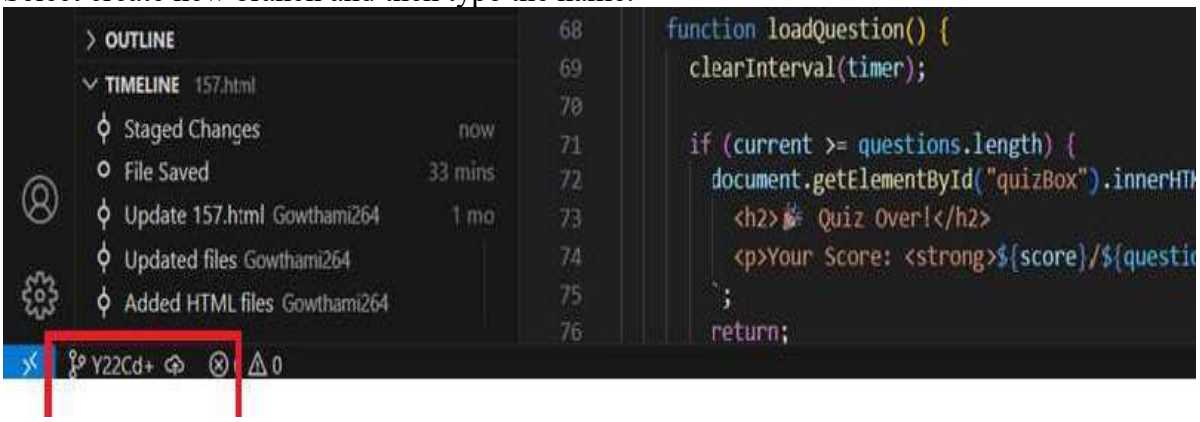


Click on drop down menu of commit button and then click on Commit & push so that the changes can be pushed into the file.



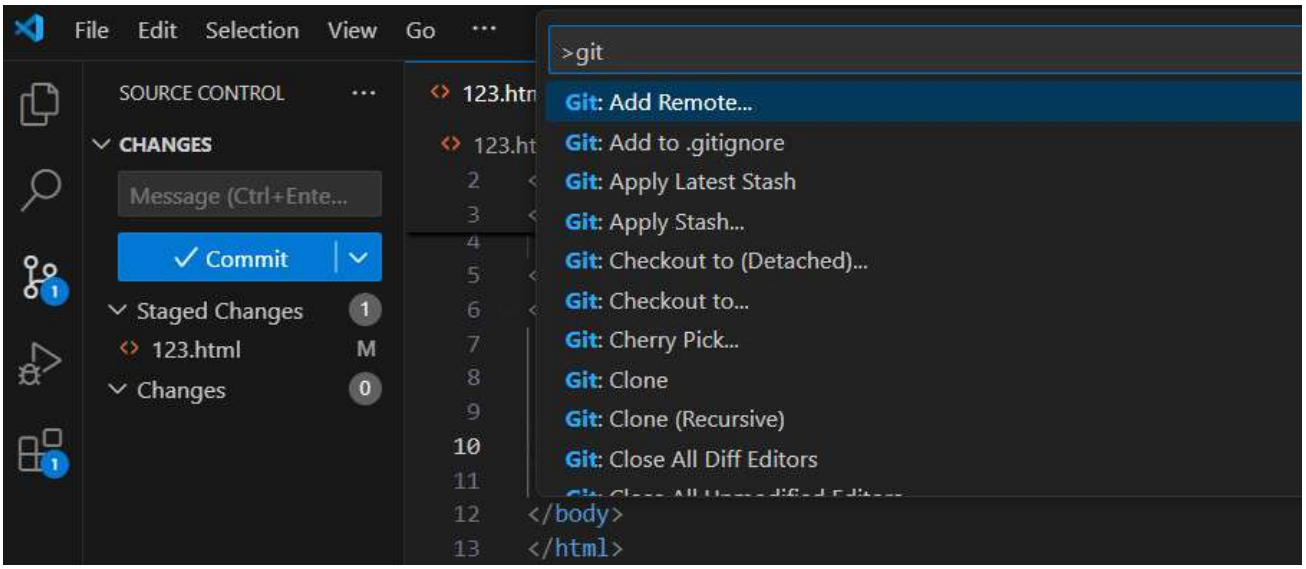
4. **Branch management:** At the bottom corner of Visual Studio click on branch we are currently working on (for ex: main)

Select create new branch and then type the name.

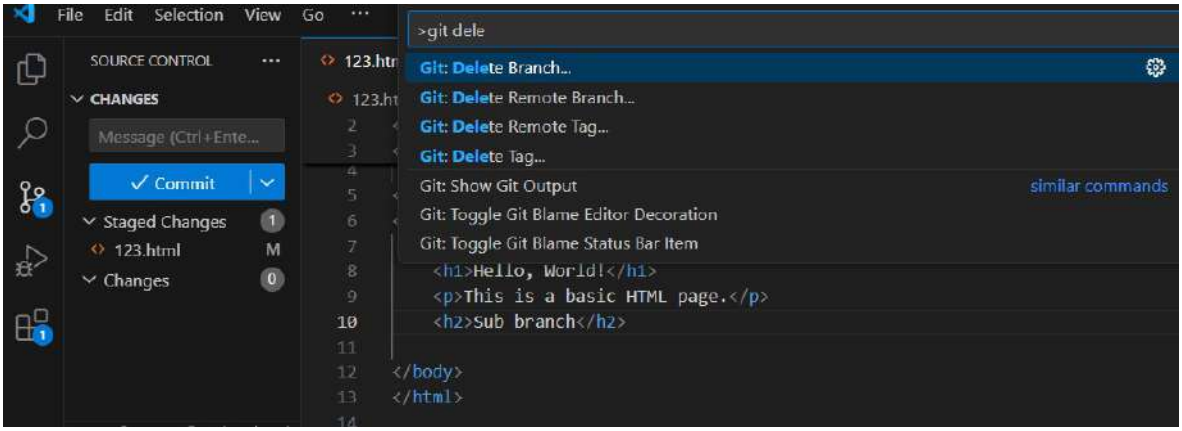


A new branch '**Y22cd**' was created here as shown in below image.

5. **Merging Branches**: Checkout from **main** branch then run the command palette **(ctrl+shift+p)** -> choose branch.



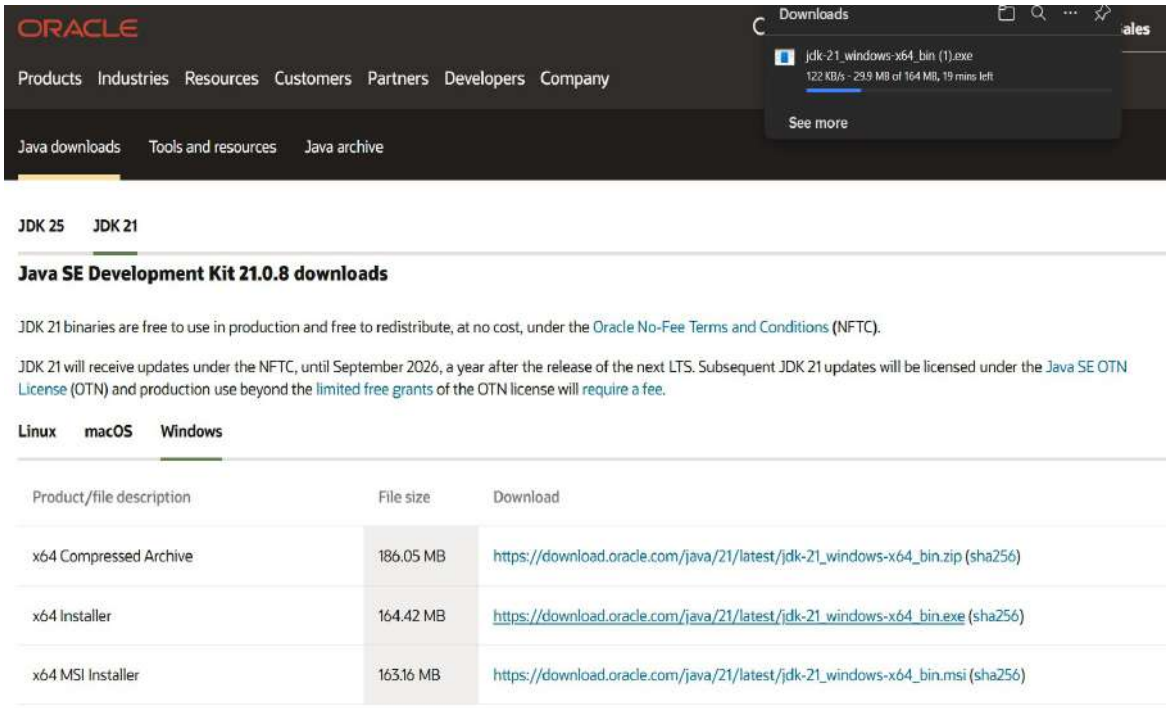
6. **Deleting a branch**: In order to delete a branch again run command palette (**Ctrl+Shift+p**) and then click on git delete branch to delete branches.



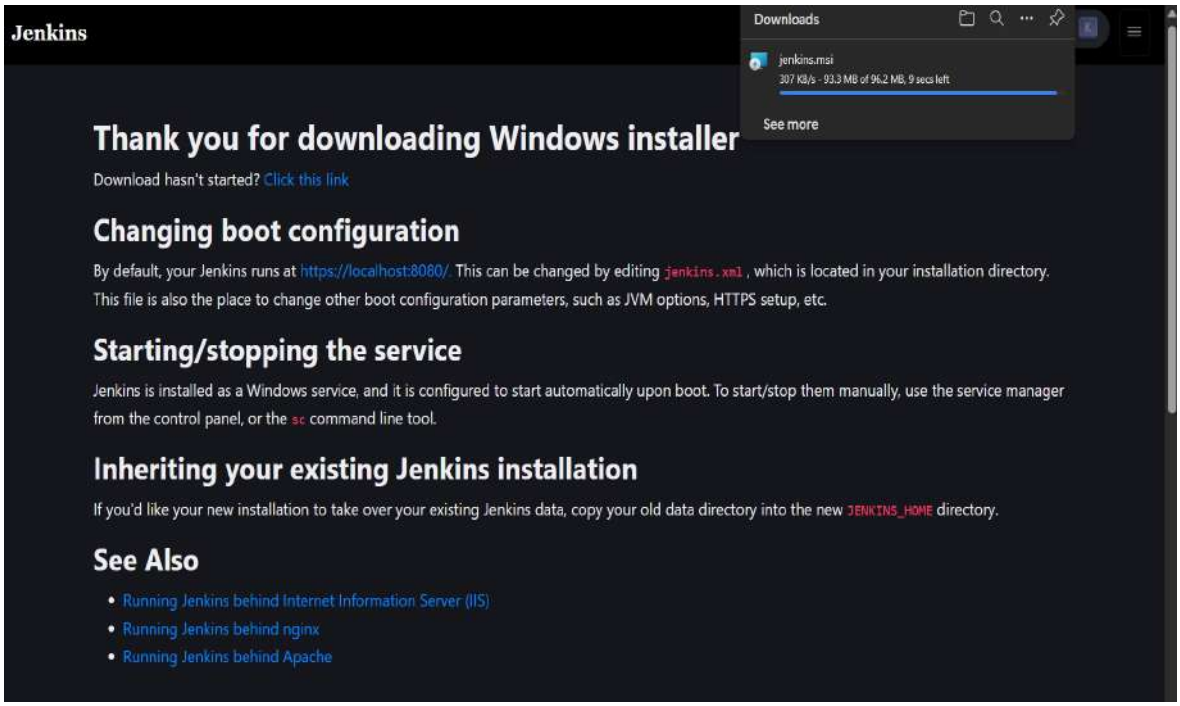
Experiment-6: Jenkins – Freestyle Project

Aim: To create and run a Freestyle project in Jenkins.

Step1: First we have to download the Java JDK 21 version as per your system requirements

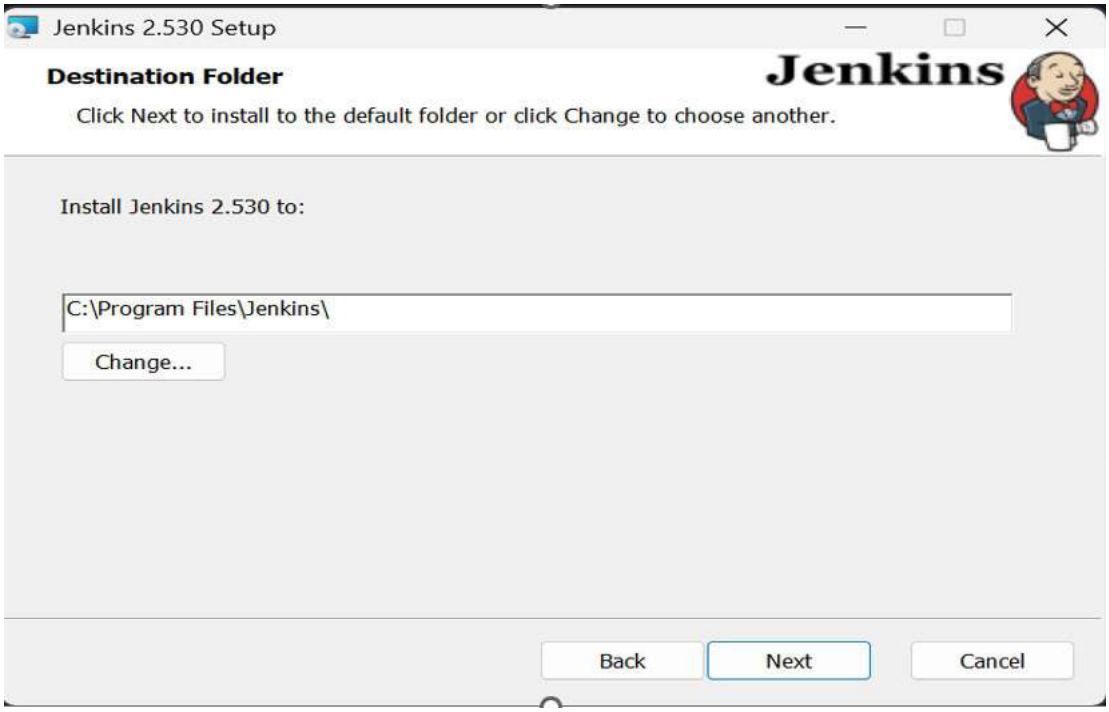


Step2: Then visit the https://www.jenkins.io/download/ and select windows then jenkins.msi file will download

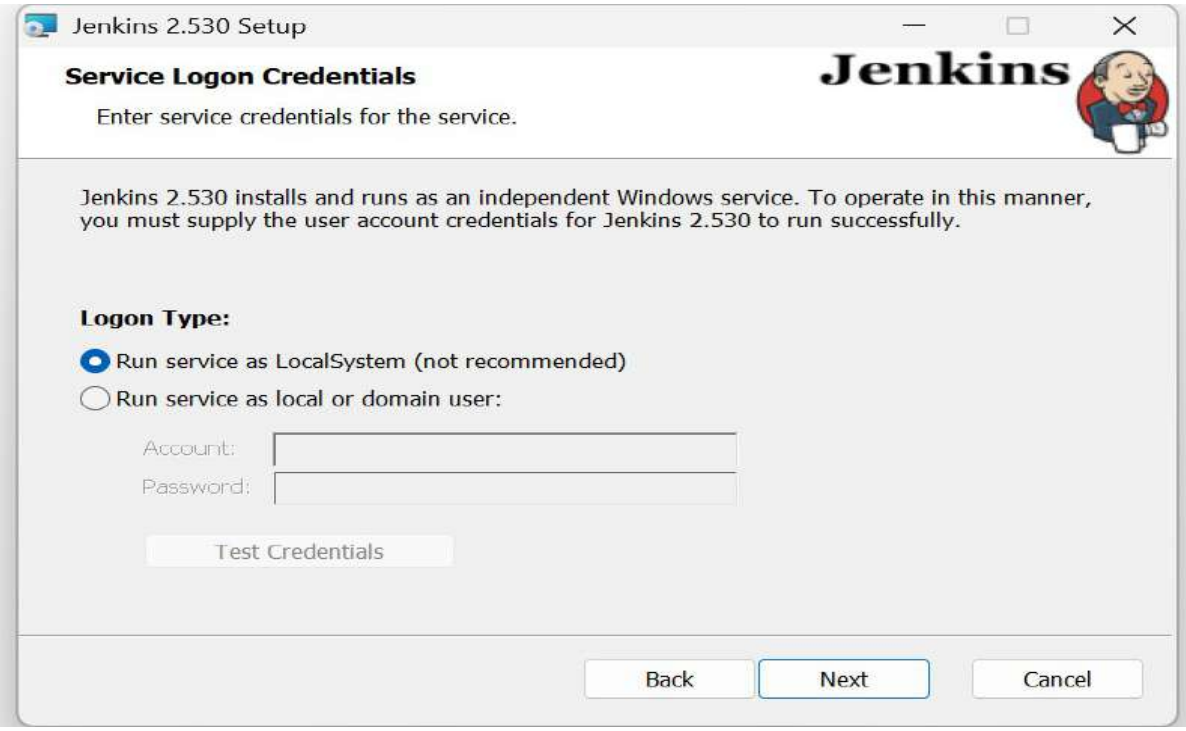


Step3: After downloading the jenkins.msi file then open the file and click on Next

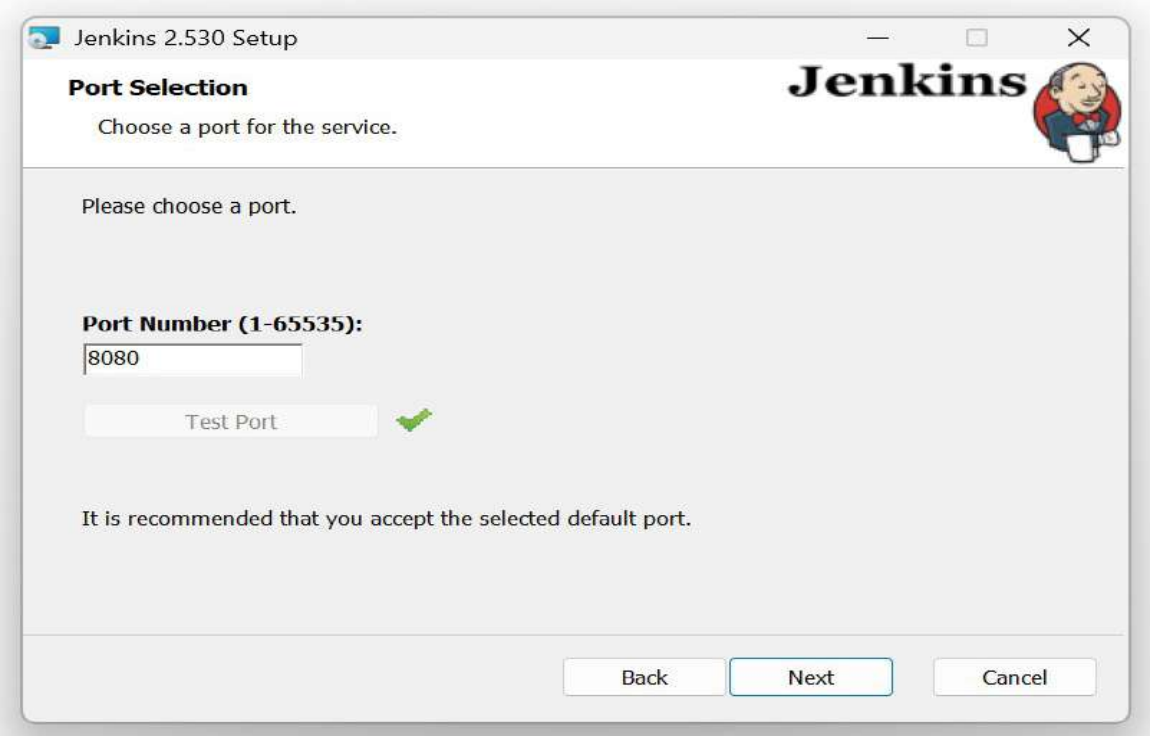
Step4: After clicking on Next then we have to setup the Jenkins by selecting the destination folder where to save the folder or else use the default destination folder and click on Next



Step5: Then we have to enter the Service Logon Credentials type, so select the Run service as LocalSystem (not recommend) and click on Next

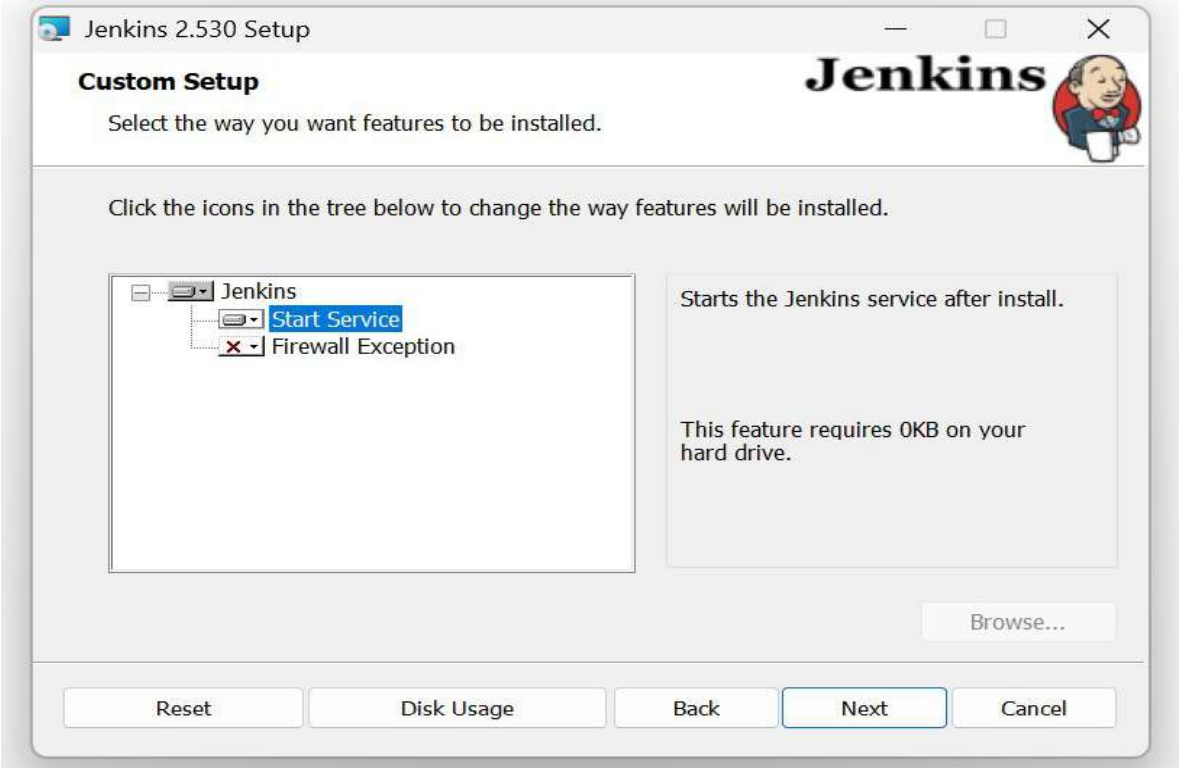


Step6: Now we have to select the Port Number, we can any number as our port but by default 8080 will be used as our Port Number and click on Test Port then click on Next

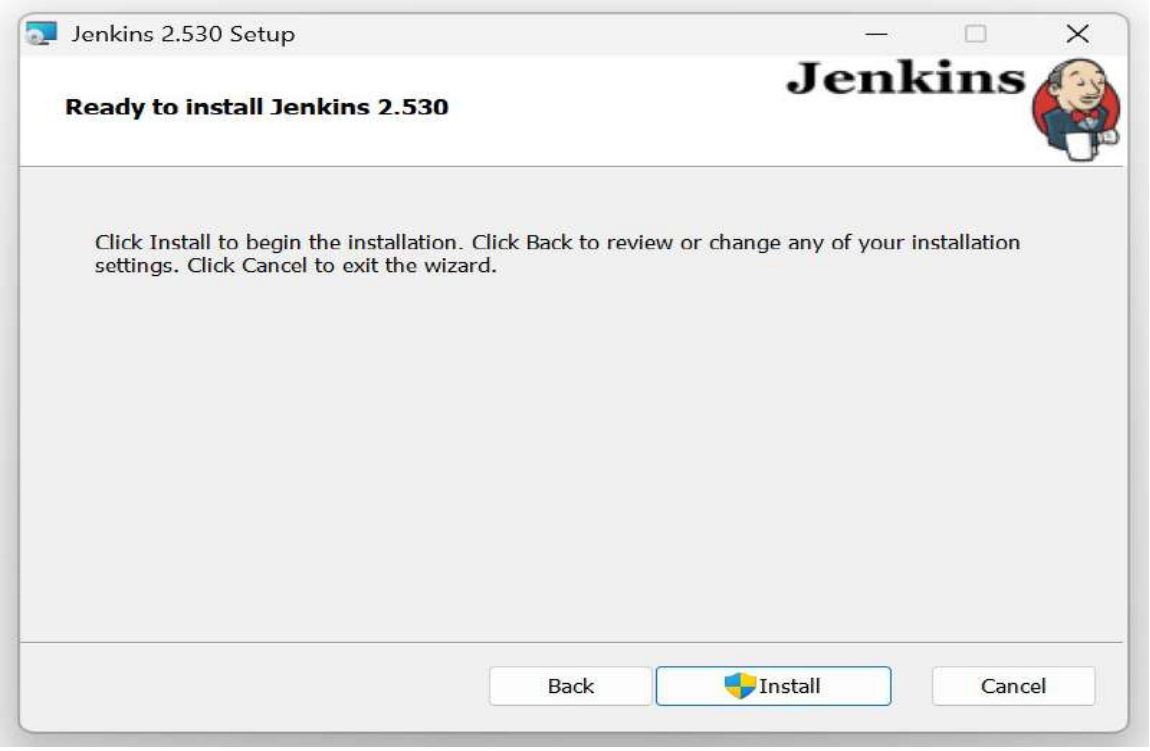


Step7: Next, we have to setup the Java JDK 21 version path which we installed earlier if the default path is not correct then you can change it then click on Next

Step8: Then we have to do the custom setup and select the start service and click on Next



Step9: Then it is ready to install Jenkins and finally click on Install



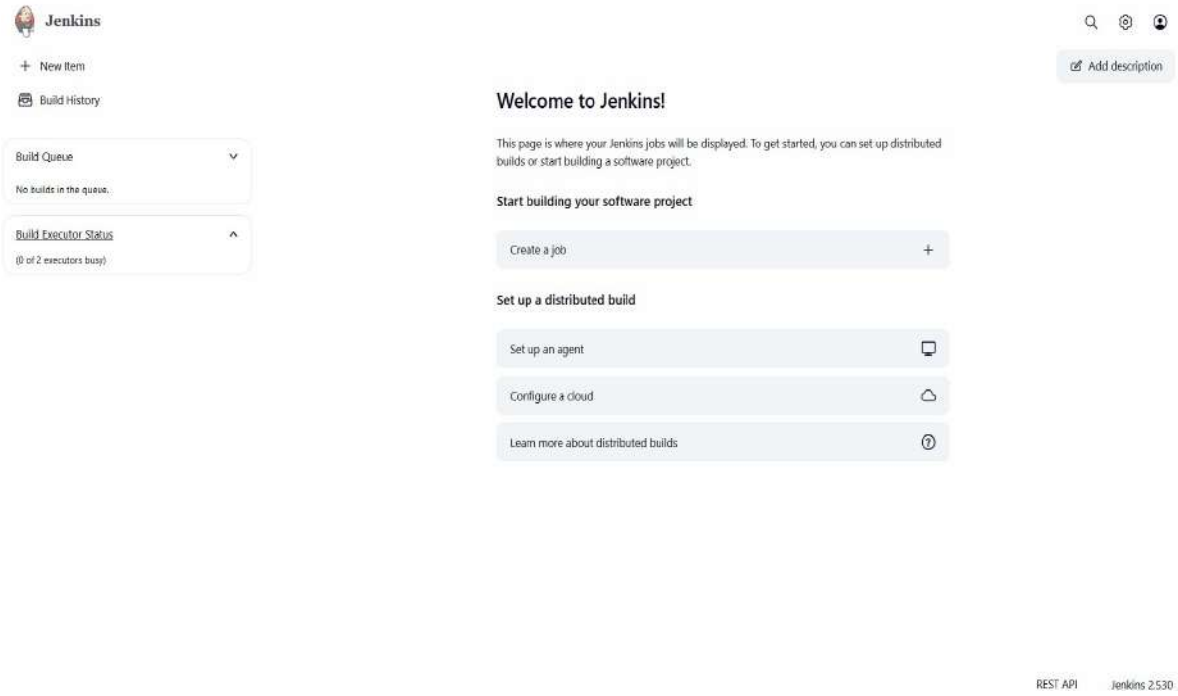
Step10: After clicking on Install, then all the Jenkins files will be installed then click on finish button then the Jenkins will be installed successfully

Step11: Now we have to open the installed Jenkins in the http://localhost:8080/

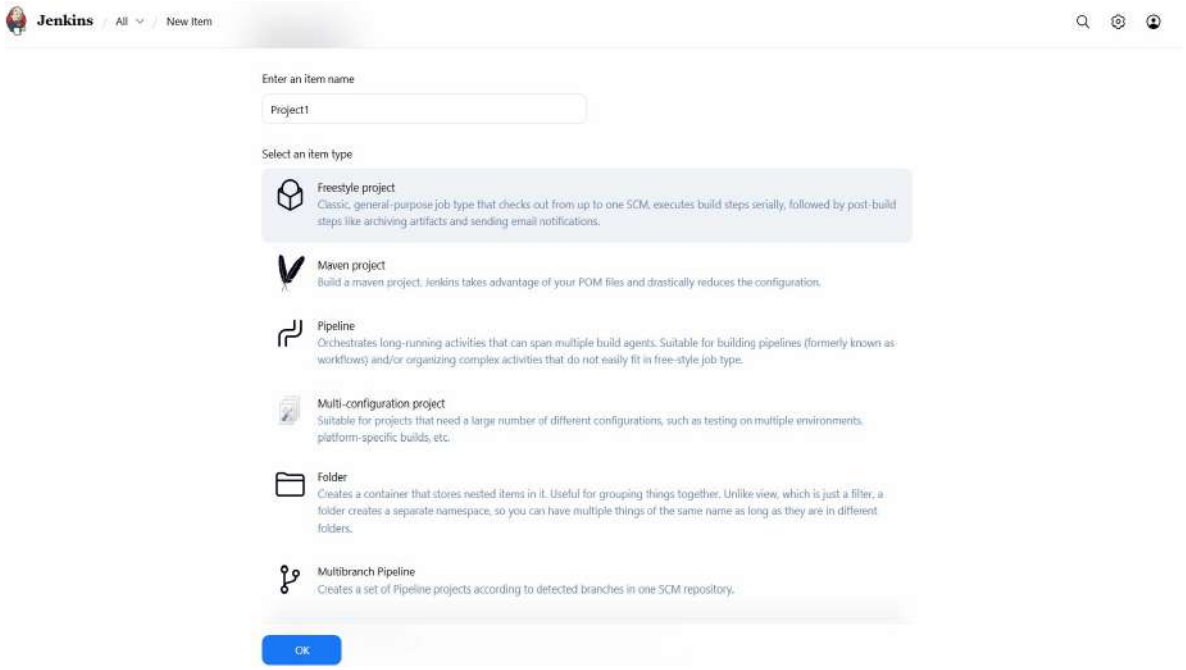
Step12: Next we have to give the Username and Password

For the Username, we can give our name or else give the user name as admin

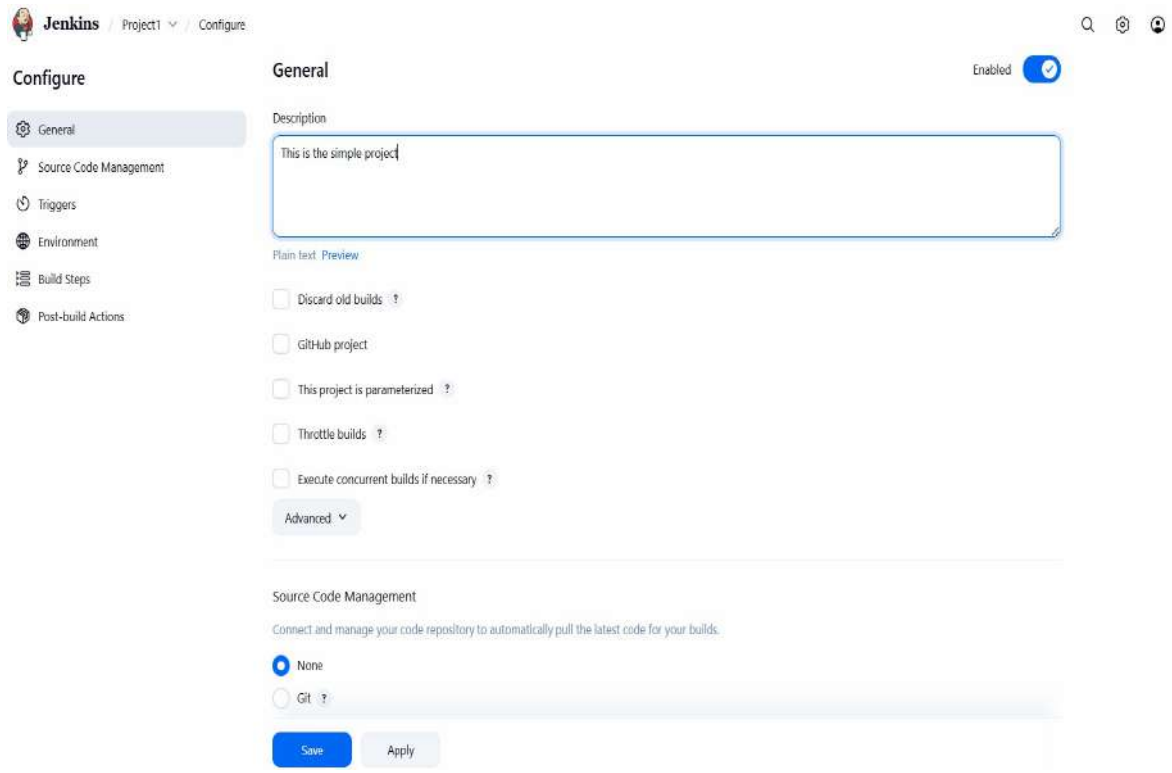
For the Password, we have to open the jenkins.err() file and copy the password to sign in



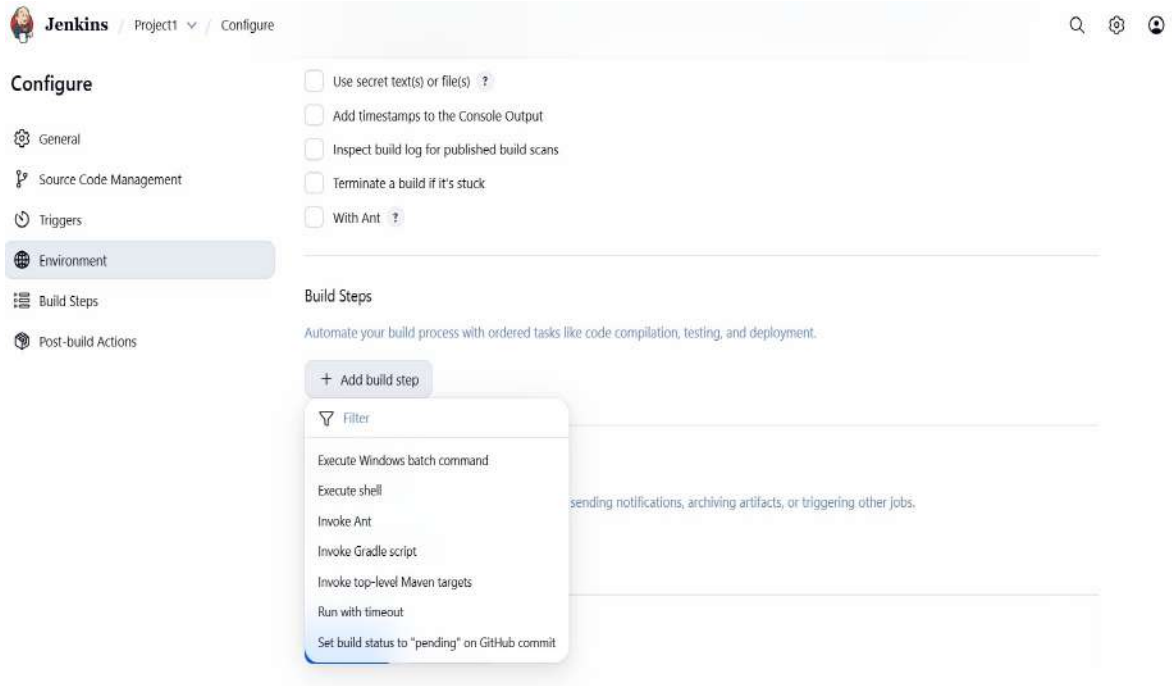
Step13: To build a simple project, we have to click on New Item and enter the project name and select the type of project like Freestyle Project and click on Ok button



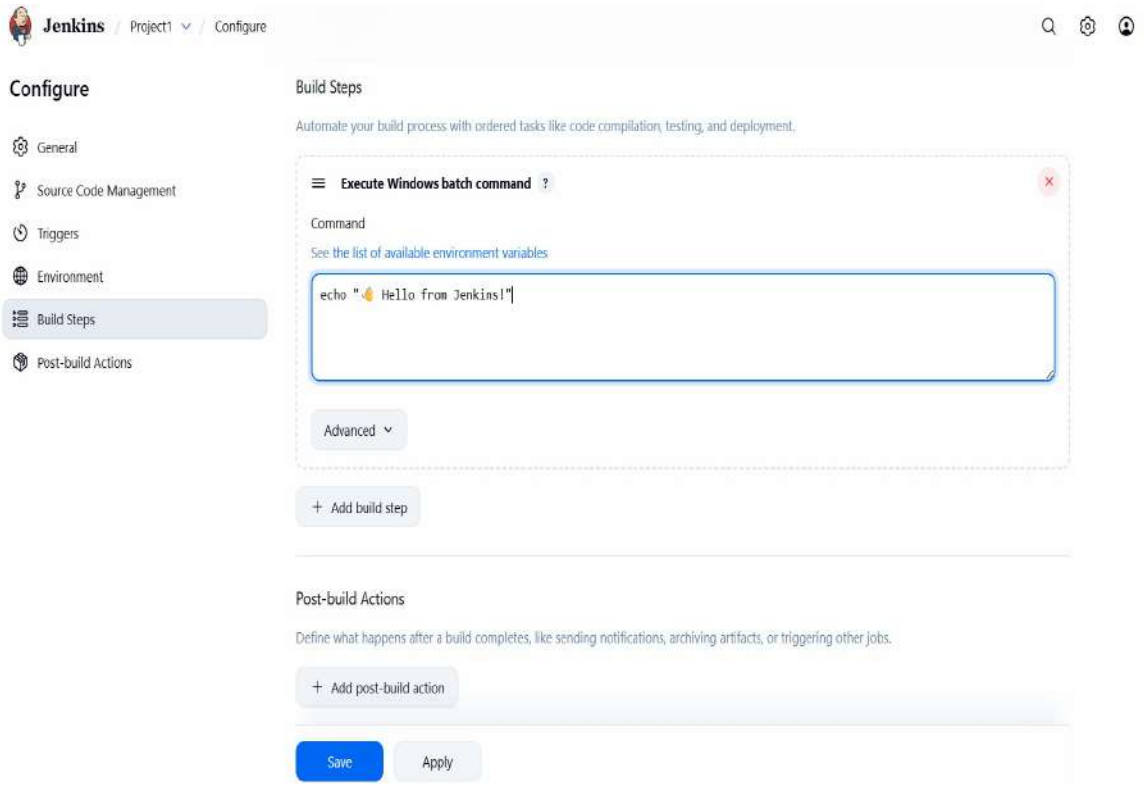
Step14: After clicking on Ok, then we have to give the simple description about our project and select the none in the Source Code Management



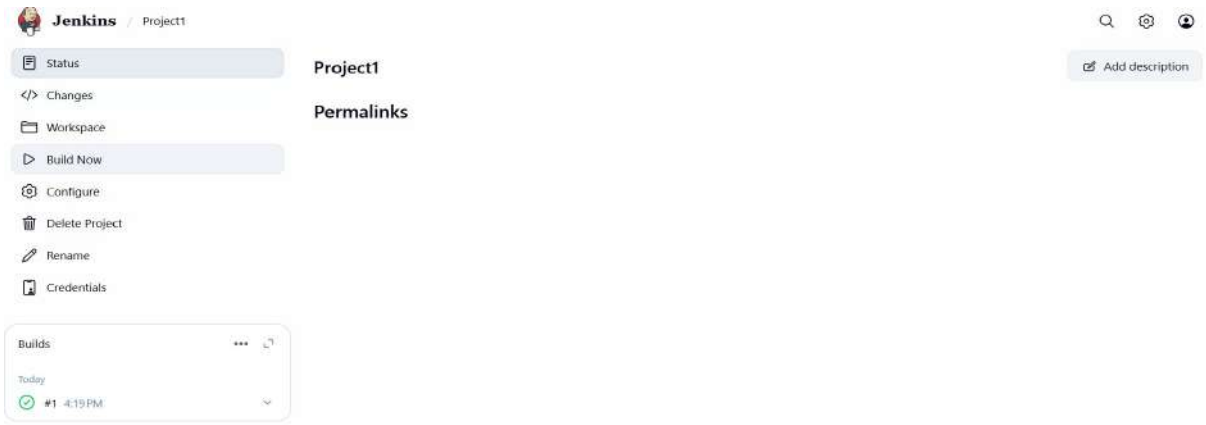
Step15: After that scroll down to the build step, and select the build step you want to execute like Execute Windows batch command



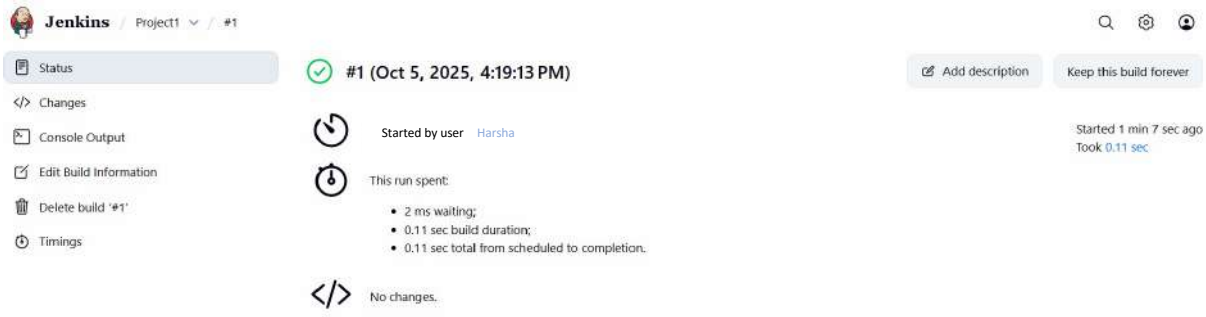
Step16: And write the simple code to execute the project and click on apply then click on Save



Step17: Then click on build now, to build the project



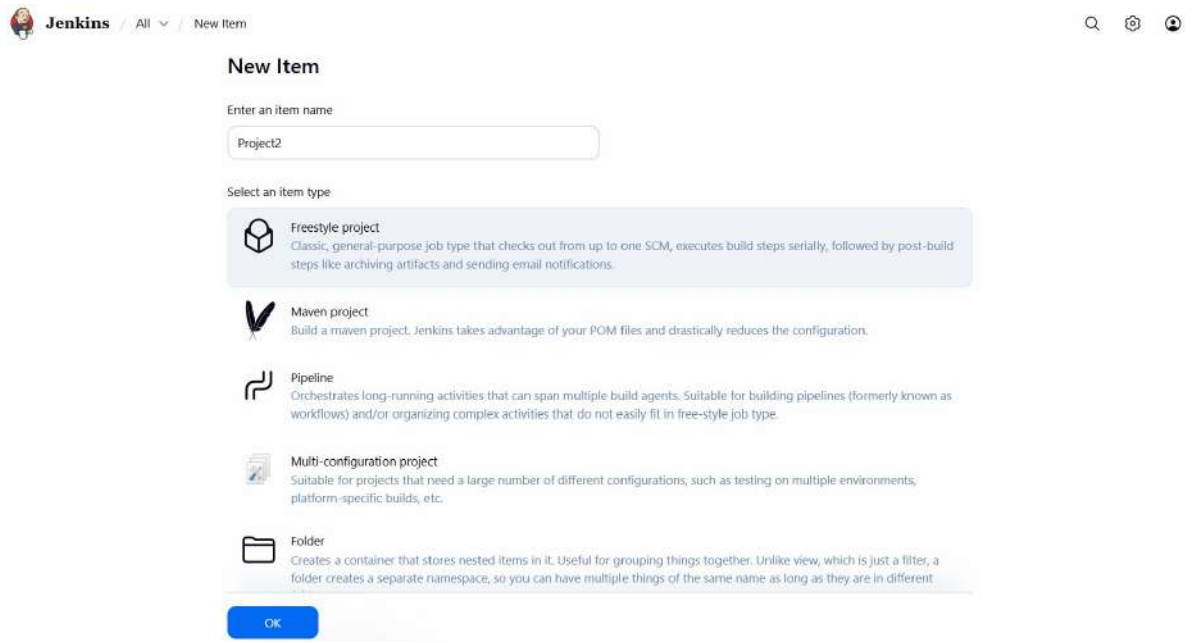
Step18: We can check the status of our project whether the build is failed or success



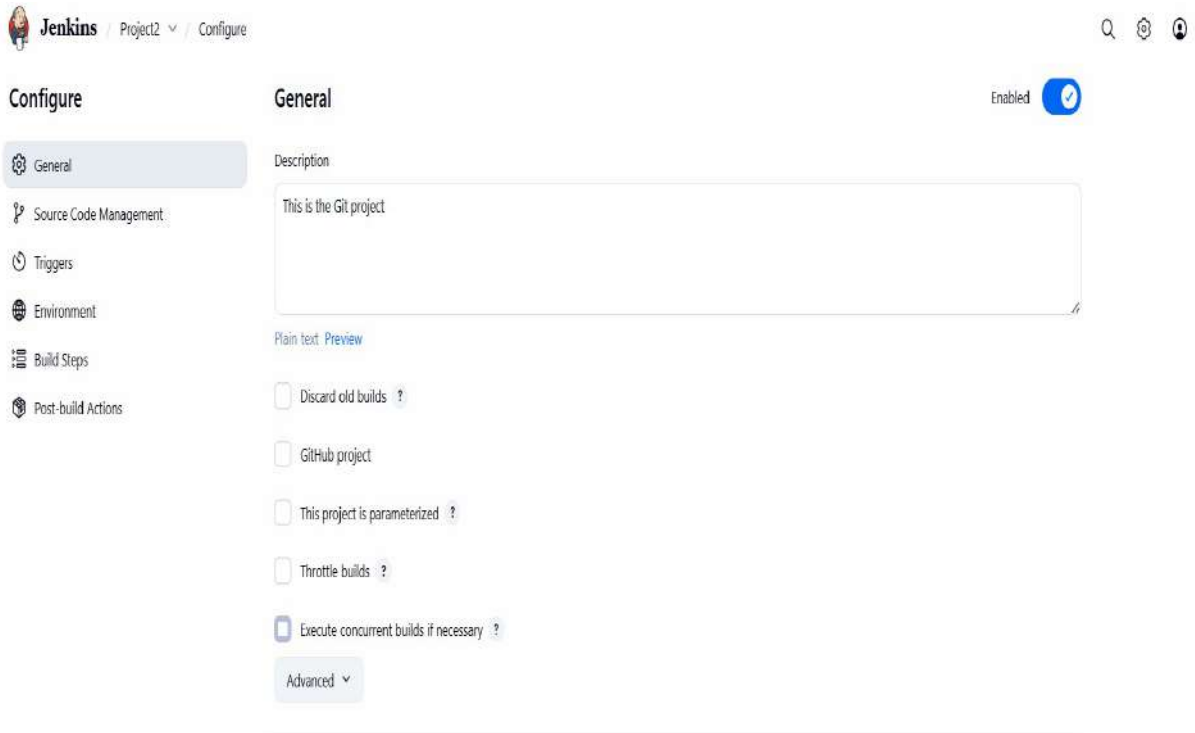
Step19: We can also see the output of our project by clicking on Console Output, it will displays the entire information of our project

Step20: Now we should create another project by using the git as Source Code Management

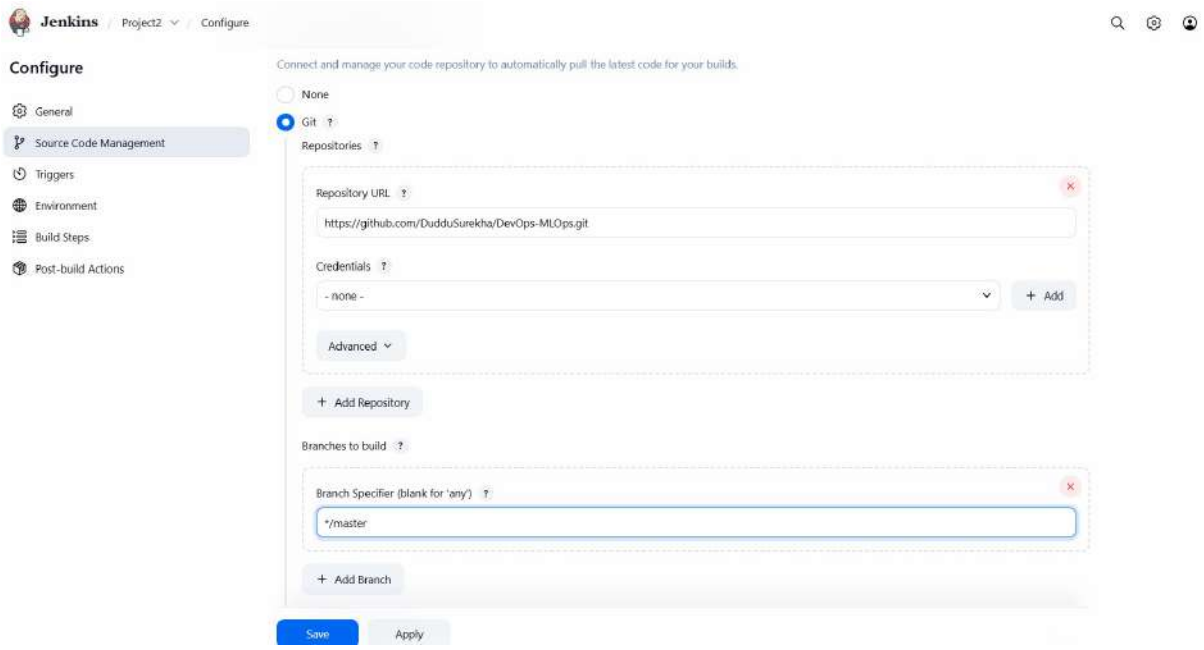
First we should create the Freestyle project and give the name of the project



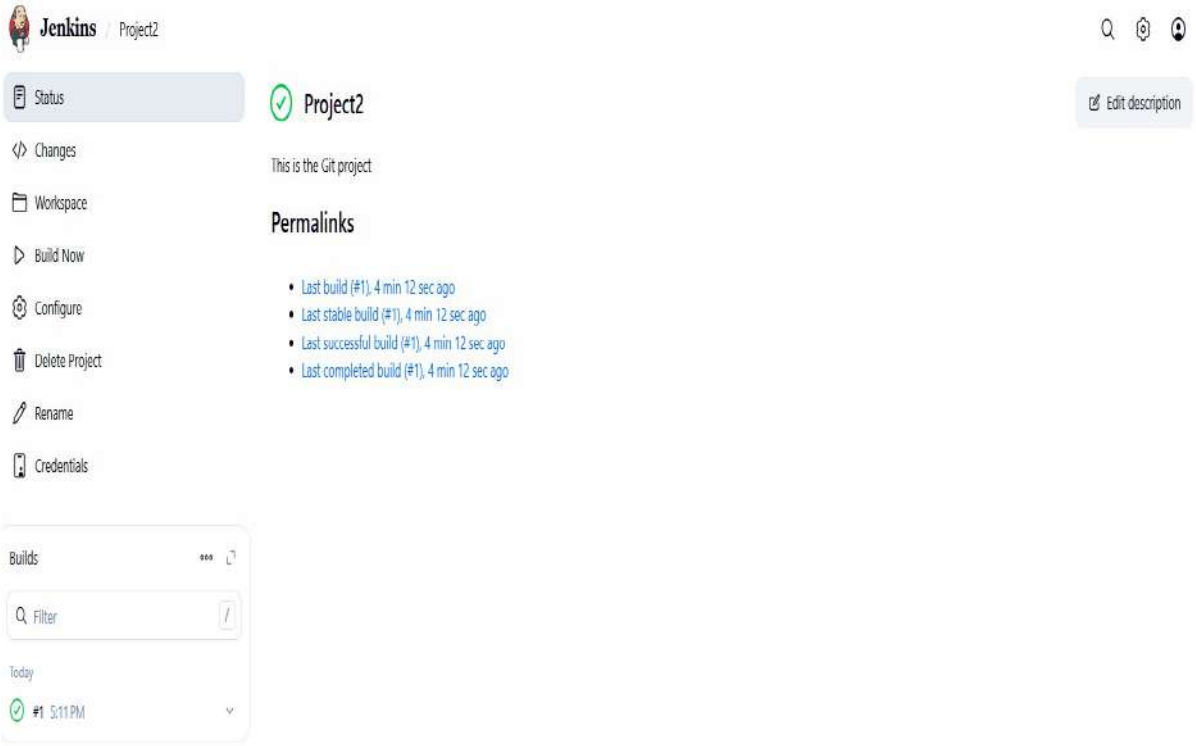
Step21: Then give the simple description about our project



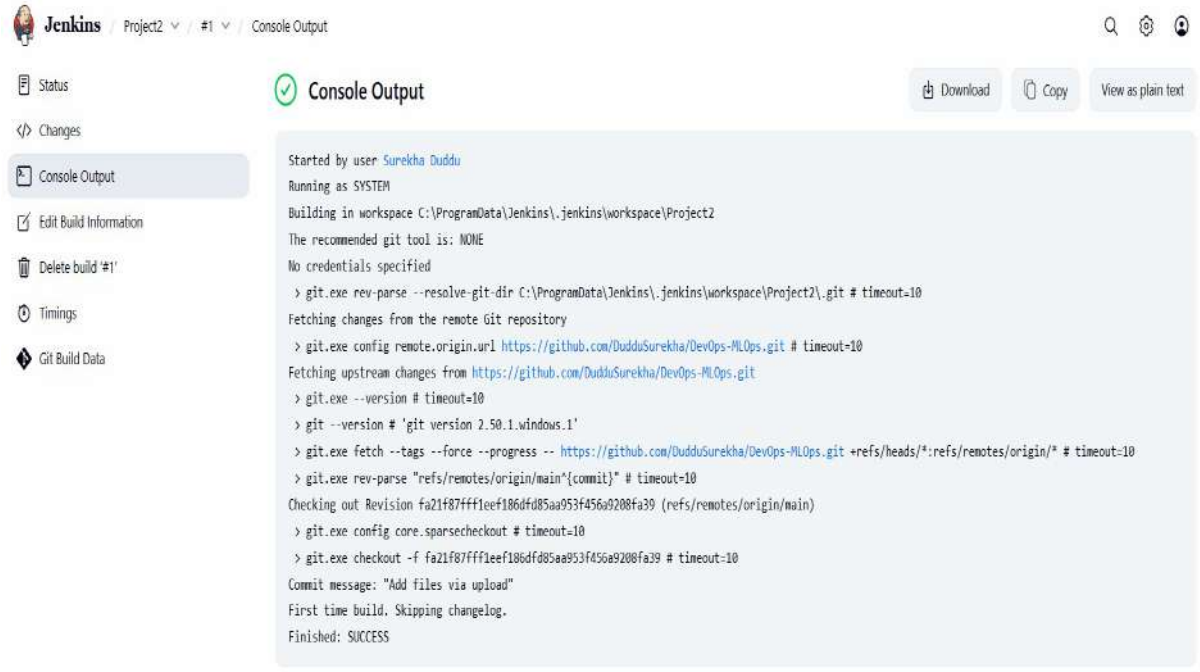
Step22: In the Source Code Management, select the git and give the git repository URL and specify the branch according to your git repository, by default it specifies the master as a branch



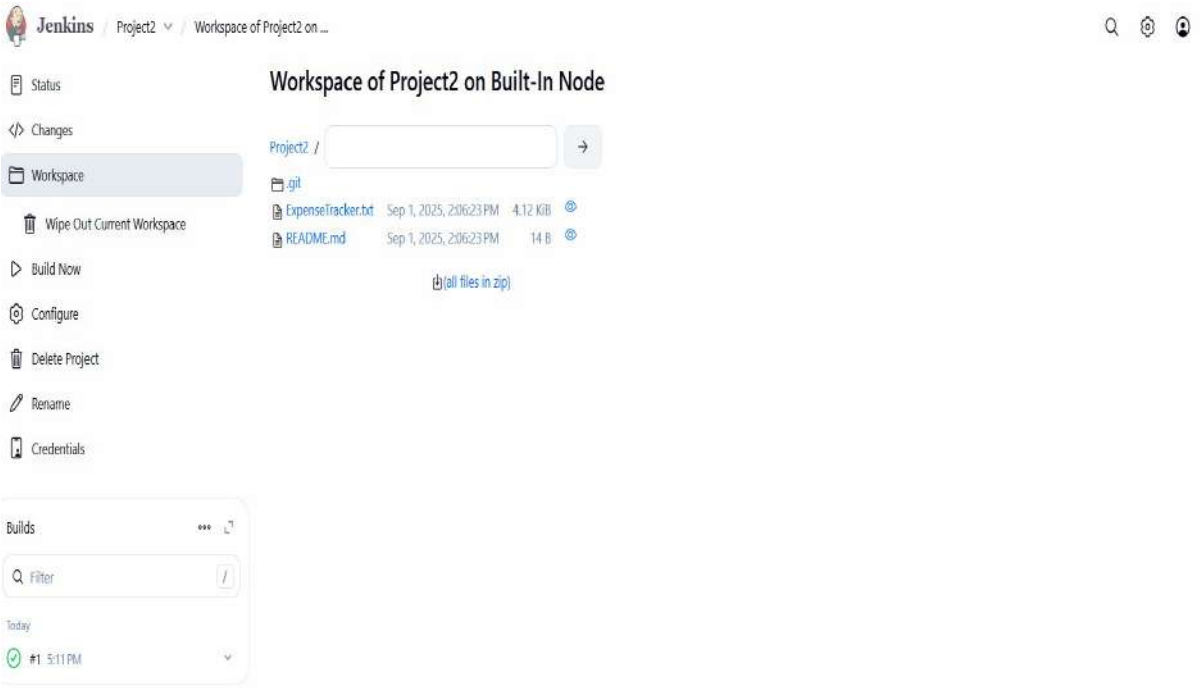
Step23: After that click on Apply then Save, and we have to build our project2 by clicking on Build Now and check the status of the build whether the build is failed or success



Step24: And we can know the entire output information by clicking on the Console Output whether the git cloned all the information from our git repository or not can be verified



Step25: We can see the files that git cloned into our local repository by clicking on the Workspace



Step26: We can also view the entire code that the files contain by clicking on the eye icon beside the file name

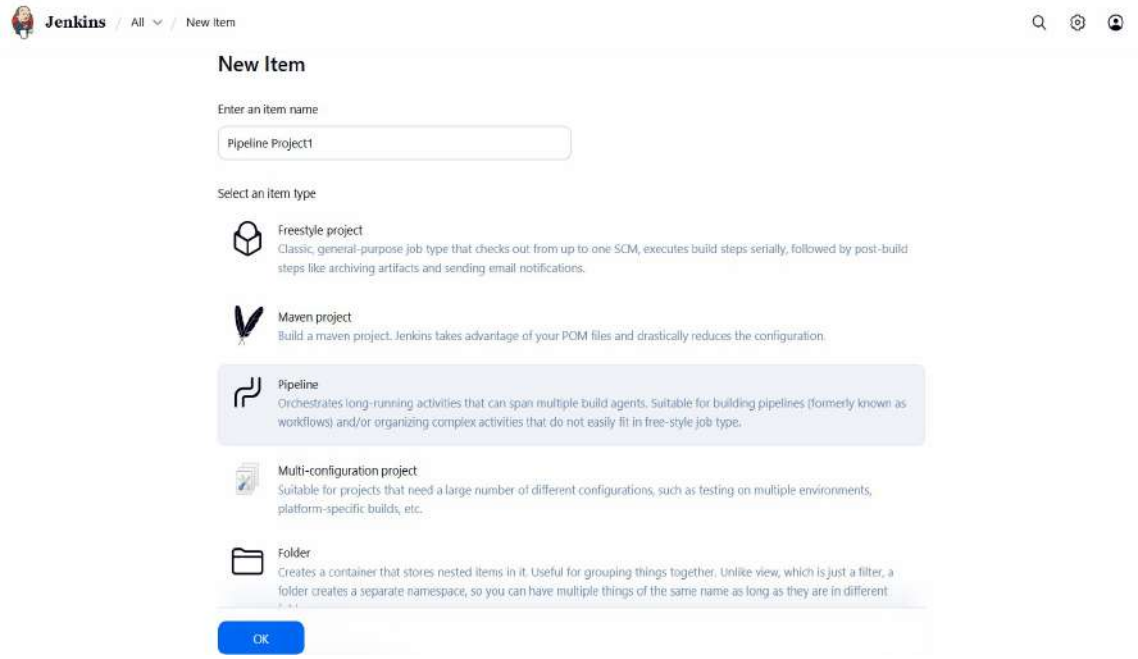

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>Expense Tracker</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f2f2f2;
      margin: 0;
      padding: 0;
    }
    .container {
      max-width: 700px;
      margin: 40px auto;
      background: white;
      padding: 20px;
      box-shadow: 0 4px 8px rgba(0,0,0,0.1);
      border-radius: 8px;
    }
    h1 {
      text-align: center;
      color: #333;
    }
    form {
      display: flex;
      flex-wrap: wrap;
      gap: 10px;
      margin-bottom: 20px;
    }
    input, select, button {
      padding: 10px;
      font-size: 16px;
    }
    input[type="text"], input[type="number"], input[type="date"], select {
      flex: 1 1 200px;
    }
    button {
      background-color: #28a745;
      border: none;
      color: white;
      cursor: pointer;
    }
    button:hover {
      background-color: #218838;
    }
    table {
      width: 100%;
      border-collapse: collapse;
      margin-bottom: 10px;
```

Step27: Overall we build 2 simple projects, project1 without Source Code Management and Project2 using git

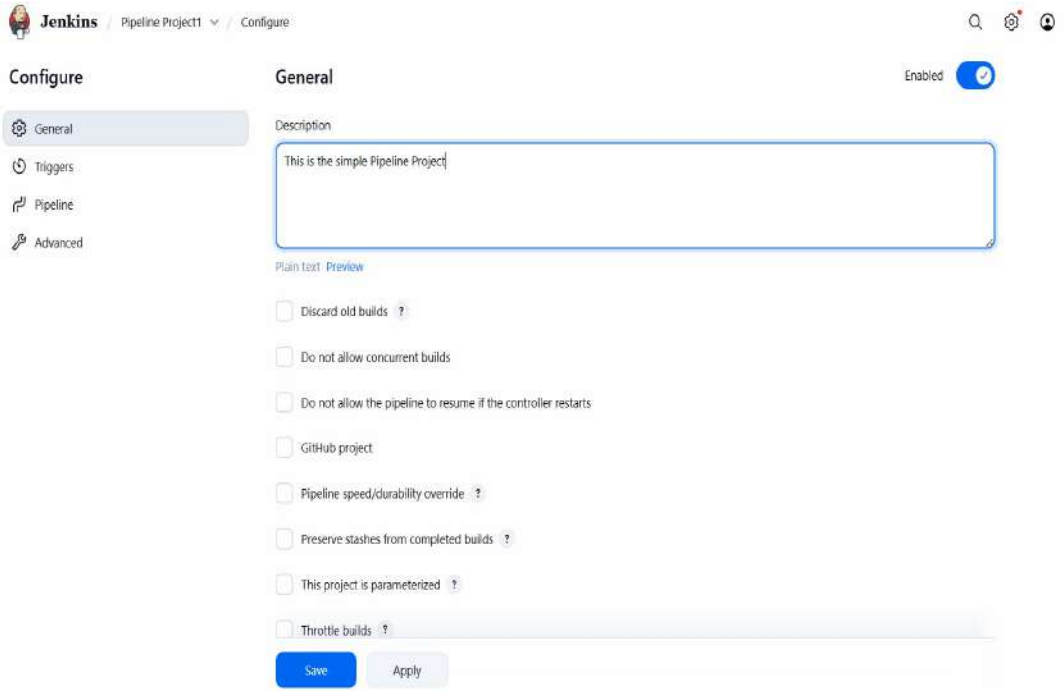
Experiment-7: Jenkins – Pipeline Project

Aim: To create and execute a Jenkins Pipeline project.

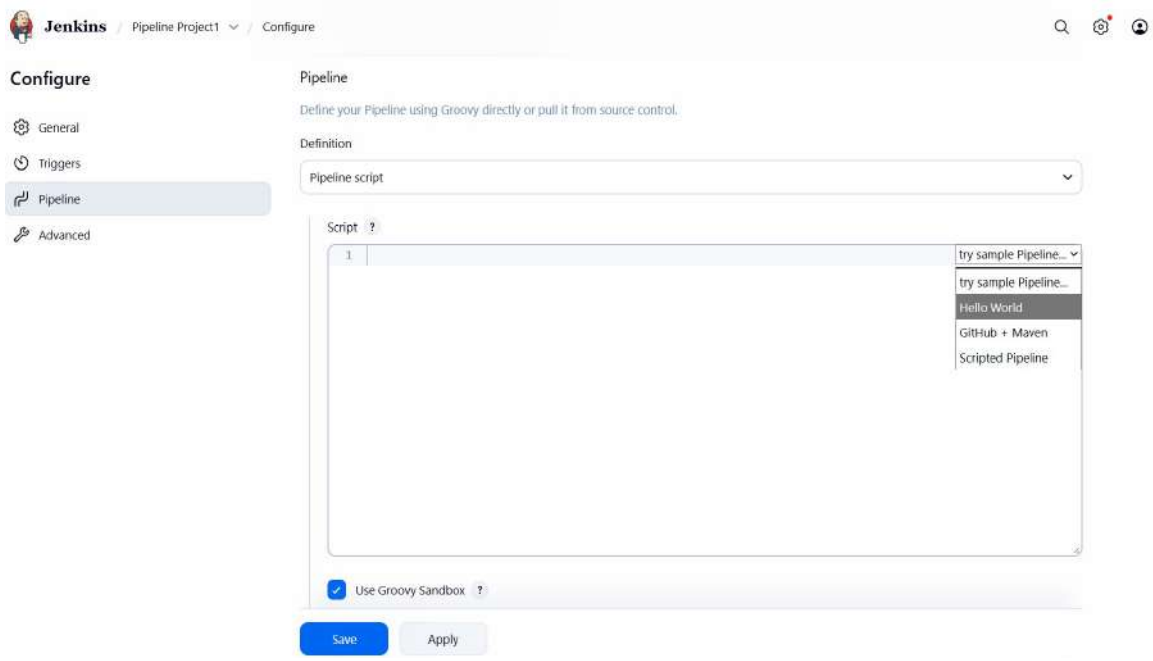
Step1: First, we have click on New Item and name the project and select the type of the project as Pipeline Project and click on Ok to create the project



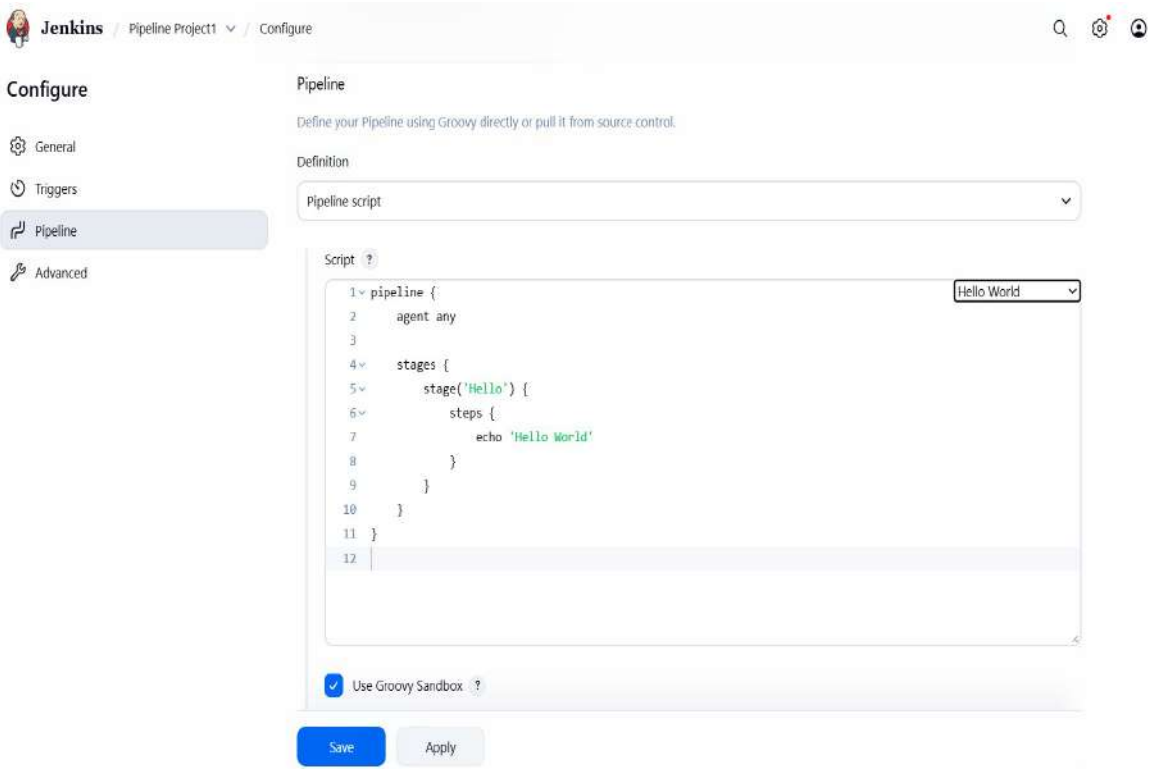
Step2: After creating the project, then give the simple description for our pipeline project



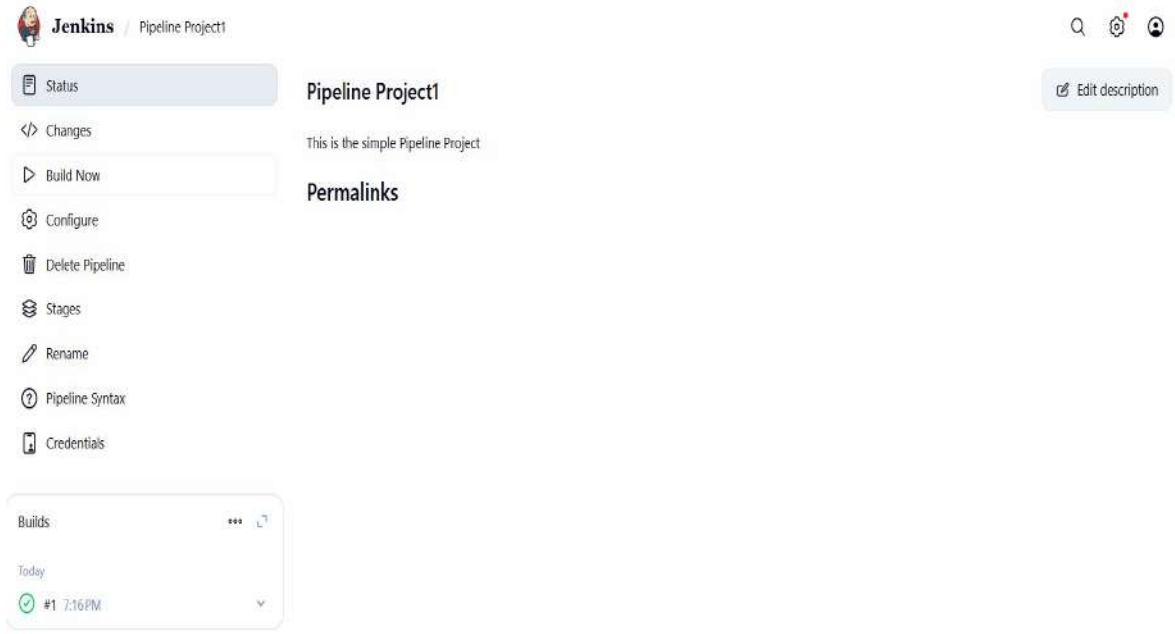
Step3: Then Scroll down to the Pipeline section and write your own pipeline code or else we can use the pipeline code directly using the Groovy



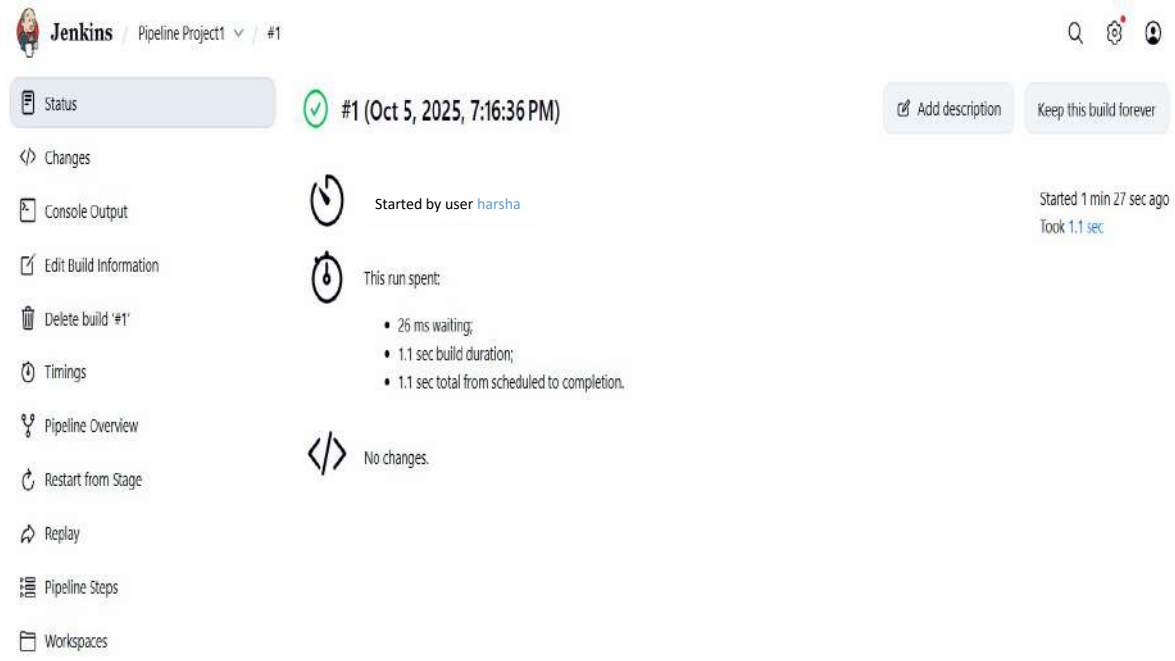
Step4: Select any pipeline code directly using Groovy from the dropdown, for example I am selecting the Hello World program from dropdown and click on Apply then Save



Step5: After saving the project, we have to build the project by clicking on Build Now for building our pipeline project

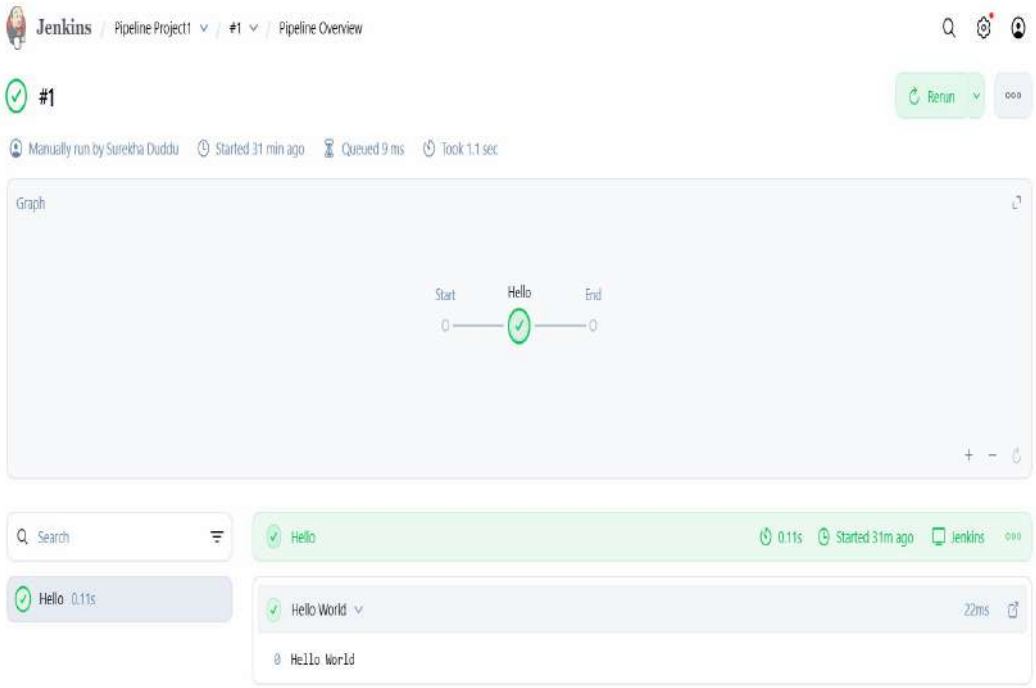


Step6: By clicking on the builds which created now, then we get the status of the build whether it is success or failed



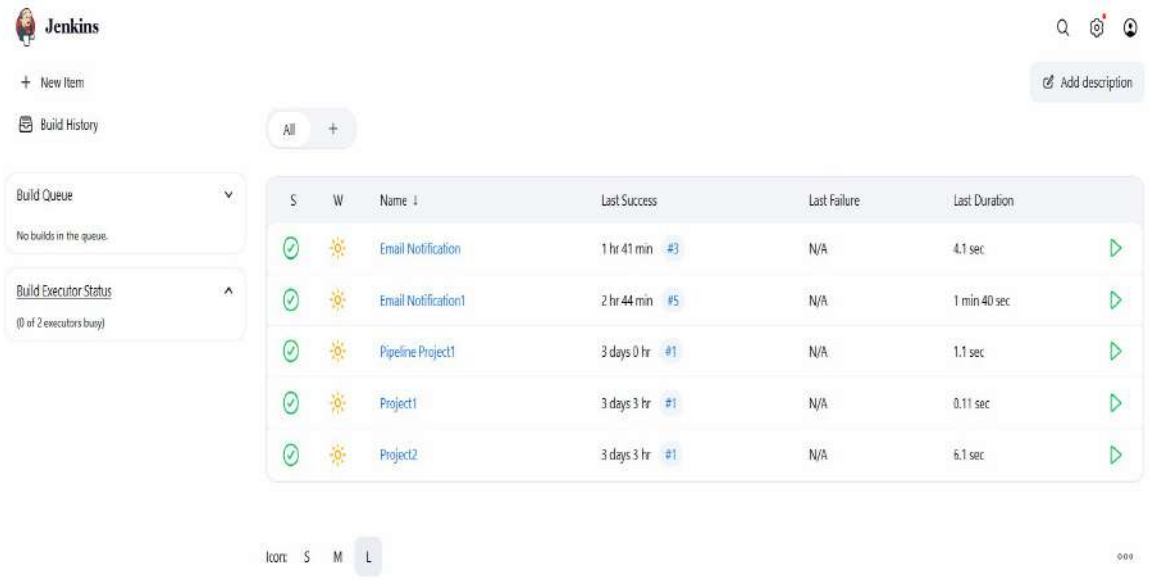
Step7: We can view the entire output by clicking on the Console Output it specifies whether our pipeline project is success or failure

Step8: We can also view the overview of our pipeline project by clicking it on the Pipeline Overview and it gives the output for our Pipeline project

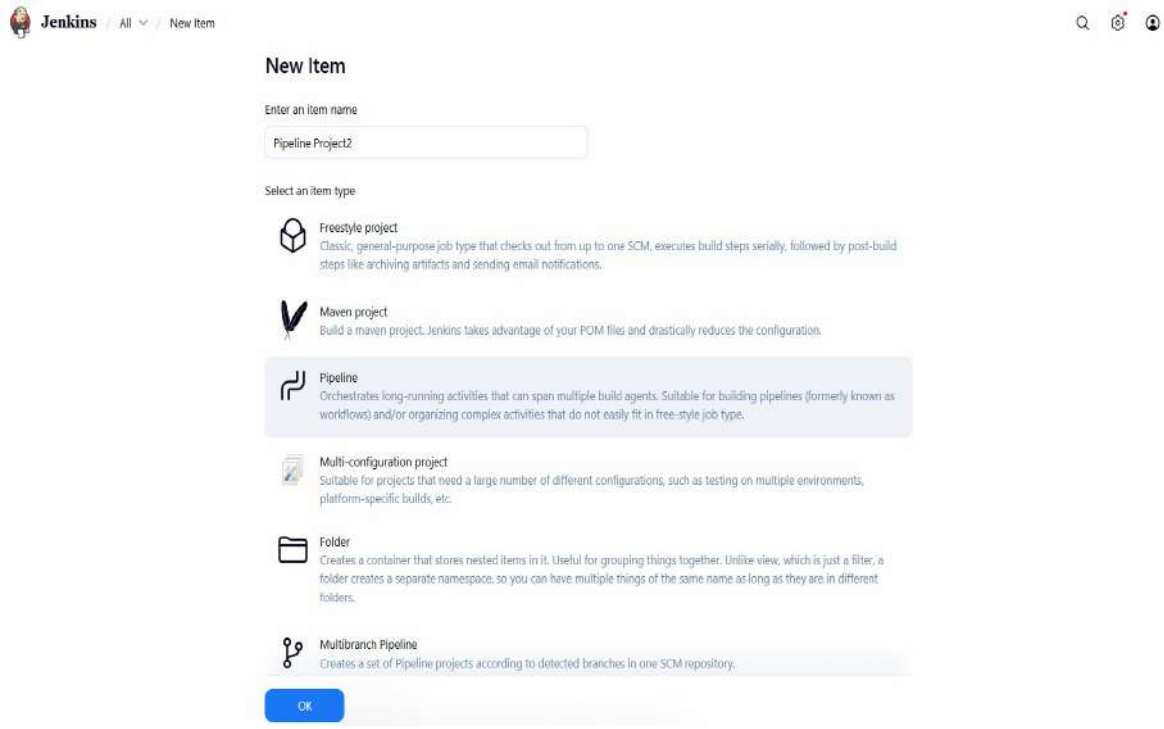


#Pipeline Project using Jenkins with Source Code Management (SCM)

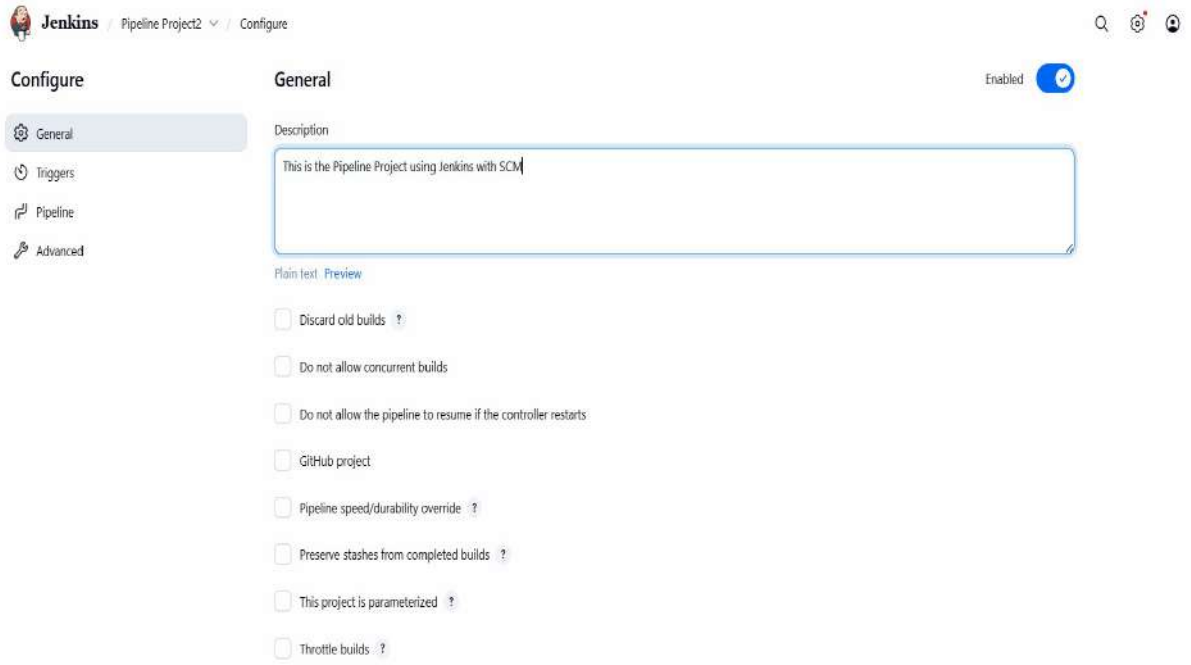
Step1: Firstly, we have to create the Pipeline Project by clicking on +New Item in the Home Page



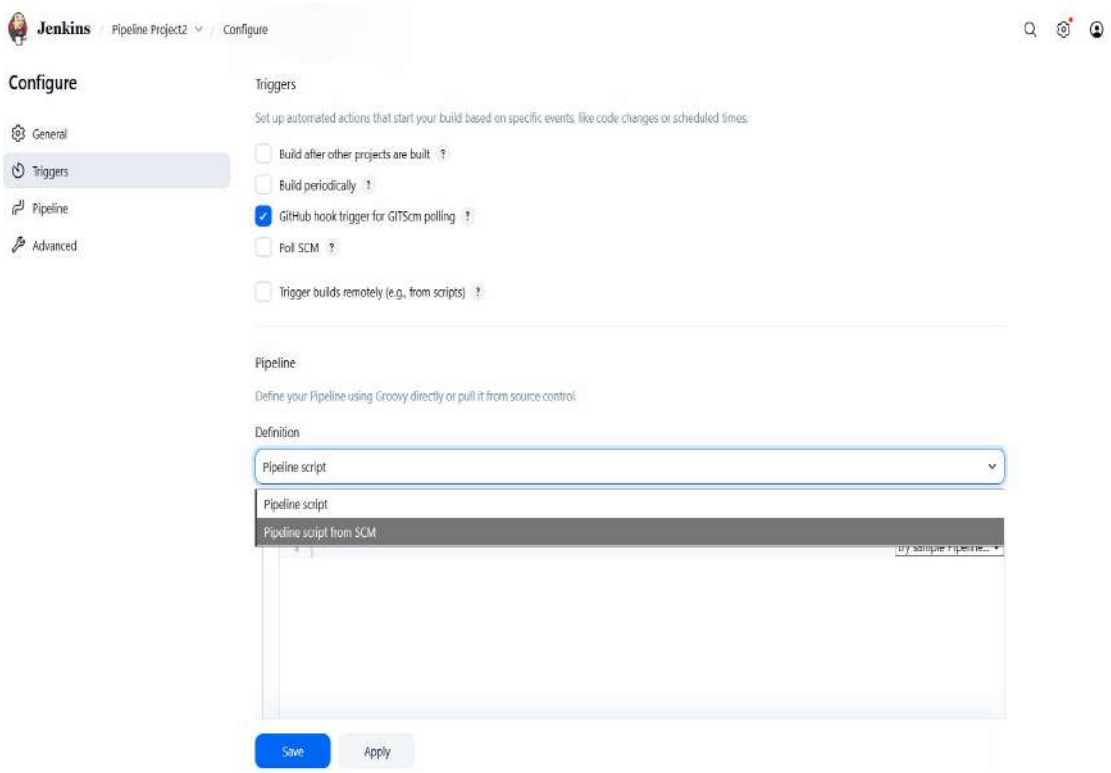
Step2: Then name the project and select the type of the project as Pipeline and click on OK



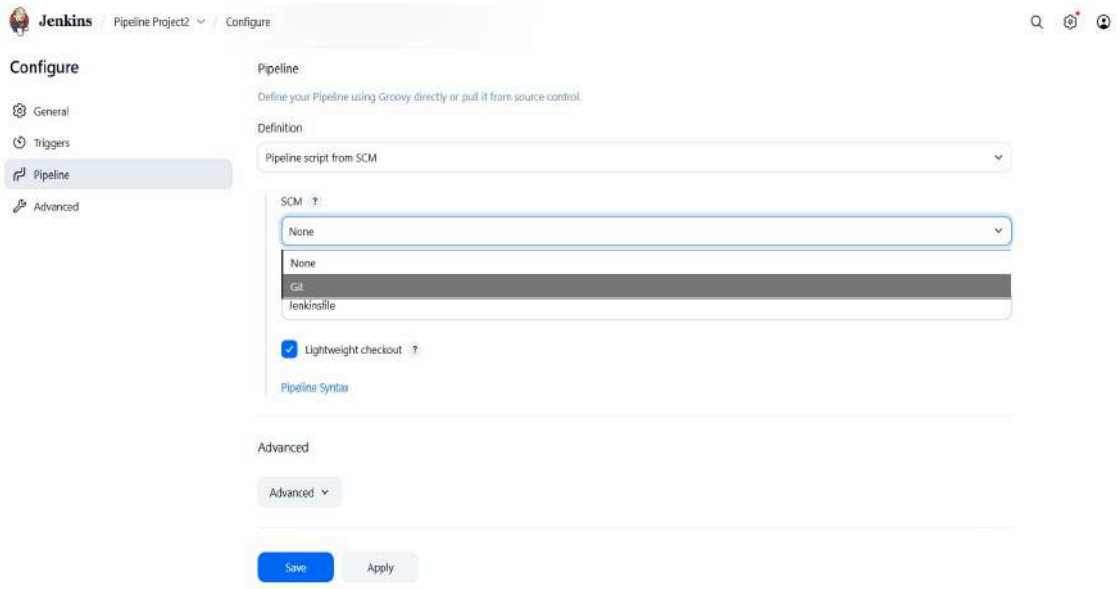
Step3: Just give a simple description for your project



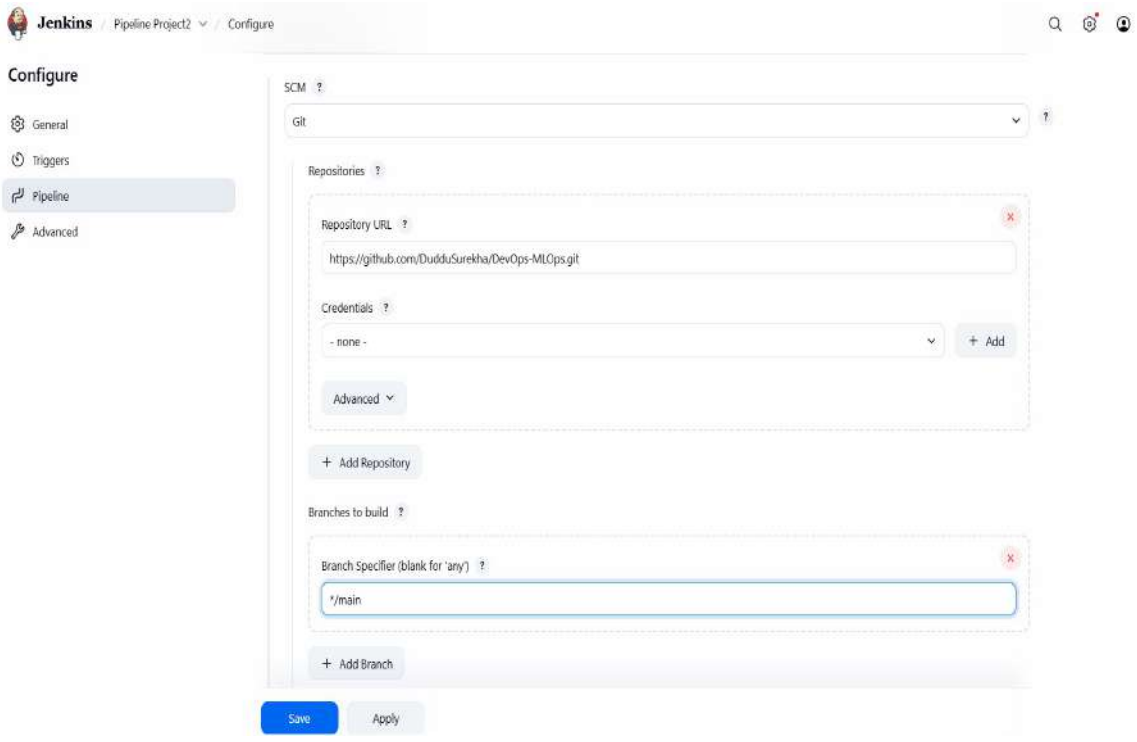
Step4: Enable the GitHub hook trigger for GITScm polling and select the Pipeline script from SCM



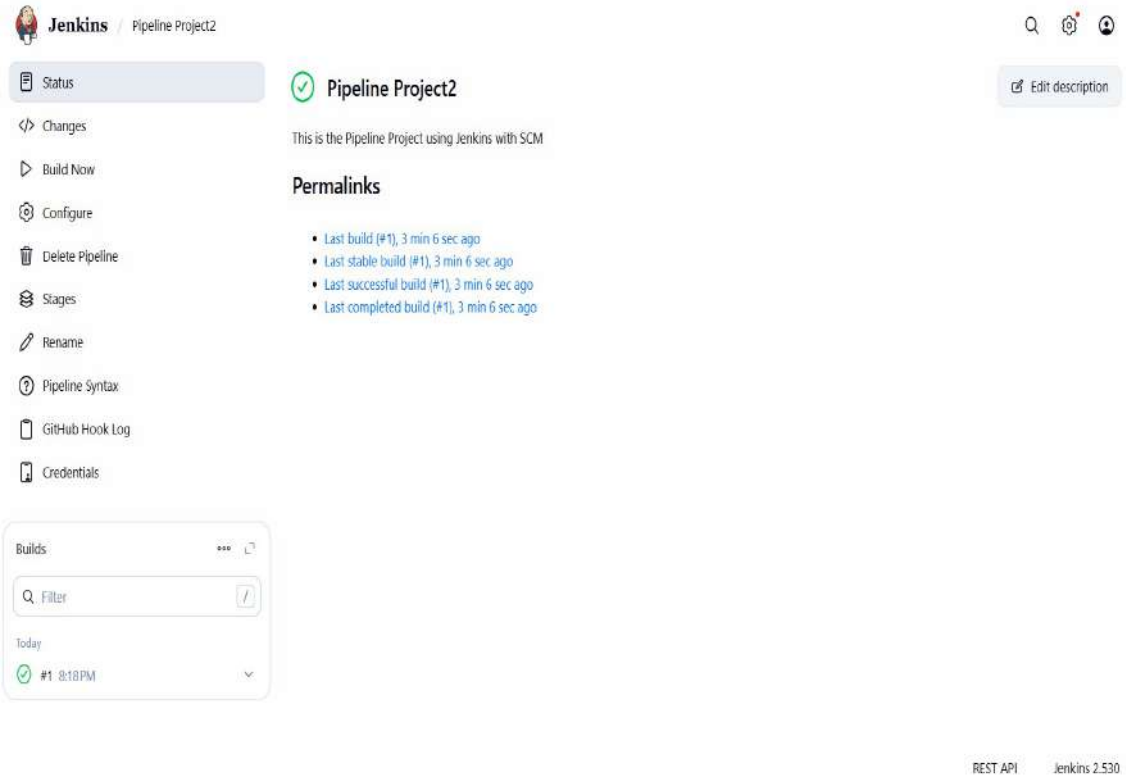
Step5: Then after selecting Pipeline script with SCM, in SCM we have to select the Git



Step6: Then give the GitHub repository URL and specify your branch whether it is master or main and click on Apply then Save

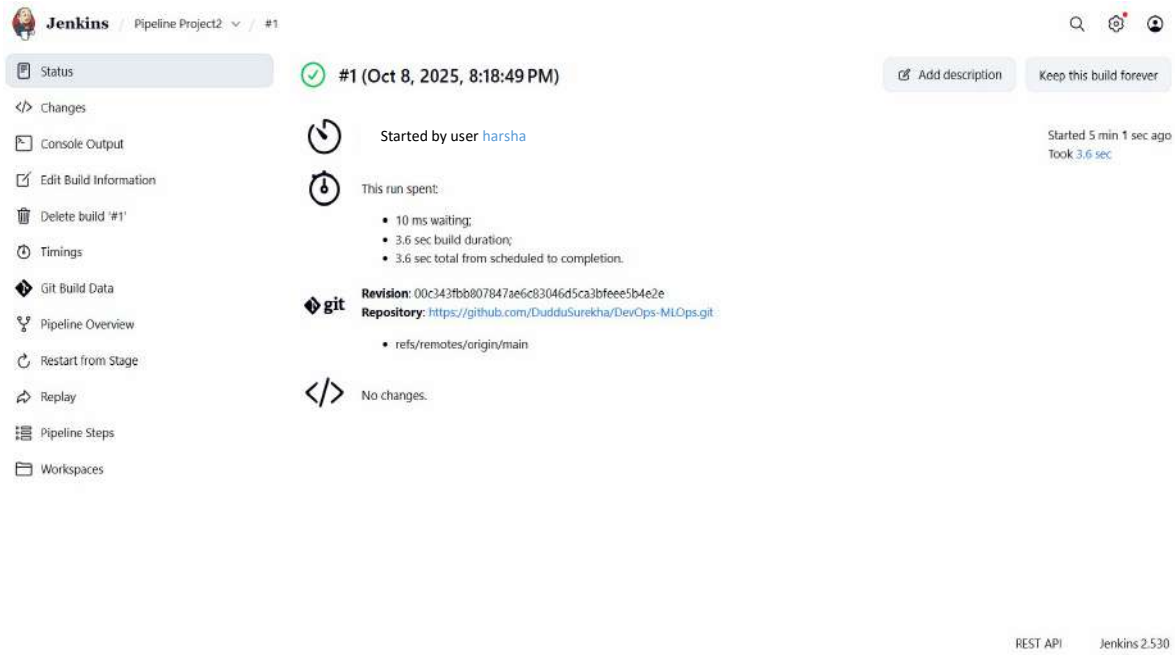


Step7: Now, we have to build our project by clicking on the Build Now

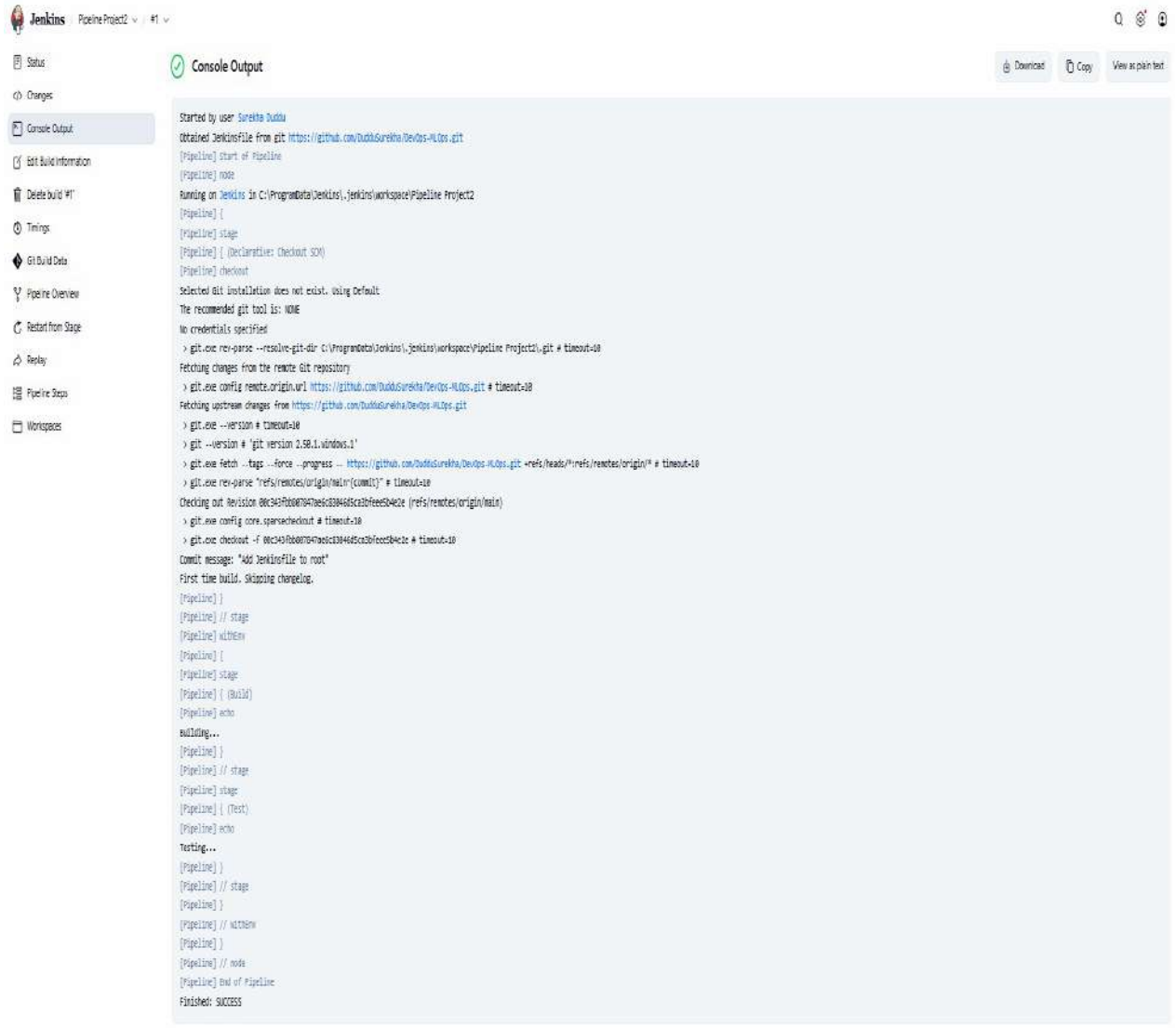


Step8: Then check the status and console output by clicking on build #1 to verify the build status whether the build is failed or succeed

#Status



Console

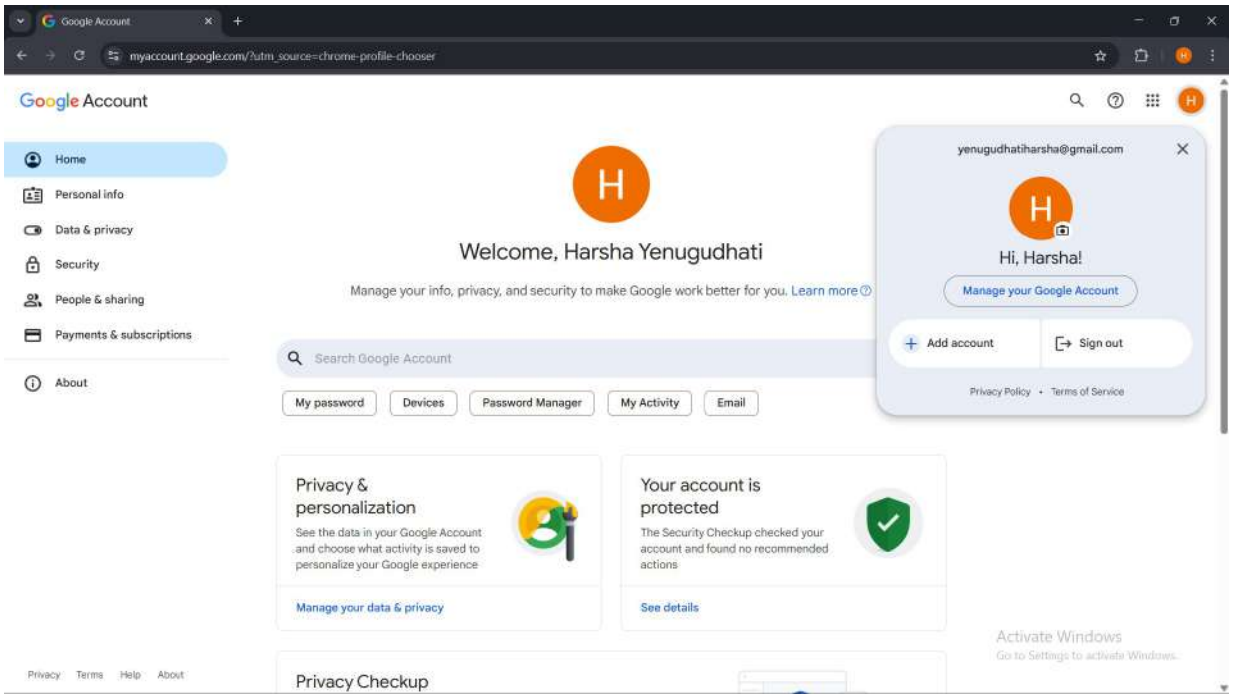


Output

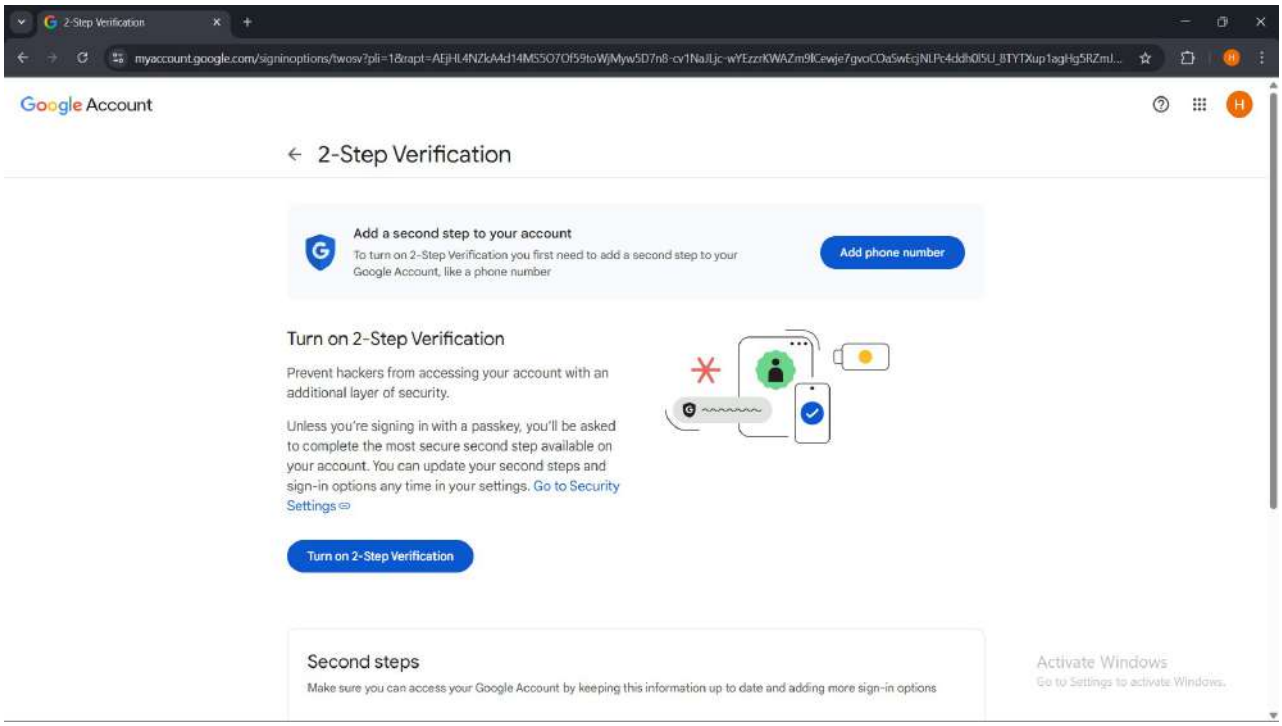
Experiment:8 Jenkins Email Notification

Aim: To configure email notifications in Jenkins.

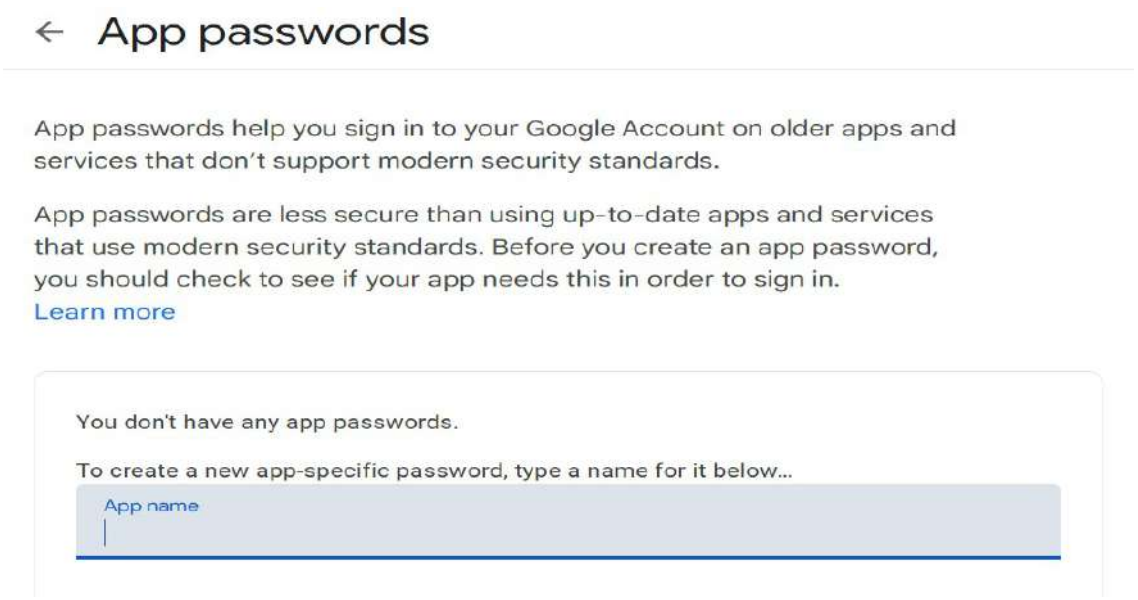
Step1: Login your mail and go to manage your google account



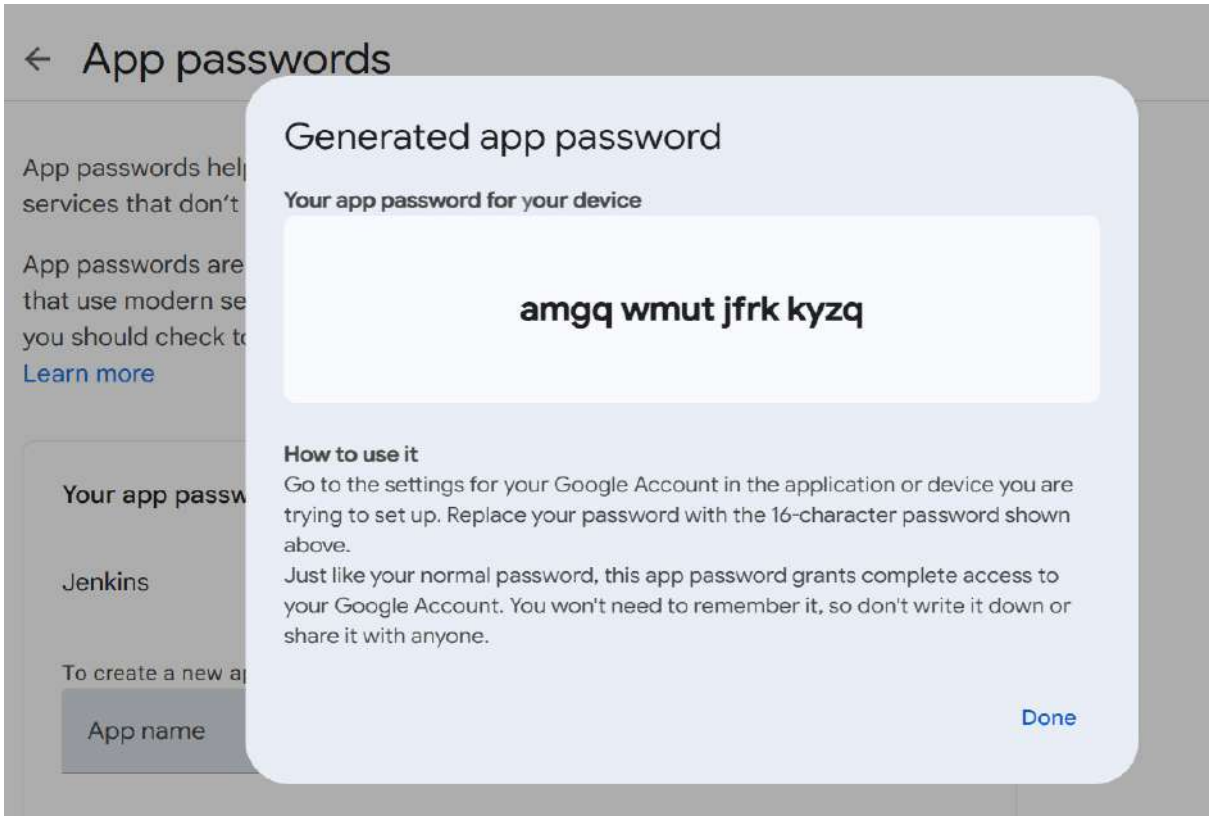
Step2: Then click on security and click on 2 step verification and turn on 2 step verification



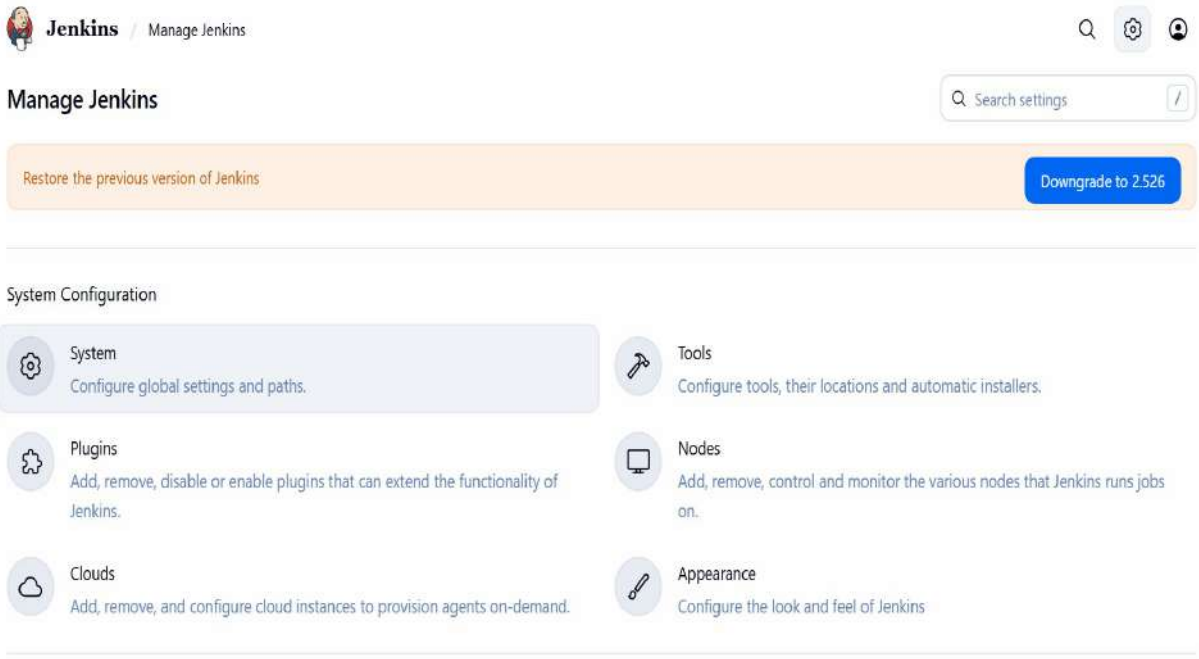
Step3: And again click on security and search for the app passwords



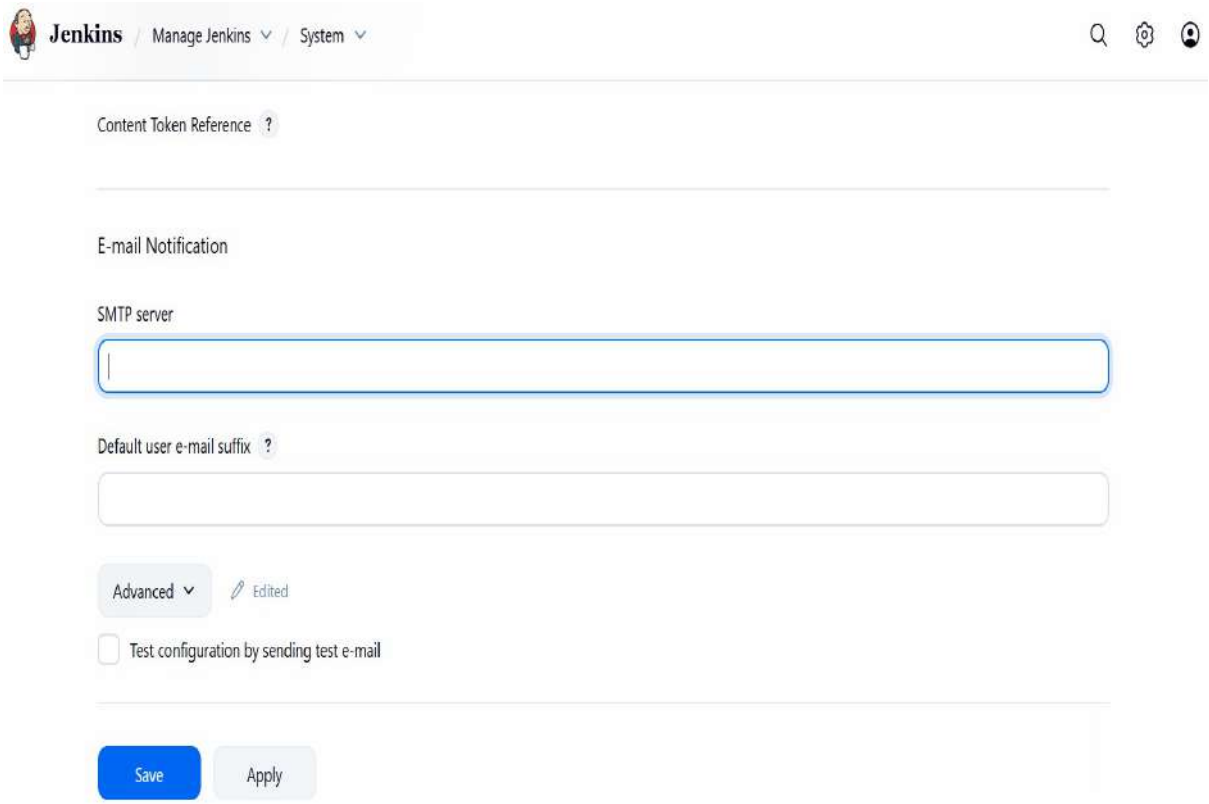
Step4: Create the app name like Jenkins and click on create and it generates the app password and this should be saved to our notepad



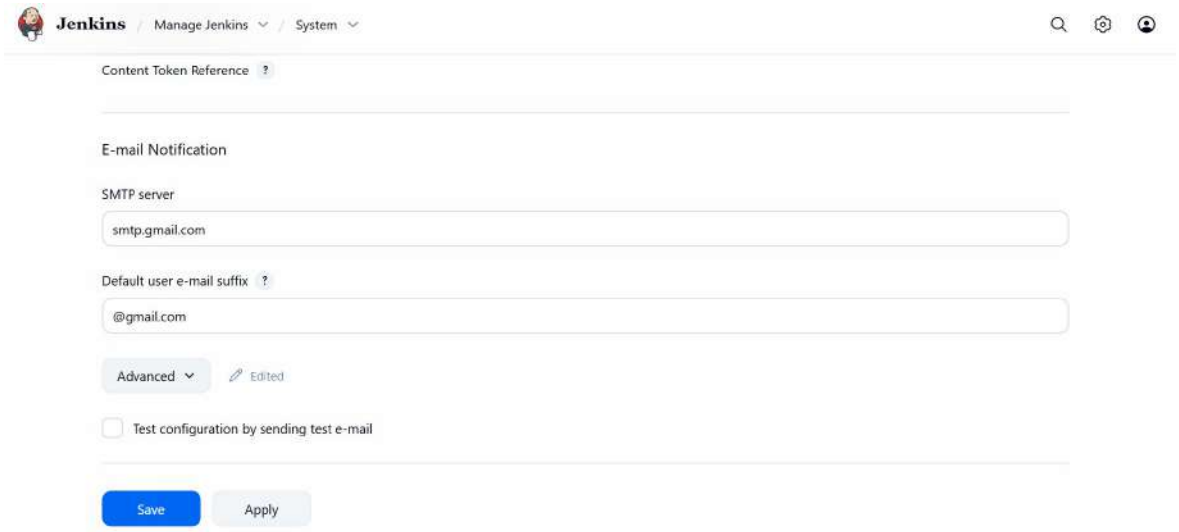
Step5: Sign in to Jenkins and go to manage Jenkins and click on system



Step6: After clicking on System then scroll down to E-mail Notification



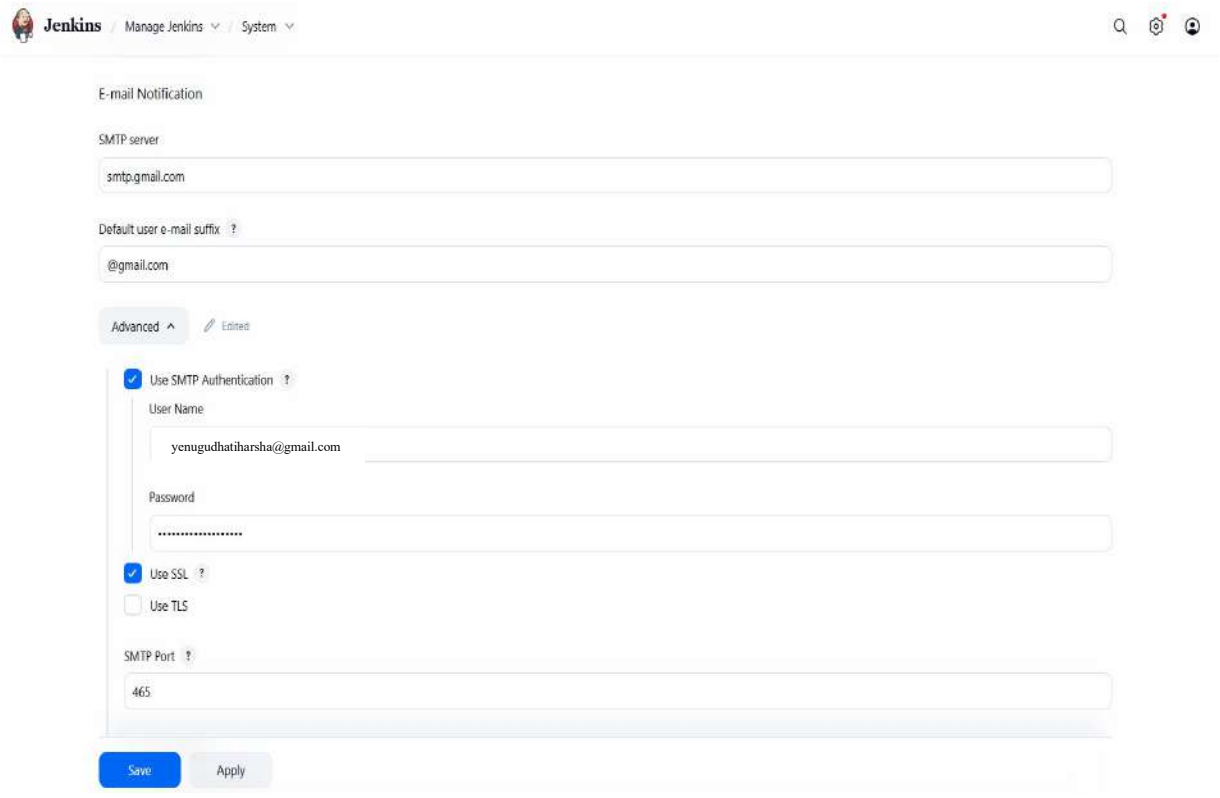
Step7: Fill the gmail stmp server like stmp.gmail.com and also set the default user email suffixes as @gmail.com



Step8: Click the Advanced Settings and apply these steps,

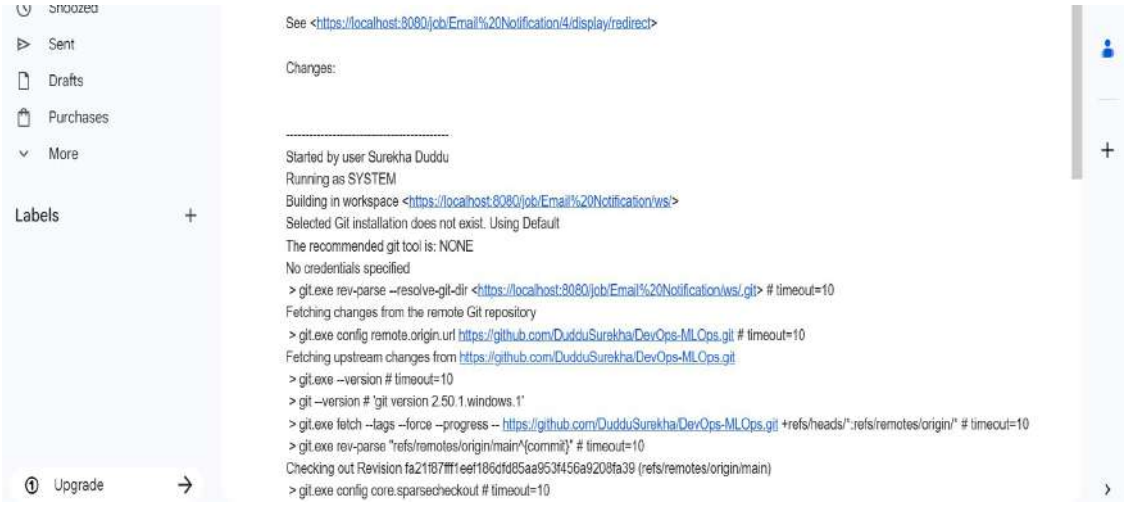
Enable SMTP Authentication and give your Username means Mail and Password means you have to give your 16-character generated App Passwords

SMTP Port 465

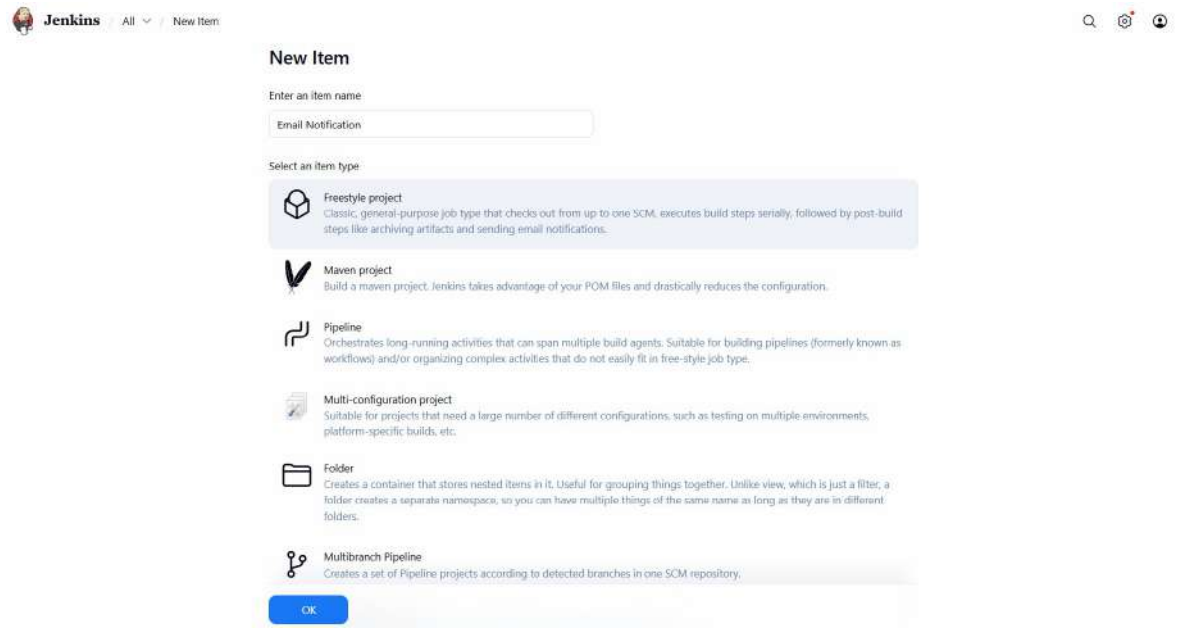


Step9: Enable Test configuration by sending test email and enter alternate/test email id to check whether email is sending to that user or not and click on Test configuration and click on apply and then save

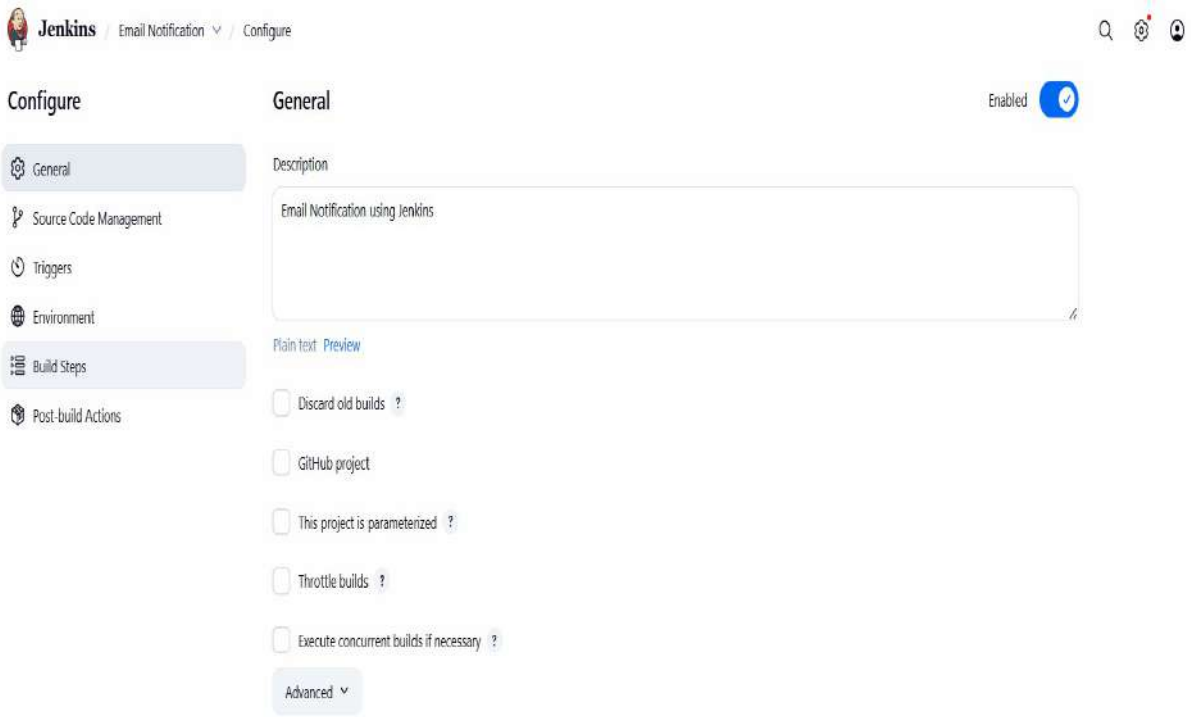
Step10: When the Email was successfully sent, then we get the test email for our mail then it specifies that our build is failed



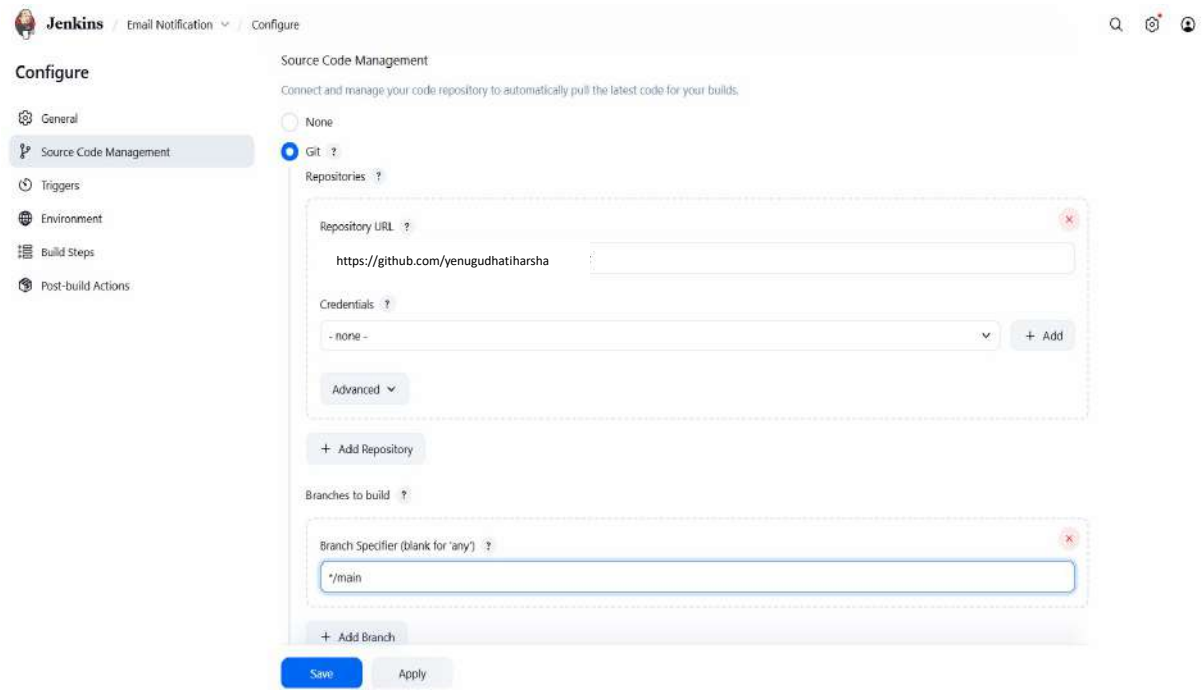
Step11: Next we have to create a project for this Email Notification so we have to click the + New Item in the Home page of Jenkins and name the project as Email Notification and select the project type as Freestyle and click on Ok



Step12: Give the simple description for the Email Notification Project

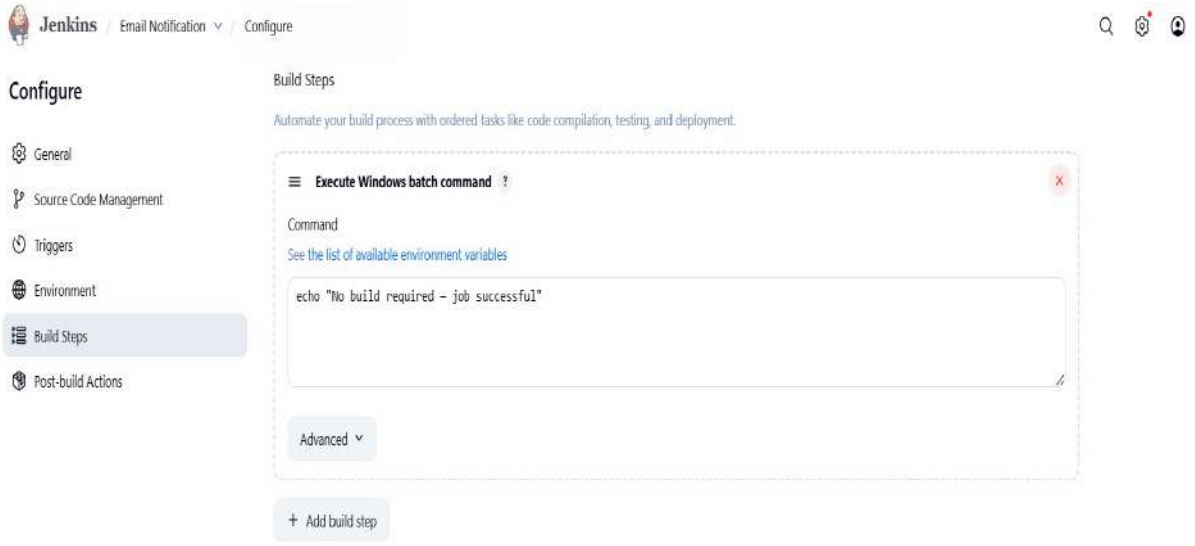


Step13: And select the git in the Source Code Management(SCM), and give your GitHub repository URL and give your branch specifier, by default it will be in master, you can change as per your branch.

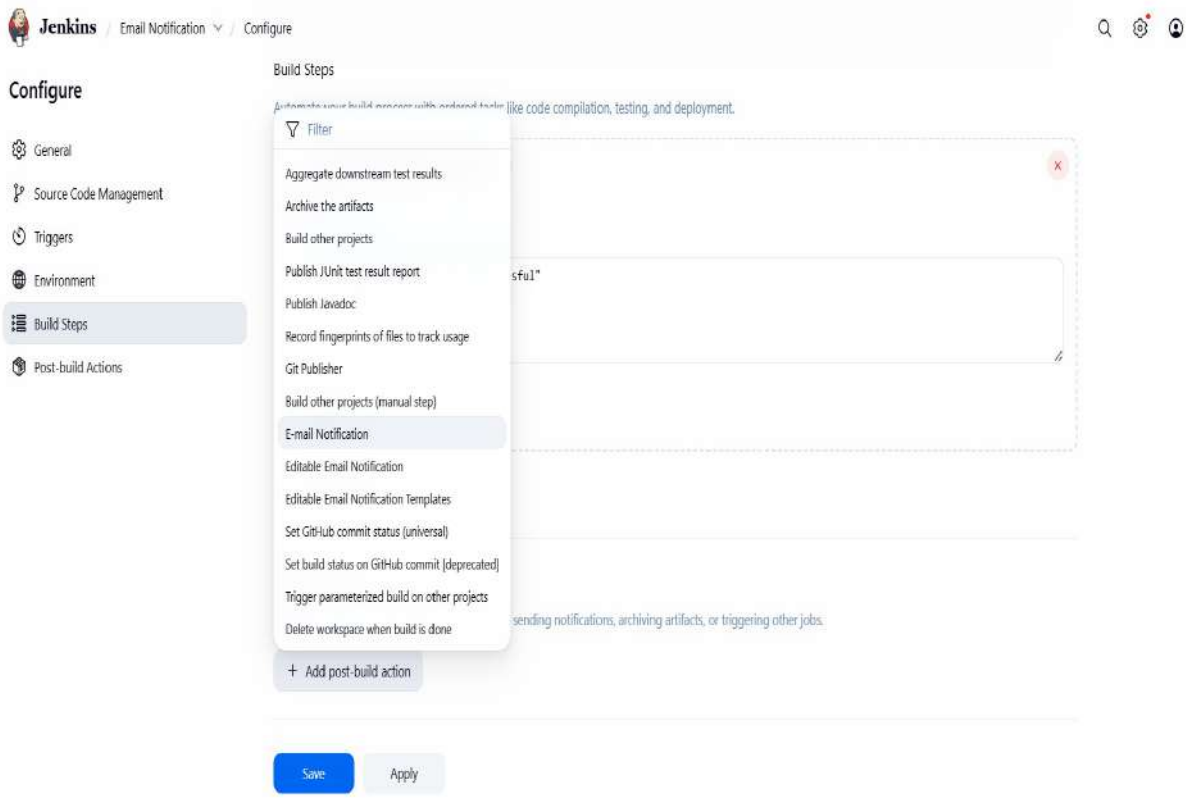


Step14: Enable the GitHub hook trigger for GITScm polling

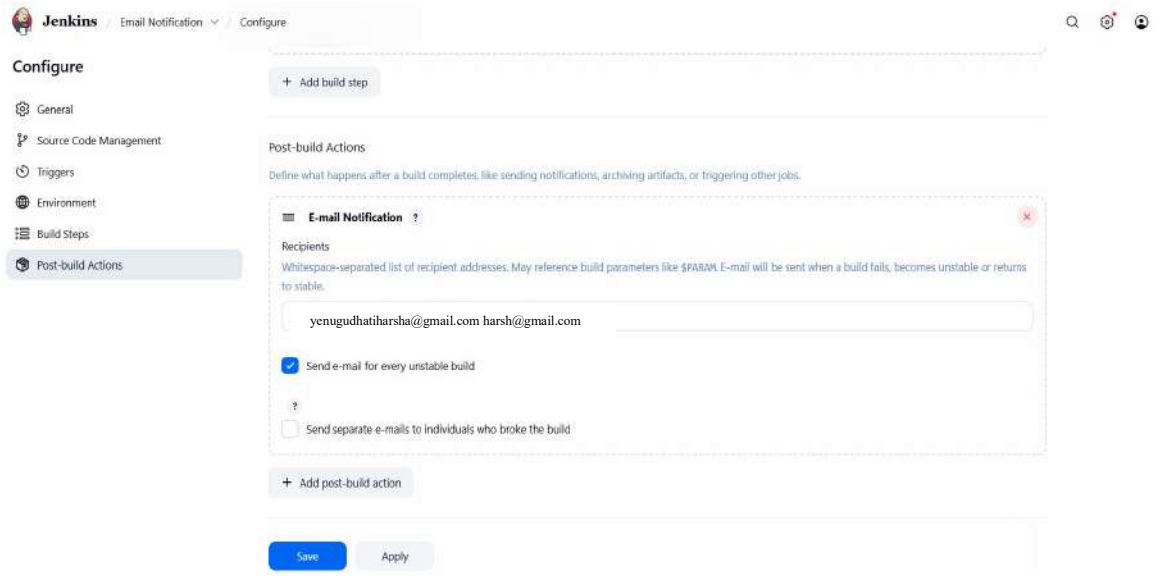
Step15: In Build Steps, you can select the windows batch command or any other from the dropdown and give your command to test whether the command is executing or not



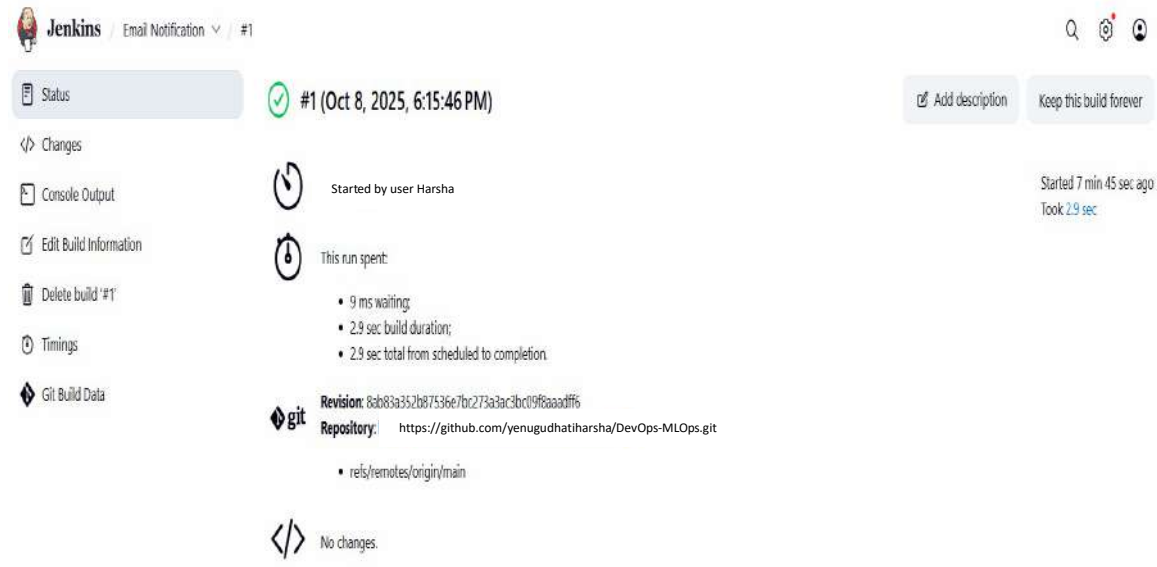
Step16: We have to add the Post build action from the dropdown, we have to select the Email Notification



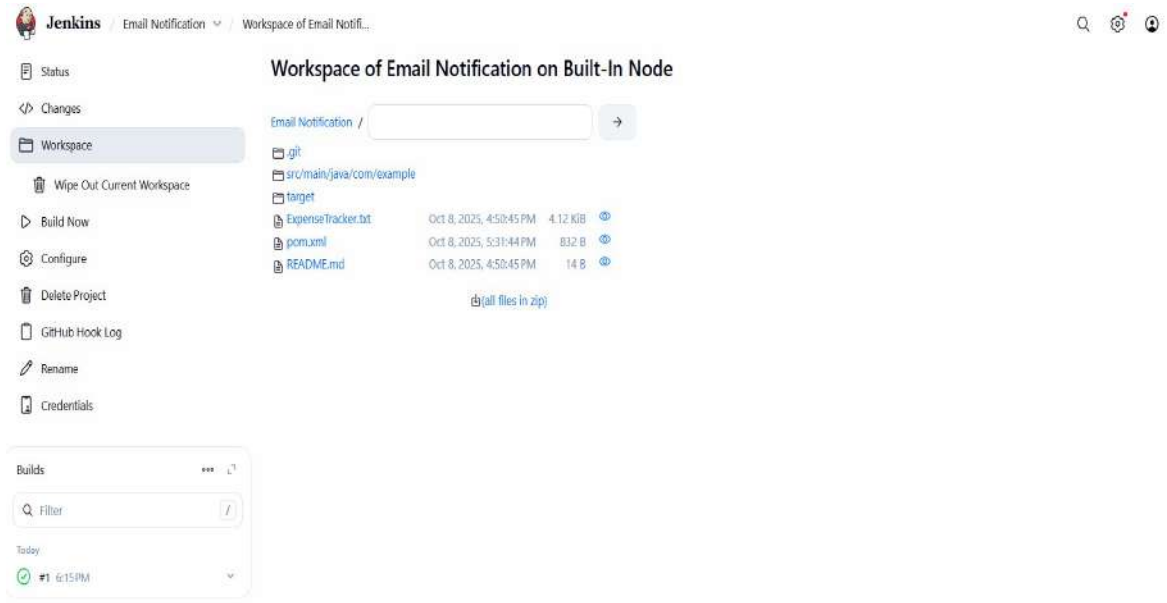
Step17: After selecting the Email Notification, then we have to enter our email and also the test email by providing space between them for the verification of email is sent to the test email or not and click on Apply then Save



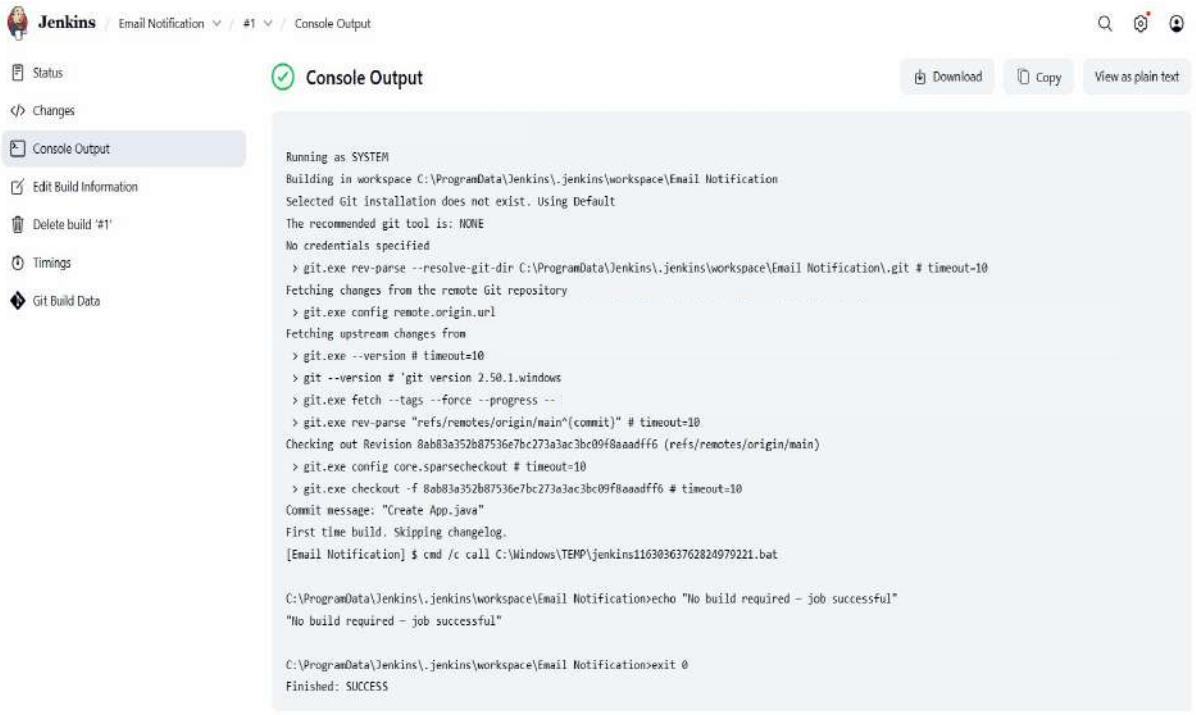
Step18: Now, we have to build our project, by clicking on the Build Now then only our project is build and we can view the status of our build by clicking on the build #1



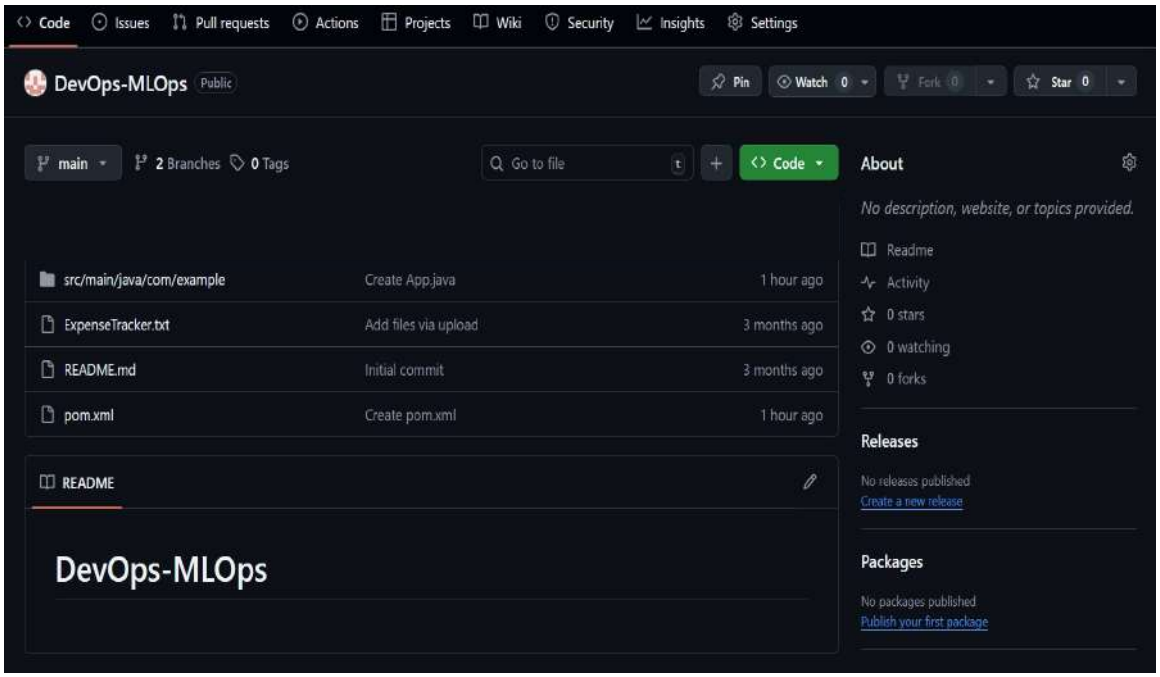
Step19: We can view our workspace of our Email Notification by clicking on Workspace



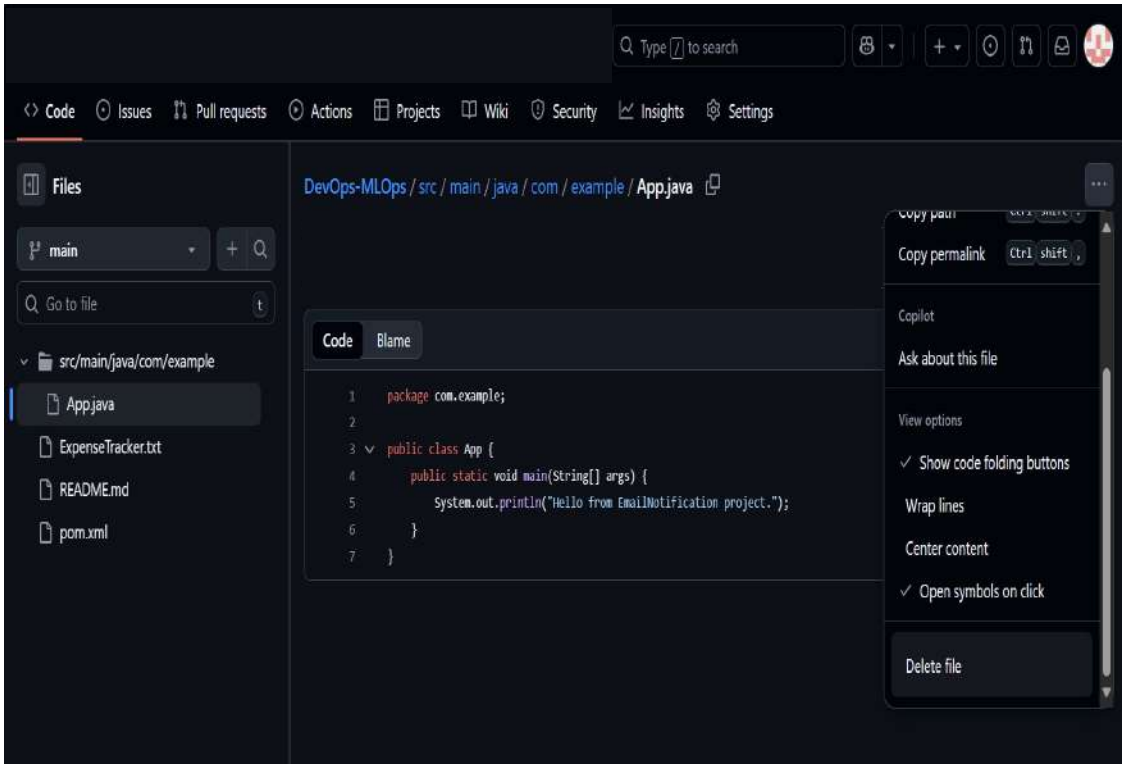
Step20: And we can view the console output whether the build is failed or success by clicking on the build #1 which was created



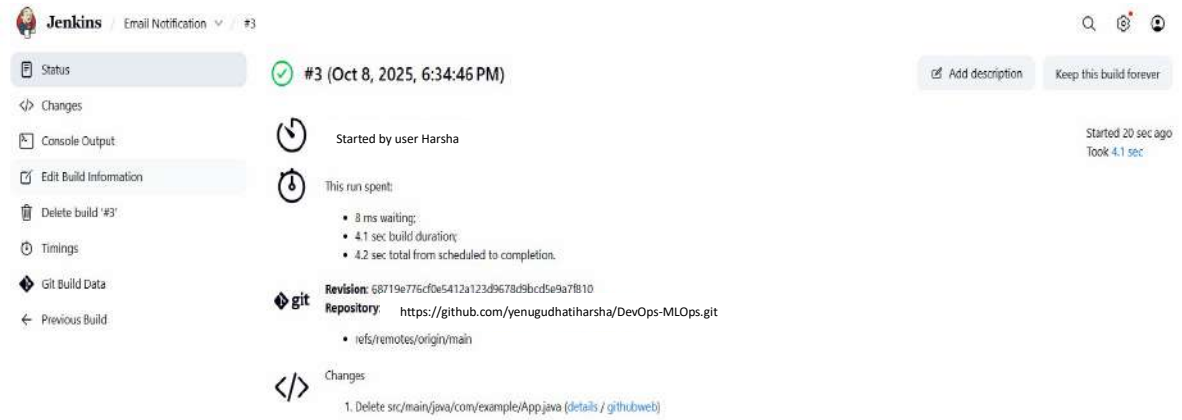
Step21: And my GitHub account contains following files in the DevOps-MLOps repository



Step22: Now I am making changes in the repository files, whether it may be deleting the files or adding the files or any code changes in the repository.



Step22: After deleting the file App.js in the DevOps-MLOps repository then again go to Jenkins and again build the project Email Notification, then observe the status of changes it displays the changes you made in the repository.

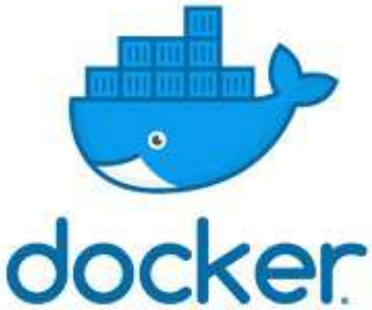


Step23: And view the entire output by clicking the Console Output, and verify whether the build is failed or succeed

Experiment-9: Working with Docker

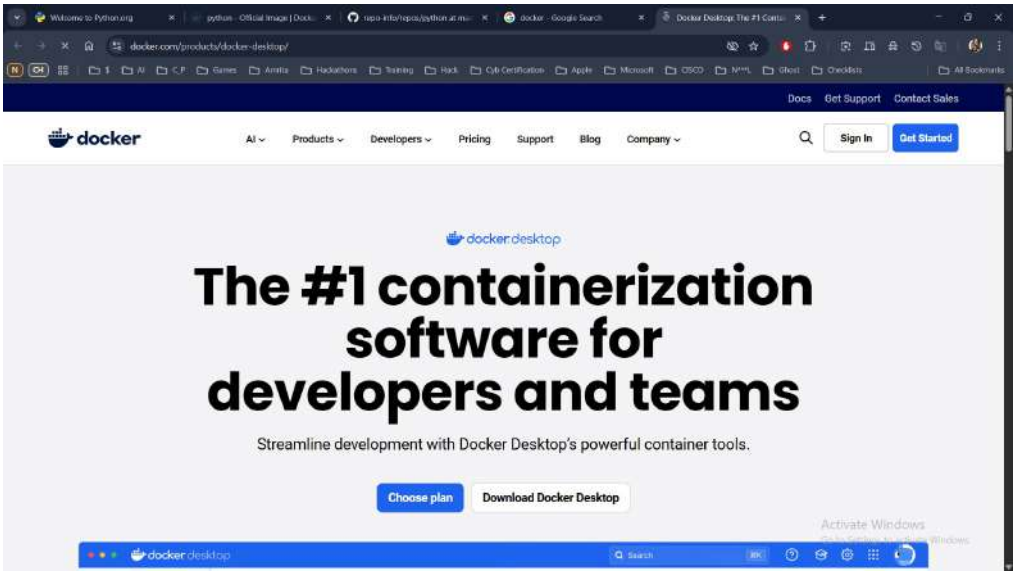
Aim: To demonstrate containerization using Docker.

- Docker is an open-source platform for developing, shipping, and running applications.
- It uses containerization technology to package applications with their dependencies.
- Ensures applications run **consistently across different environments** (development, testing, production).
- Provides **lightweight, portable, and scalable** solutions compared to traditional virtual machines.
- Widely used in **DevOps, cloud computing, and microservices** architecture.



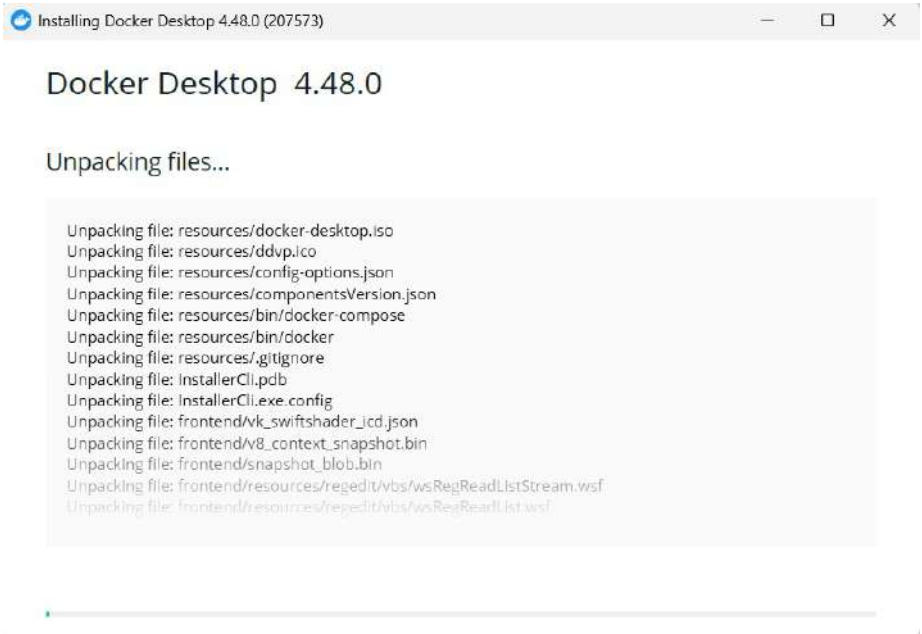
Steps to install Docker Desktop

1. **Check System Requirements**
    1. Windows: Windows 10/11 Pro, Enterprise, or Education (with WSL2 enabled)
    2. macOS: macOS 10.15+
    3. Linux: Ubuntu/Debian/Fedora/CentOS
2. **Download Docker Desktop**
    1. Visit https://www.docker.com/products/docker-desktop



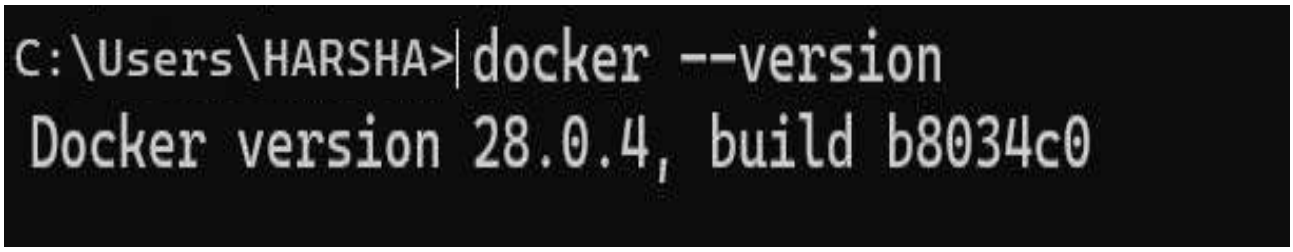
3. Install Docker

- Run the installer and follow setup instructions.
- Accept terms and complete installation.



4. Verify Installation

1. Open terminal/command prompt.
2. Run docker –version

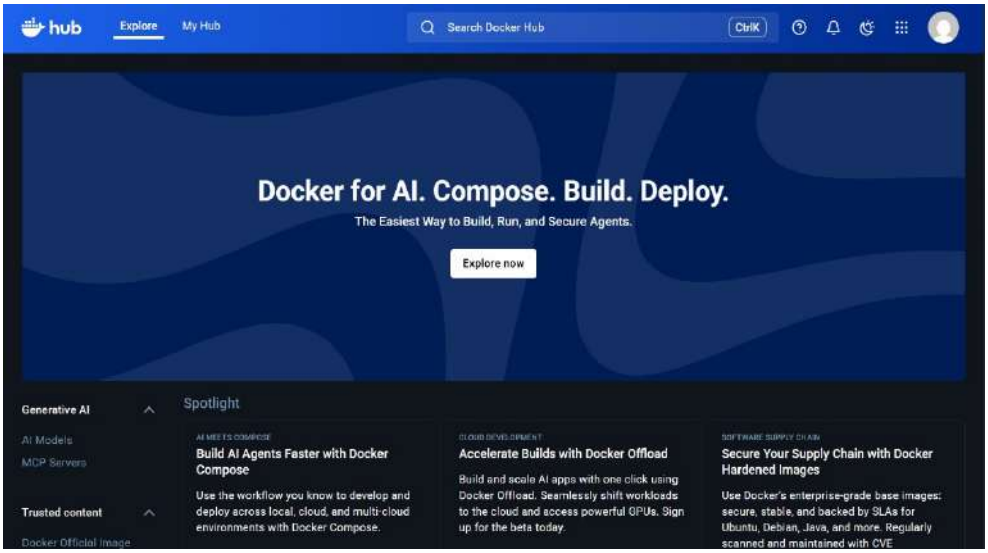


Docker Hub

Docker Hub is Docker's official cloud-based registry service.

- It allows developers to:
  - **Store** Docker images.
  - **Share** images publicly or privately.
  - **Download** (pull) ready-to-use images for apps, databases, and tools.
- Acts like a Github for Docker images

- Provides both:
  - **Official images** (verified by Docker, e.g., MySQL, Nginx, Python).
  - **Community images** (shared by developers).
- Create a Docker Hub Account



Docker Image

A **Docker Image** is like a **blueprint** – it is a read-only template that contains the application code, dependencies, libraries, and configuration needed to run an application. You cannot run an image directly.

Docker Container

when you **start an image**, it becomes a **Docker Container**. A **Docker Container** is a running instance of an image – it is a lightweight, isolated environment where the application executes. You can create multiple containers from the same image, just like you can build multiple houses from the same blueprint.

Creating a Docker image

There are Multiple ways to create the Docker images:

1. **Using Dockerfile**
   - Write a text file (Dockerfile) with instructions.
   - Define base image, dependencies, environment, and commands.

Dockerfile

```
FROM openjdk:11
VOLUME /tmp
ADD target/my-app.jar my-app.jar
EXPOSE 8080
ENTRYPOINT ["java","-jar","/my-app.jar"]
```

- Steps to build the docker image from docker file and push to docker hub:

1. **Login to Docker Hub**

   docker login

2. **Build the Docker Image from Dockerfile**

   docker build -t myapp:1.0 .

3. **Tag the Image for Docker Hub**

   docker tag myapp:1.0 username/myapp:1.0

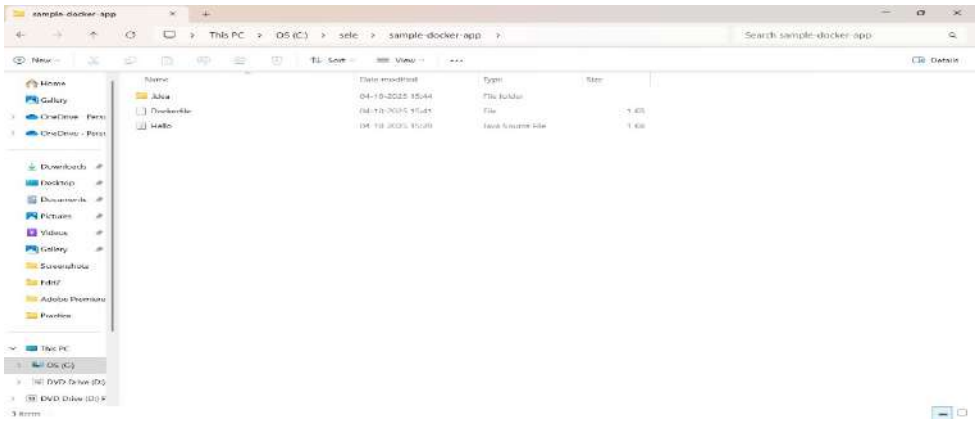
4. **Push the Image to Docker Hub**

   docker push username/myapp:1.0

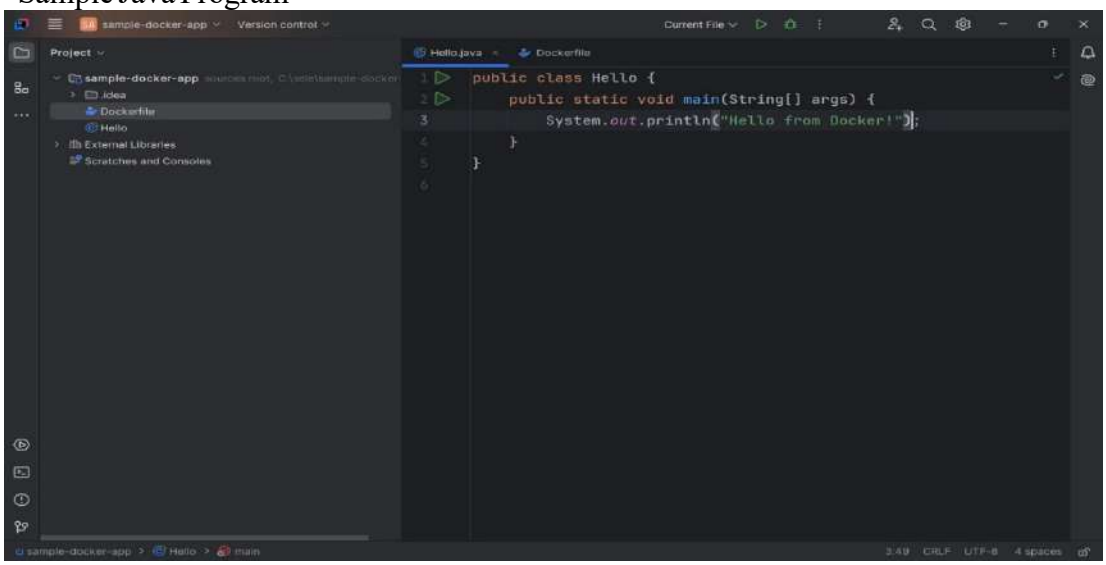
5. **Build the Docker Image from Dockerfile**

   docker build -t myapp:1.0 .

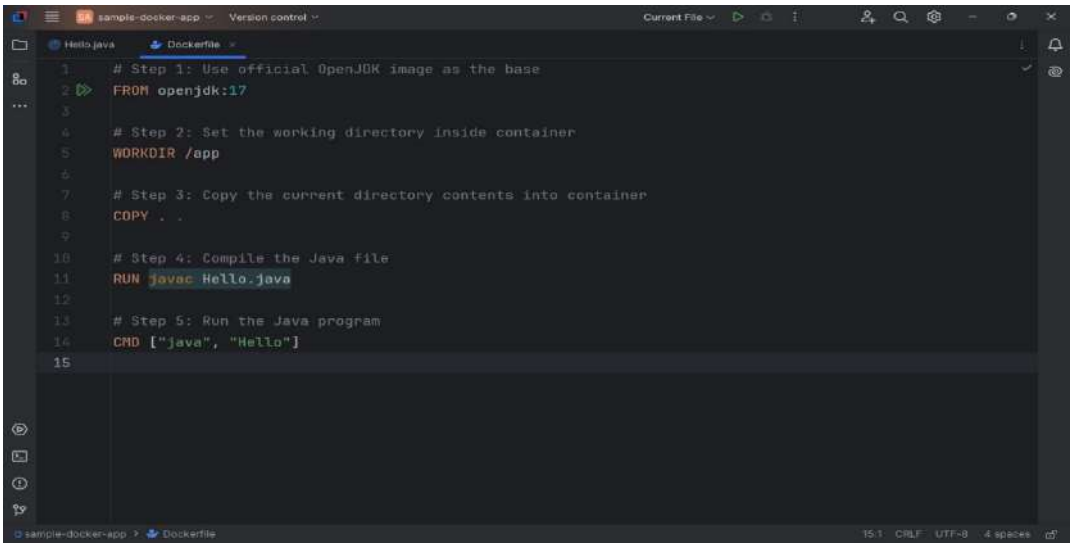
Make a folder and write a sample program and docker file in it.



- Sample Java Program



- Sample Docker File



- Build the docker image

```
C:\Users\HARSHA>le-docker-app> docker build -t javaapp:1.0 .
[+] Building 68.9s (10/10) FINISHED                                                    docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                0.0s
 => => transferring dockerfile: 364B                                                                0.0s
 => [internal] load metadata for docker.io/library/openjdk:17                                       6.2s
 => [auth] library/openjdk:pull token for registry-1.docker.io                                      0.0s
 => [internal] load .dockerignore                                                                   0.0s
 => => transferring context: 2B                                                                     0.0s
 => [1/4] FROM docker.io/library/openjdk:17@sha256:528707081fdb9562eb819128a9f85ae7fe000e2fbaeaf9f87662e7b3f38cb7d8  60.9s
 => => resolve docker.io/library/openjdk:17@sha256:528707081fdb9562eb819128a9f85ae7fe000e2fbaeaf9f87662e7b3f38cb7d8  0.0s
 => => sha256:528707081fdb9562eb819128a9f85ae7fe000e2fbaeaf9f87662e7b3f38cb7d8 1.04kB / 1.04kB       0.0s
 => => sha256:98f0304b3a3b7c12ce641177a99d1f3be56f532473a528fda38d53d519cafb13 954B / 954B           0.0s
 => => sha256:5e28ba2b4cdb3a7c3bd0ee2e635a5f6481682b77eabf8b51a17ea8bfe1c05697 4.45kB / 4.45kB       0.0s
 => => sha256:38a980f2cc8accf69c23deae6743d42a87eb34a54f02396f3fcfd7c2d06e2c5b 42.11MB / 42.11MB    36.9s
 => => sha256:de849f1cfbe60b1c06a1db83a3129ab0ea397c4852b98e3e4300b12ee57ba111 13.53MB / 13.53MB    45.1s
 => => sha256:a7203ca35e75e068651c9907d659adc721dba823441b78639fde66fc988f042f 187.53MB / 187.53MB  58.6s
 => => extracting sha256:38a980f2cc8accf69c23deae6743d42a87eb34a54f02396f3fcfd7c2d06e2c5b           4.5s
 => => extracting sha256:de849f1cfbe60b1c06a1db83a3129ab0ea397c4852b98e3e4300b12ee57ba111           0.5s
 => => extracting sha256:a7203ca35e75e068651c9907d659adc721dba823441b78639fde66fc988f042f           2.1s
 => [internal] load build context                                                                   0.0s
 => => transferring context: 3.72kB                                                                 0.0s
 => [2/4] WORKDIR /app                                                                              0.3s
 => [3/4] COPY . .                                                                                  0.0s
 => [4/4] RUN javac Hello.java                                                                      0.7s
 => exporting to image                                                                              0.1s
 => => exporting layers                                                                             0.0s
 => => writing image sha256:6cb81cd298378c7c8d38175e03da7a707563670f5ee9fa81710dcd543ec6b7f7        0.0s
 => => naming to docker.io/library/javaapp:1.0                                                      0.0s
```

- Run it locally by command
  - docker run javaapp:1.0

```
PS C:\Users\HARSHA>-app> docker run javaapp:1.0
Hello from Docker!
C:\Users\HARSHA>locker-app>
```

Tag the Image for Docker Hub

docker tag myapp:1.0 username/myapp:1.0

```
PS C:\sele\sample-docker-app> docker tag javaapp:1.0 yuvanagavenkat/javaapp:1.0
PS C:\sele\sample-docker-app>
```

- Checking the docker images

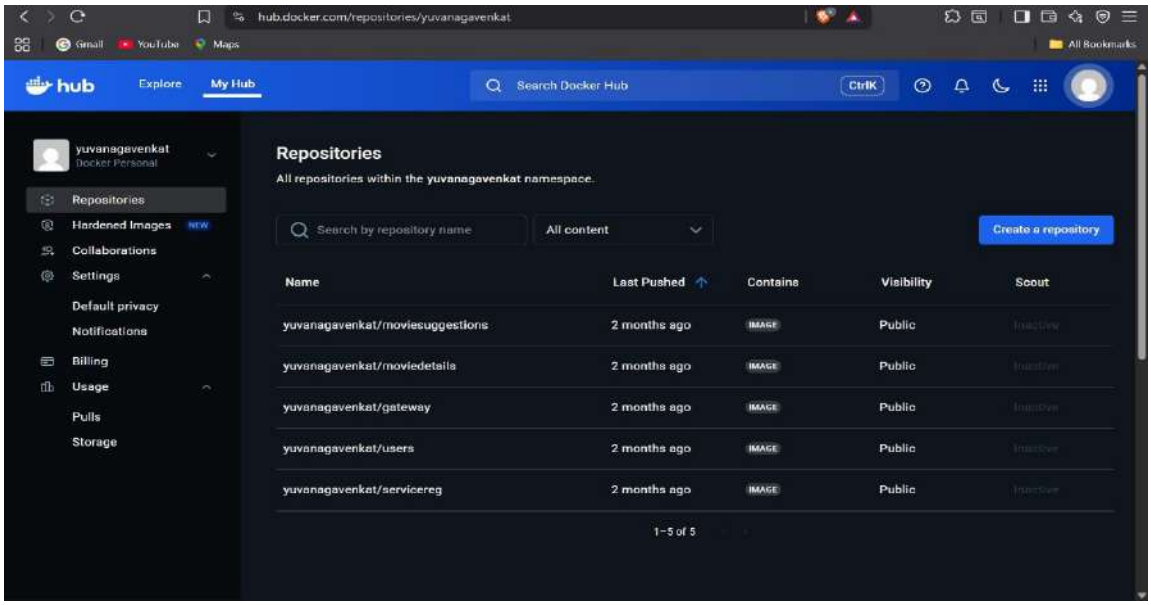
```
PS C:\sele\sample-docker-app> docker images
REPOSITORY                      TAG       IMAGE ID       CREATED          SIZE
javaapp                         1.0       6cb81cd29837   43 minutes ago   471MB
yuvanagavenkat/javaapp          1.0       6cb81cd29837   43 minutes ago   471MB
```

Push the Image to Docker Hub

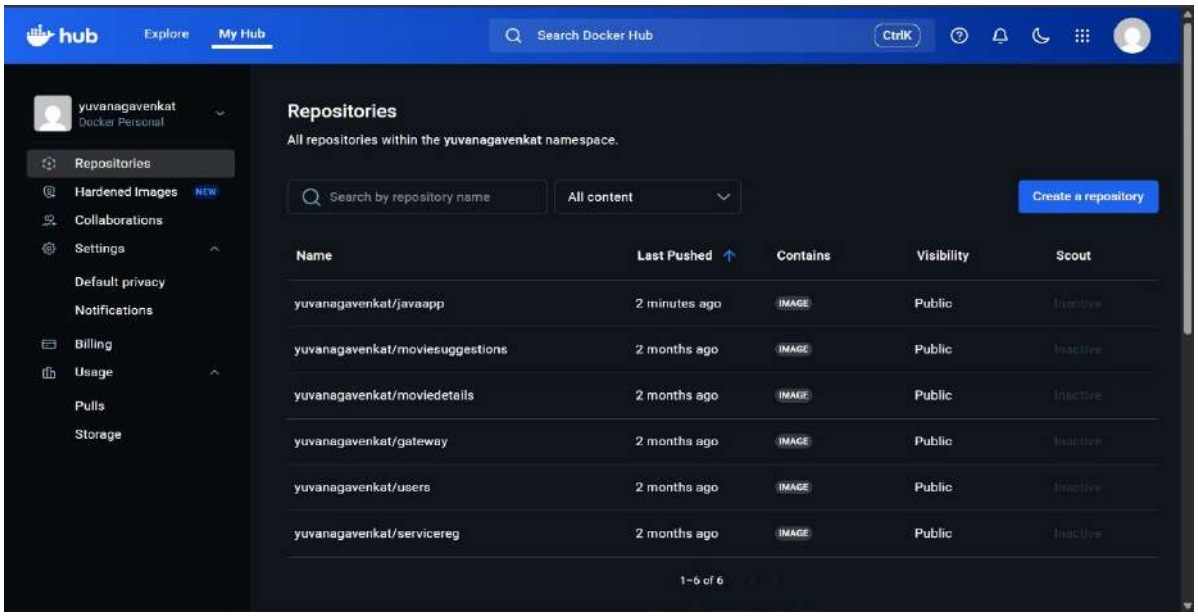
docker push username/myapp:1.0

```
PS C:\sele\sample-docker-app> docker push yuvanagavenkat/javaapp:1.0
The push refers to repository [docker.io/yuvanagavenkat/javaapp]
6092df110b67: Pushed
080d632ffa30: Pushed
b19ac98c4147: Pushed
dc9fa3d8b576: Mounted from library/openjdk
27ee19dc88f2: Mounted from library/openjdk
c8dd97366670: Mounted from library/openjdk
1.0: digest: sha256:1974a2f8ca4f7575f5577335804a6d3dcc5c0512221d8e4b33a8c859b7df5ec9 size: 1575
```

- Docker Hub Previously :



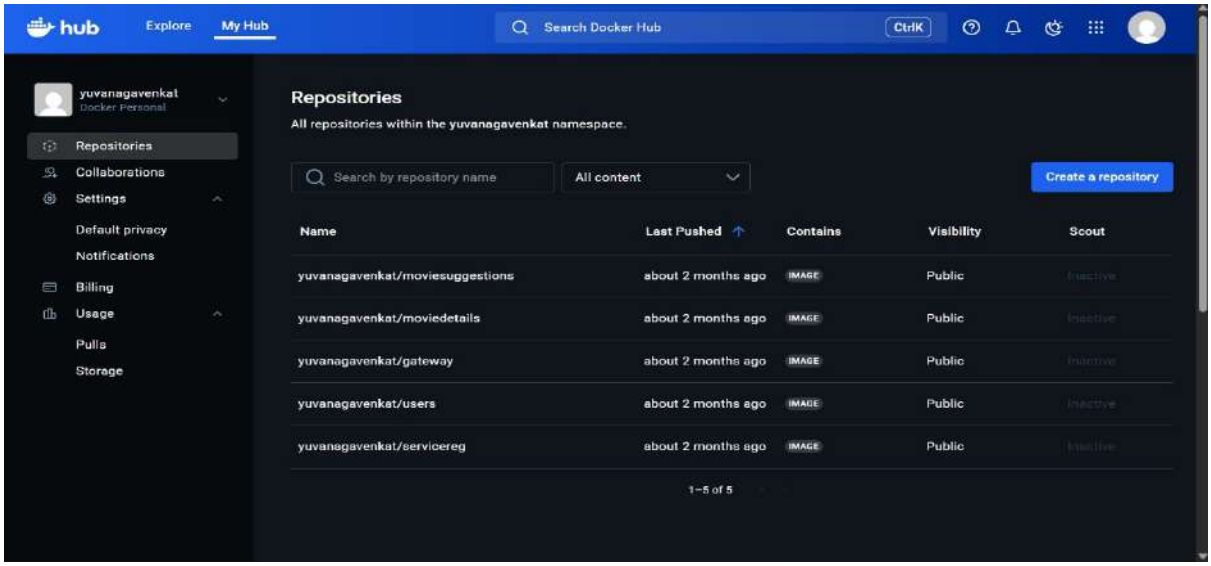
- Docker Hub After:



2. Using Maven Wrapper (for Java apps)

- Add Docker plugin in pom.xml.
- Build Docker image as part of Maven build process.
- Run the example command
- ./mvnw spring-boot:build-image "-Dspring-boot.build-image.imageName=yuvanagavenkat/users"
- Here by using the above command docker image is directly build
- And also pushed into docker hub

Sample Docker Images in Docker Hub



Pulling the Docker Image

- Docker images are like blueprints – they need to exist on our local machine to run a container.
- If the image does not exist locally, we must pull it from docker hub.
- Example command to pull the docker image from the docker hub

```
PS C:\Movies_Project\Backend> docker pull yuvanagavenkat/moviesuggestions
Using default tag: latest
latest: Pulling from yuvanagavenkat/moviesuggestions
Digest: sha256:66624eb72415238e7e329ddf034722b18e4cb2c86e04f46821876bab8d5fc10d
Status: Image is up to date for yuvanagavenkat/moviesuggestions:latest
docker.io/yuvanagavenkat/moviesuggestions:latest
```

Checking the Docker images

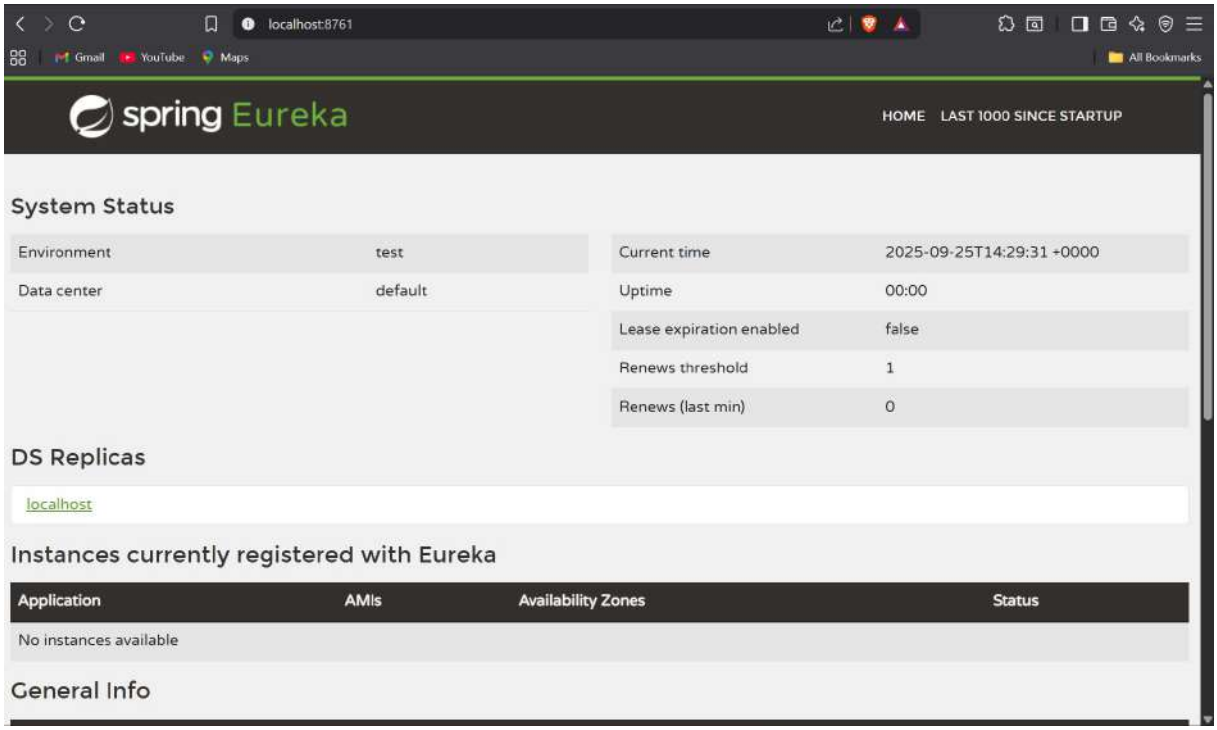
We Can Check the docker images by the command:

```
PS C:\Movies_Project\Backend> docker images
REPOSITORY                                TAG           IMAGE ID       CREATED         SIZE
jenkins/jenkins                           lts           091ba5742e7e   5 weeks ago     472MB
paketobuildpacks/ubuntu-noble-run-tiny    0.0.21        db790807286f   8 weeks ago     22MB
mongo                                     latest        3628b824922a   2 months ago    908MB
dpage/pgadmin4                            latest        2c990ea76ddb   2 months ago    531MB
mysql                                     8.0           11b713560ad5   2 months ago    781MB
paketobuildpacks/ubuntu-noble-run-tiny    0.0.20        f0aa20be8278   2 months ago    22MB
postgres                                  latest        8663c6099632   3 months ago    438MB
openzipkin/zipkin                         latest        db081968d3db   5 months ago    227MB
rabbitmq                                  3-management  17752bd0f348   12 months ago   252MB
yuvanagavenkat/moviesuggestions           latest        3fd2002dac4c   45 years ago    307MB
paketobuildpacks/builder-noble-java-tiny  latest        617393b759e3   45 years ago    803MB
yuvanagavenkat/moviedetails               latest        2d009ec38e24   45 years ago    307MB
yuvanagavenkat/users                      latest        fc789f6d9721   45 years ago    324MB
yuvanagavenkat/moviesuggestions           <none>        515042dcae68   45 years ago    307MB
```

Running the image as Container

We Can Run the image as container by using the command: docker run [OPTIONS] IMAGE [COMMAND] [ARG...]

```
PS C:\Movies_Project\Backend> docker run -p 8761:8761 yuvanagavenkat/servicereg
Calculating JVM memory based on 6329312K available memory
For more information on this calculation, see https://paketo.io/docs/reference/java-reference/#memory-calculator
Calculated JVM Memory Configuration: -XX:MaxDirectMemorySize=10M -Xmx5690245K -XX:MaxMetaspaceSize=127066K -XX:ReservedCodeCacheSize=
240M -Xss1M (Total Memory: 6329312K, Thread Count: 250, Loaded Class Count: 20020, Headroom: 0%)
Enabling Java Native Memory Tracking
Adding 146 container CA certificates to JVM truststore
Spring Cloud Bindings Enabled
Picked up JAVA_TOOL_OPTIONS: -Djava.security.properties=/layers/paketo-buildpacks_bellsoft-liberica/java-security-properties/java-sec
urity.properties -XX:+ExitOnOutOfMemoryError -XX:MaxDirectMemorySize=10M -Xmx5690245K -XX:MaxMetaspaceSize=127066K -XX:ReservedCodeCa
cheSize=240M -Xss1M -XX:+UnlockDiagnosticVMOptions -XX:NativeMemoryTracking=summary -XX:+PrintNMTStatistics -Dorg.springframework.clo
ud.bindings.boot.enable=true

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/

 :: Spring Boot ::                (v3.5.0)

2025-09-25T14:29:02.642Z  INFO 1 --- [eureka] [           main] com.movies.eureka.EurekaApplication      : Starting EurekaApplication
 v0.0.1-SNAPSHOT using Java 21.0.8 with PID 1 (/workspace/BOOT-INF/classes started by cnb in /workspace)
2025-09-25T14:29:02.645Z  INFO 1 --- [eureka] [           main] com.movies.eureka.EurekaApplication      : No active profile set, fal
ling back to 1 default profile: "default"
2025-09-25T14:29:03.853Z  INFO 1 --- [eureka] [           main] o.s.cloud.context.scope.GenericScope     : BeanFactory id=43258acc-55
67-3052-b0d3-7b847cfc4b18
2025-09-25T14:29:06.326Z  INFO 1 --- [eureka] [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with po
rt 8761 (http)
2025-09-25T14:29:06.343Z  INFO 1 --- [eureka] [           main] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2025-09-25T14:29:06.343Z  INFO 1 --- [eureka] [           main] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [
```



Pulling the Public images

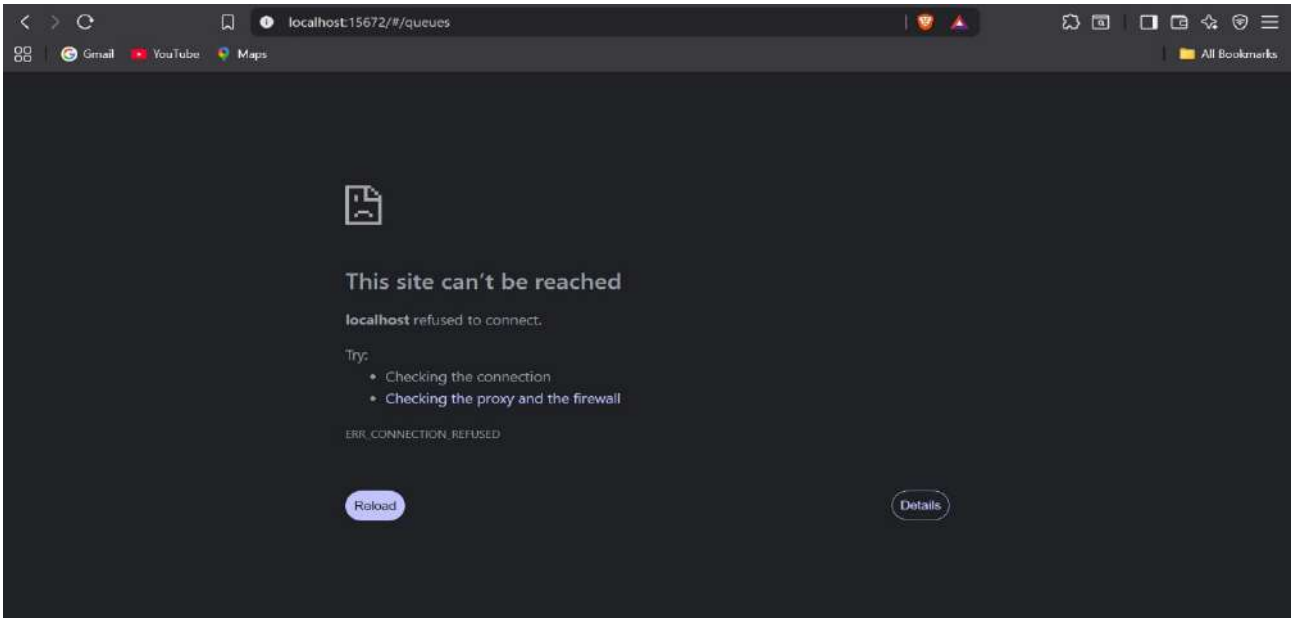
- We Can also pull the public images that are available in docker and we can use it with out installing the application.

```
PS C:\sele\sample-docker-app> docker pull rabbitmq:3-management
3-management: Pulling from library/rabbitmq
a1a21c96bc16: Pull complete
fd14d01c5c8e: Pull complete
c62580be56ce: Pull complete
8b5ba30aa19d: Pull complete
f137dfc7a122: Pull complete
f771a35138e1: Pull complete
cf5f8869f4db: Pull complete
504550abb9b1: Retrying in 1 second
fe24e20a0a5a: Download complete
603cd2e526f7: Download complete
```

- Running the container:
- docker run -d --name myrabbit -p 5672:5672 -p 15672:15672 rabbitmq:3-management

```
PS C:\sele\sample-docker-app> docker run -d --name myrabbit -p 5672:5672 -p 15672:15672 rabbitmq:3-management
82bbebed642f36a08ccc541dd31e840ec1d4dc76ef12463bb8ef068e2c5e7fdf
```

- Before Running the image



- After Running the image

Pulling the Python Image From docker Hub

```
PS C:\sele\sample-docker-app> docker pull python:3.9
3.9: Pulling from library/python
cae3b572364a: Pull complete
bd090f42c4b7: Pull complete
f0c9d6d993ac: Pull complete
a2ade626d67a: Pull complete
48638adda46f: Pull complete
74b4209c01d1: Pull complete
5f70f37c9014: Pull complete
Digest: sha256:1ed76782d6e927c60c7e631a3642cee0f4fe786401c73645e3bb646f36137234
Status: Downloaded newer image for python:3.9
docker.io/library/python:3.9
PS C:\sele\sample-docker-app>
```

- **Docker Hub provides ready-made public images** for popular software like Python, MySQL, Nginx, RabbitMQ, etc.
- **No manual download or installation needed** – just pull and run in one command.
- **Auto-pull feature** – if the image is not on your system, Docker automatically fetches it from Docker Hub.
- **Run multiple versions easily** (e.g., Python 3.8, 3.9, 3.10) without affecting your system.
- **Consistency across machines** – the same image works on Windows, Linux, or Mac, avoiding "it works on my machine" issues.
- In short we can directly pull the image and use it without installing the application.

Run Containers Using Docker Compose

- Docker Compose lets you run single or multiple cotainers from images using a YAML file.
- Instead of running docker run manually for each container, you define **services** in the Compose file.
- Command to run docker compose file :

Running in detach Mode:

- Use **docker compose up -d** → Starts all services from docker- compose.yml in the background.
- Containers keep running even after you close the terminal → check them with docker ps.

```
PS C:\Movies_Project\Backend> docker compose up -d
time="2025-09-26T05:44:07+05:30" level=warning msg="C:\\Movies_Project\\Backend\\docker-compose.yml: the attribute `version` is obsol
ete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 8/8
 ✔Container mongodb                  Started                                                                      0.5s
 ✔Container mysql-db                 Started                                                                      0.5s
 ✔Container eureka-server            Started                                                                      0.5s
 ✔Container rabbitmq-server          Started                                                                      0.3s
 ✔Container users-service            Started                                                                      0.8s
 ✔Container api-gateway              Started                                                                      0.9s
 ✔Container moviedetails-service     Started                                                                      1.1s
 ✔Container moviesuggestions-service Started                                                                      1.4s
PS C:\Movies_Project\Backend>
```

- Viewing the Running Containers:
- To View the running containers we use the command
- docker ps

```
PS C:\Movies_Project\Backend> docker ps
CONTAINER ID   IMAGE                              COMMAND                  CREATED        STATUS          PORTS
                                                                                    NAMES
8df8b7c86b23   yuvanagavenkat/moviesuggestions   "/cnb/process/web"       10 hours ago   Up 6 minutes
                                                                                    moviesuggestions-service
2ddc0453b5c7   yuvanagavenkat/moviedetails       "/cnb/process/web"       10 hours ago   Up 6 minutes
                                                                                    moviedetails-service
d8ebd1e406a5   yuvanagavenkat/users              "/cnb/process/web"       10 hours ago   Up 6 minutes
                                                                                    users-service
b121d9b49355   yuvanagavenkat/gateway            "/cnb/process/web"       10 hours ago   Up 6 minutes    0.0.0.0:8080->8080/tcp
                                                                                    api-gateway
bf3653f7976f   mysql:8.0                         "docker-entrypoint.s…"   10 hours ago   Up 6 minutes    33060/tcp, 0.0.0.0:3307->330
6/tcp                                                                               mysql-db
efda1d94c3a3   rabbitmq:3-management             "docker-entrypoint.s…"   10 hours ago   Up 6 minutes    4369/tcp, 5671/tcp, 0.0.0.0:
5672->5672/tcp, 15671/tcp, 15691-15692/tcp, 25672/tcp, 0.0.0.0:15672->15672/tcp   rabbitmq-server
715e2fb741b8   mongo:latest                      "docker-entrypoint.s…"   10 hours ago   Up 6 minutes    0.0.0.0:27017->27017/tcp
                                                                                    mongodb
61fb828d3941   yuvanagavenkat/servicereg         "/cnb/process/web"       10 hours ago   Up 6 minutes    0.0.0.0:8761->8761/tcp
                                                                                    eureka-server
b0e396b7bfcf   dpage/pgadmin4                    "/entrypoint.sh"         7 weeks ago    Up 10 minutes   443/tcp, 0.0.0.0:5050->80/tc
p                                                                                   pgadmin_container
997a8c7f9e0d   postgres                          "docker-entrypoint.s…"   7 weeks ago    Up 10 minutes   0.0.0.0:5432->5432/tcp
                                                                                    postgres_container
PS C:\Movies_Project\Backend>
```

Stopping the Container

- To stop the running container we use the command
- docker stop <container_id_or_name>

```
PS C:\Movies_Project\Backend> docker stop 8df8b7c86b2
8df8b7c86b23
PS C:\Movies_Project\Backend>
```

```
PS C:\Movies_Project\Backend> docker compose down
time="2025-09-26T05:58:29+05:30" level=warning msg="C:\\Movies_Project\\Backend\\docker-compose.yml: the attribute `version` is obsol
ete, it will be ignored, please remove it to avoid potential confusion"
[+] Running 9/9
 ✔Container api-gateway                Removed                                                                          6.6s
 ✔Container moviesuggestions-service   Removed                                                                          0.1s
 ✔Container moviedetails-service       Removed                                                                          4.0s
 ✔Container rabbitmq-server            Removed                                                                          2.8s
 ✔Container users-service              Removed                                                                          4.1s
 ✔Container mongodb                    Removed                                                                          0.9s
 ✔Container mysql-db                   Removed                                                                          1.8s
 ✔Container eureka-server              Removed                                                                          0.8s
 ✔Network backend_microservices-net    Removed                                                                          1.0s
PS C:\Movies_Project\Backend> docker ps
CONTAINER ID   IMAGE           COMMAND                  CREATED       STATUS          PORTS                                NAMES
b0e396b7bfcf   dpage/pgadmin4  "/entrypoint.sh"         7 weeks ago   Up 18 minutes   443/tcp, 0.0.0.0:5050->80/tcp        pgadmin_contai
ner
997a8c7f9e0d   postgres        "docker-entrypoint.s…"   7 weeks ago   Up 18 minutes   0.0.0.0:5432->5432/tcp               postgres_conta
iner
PS C:\Movies_Project\Backend>
```

Checking the running containers after stopping one container :

```
PS C:\Movies_Project\Backend> docker ps
CONTAINER ID   IMAGE                            COMMAND                  CREATED        STATUS          PORTS
          NAMES
2ddc0453b5c7   yuvanagavenkat/moviedetails      "/cnb/process/web"       10 hours ago   Up 11 minutes
          moviedetails-service
d8ebd1e406a5   yuvanagavenkat/users             "/cnb/process/web"       10 hours ago   Up 11 minutes
          users-service
b121d9b49355   yuvanagavenkat/gateway           "/cnb/process/web"       10 hours ago   Up 11 minutes   0.0.0.0:8080->8080/tcp
          api-gateway
bf3653f7976f   mysql:8.0                        "docker-entrypoint.s…"   10 hours ago   Up 11 minutes   33060/tcp, 0.0.0.0:3307->3306/tc
p         mysql-db
efda1d94c3a3   rabbitmq:3-management            "docker-entrypoint.s…"   10 hours ago   Up 11 minutes   4369/tcp, 5671/tcp, 0.0.0.0:5672
->5672/tcp, 15671/tcp, 15691-15692/tcp, 25672/tcp, 0.0.0.0:15672->15672/tcp   rabbitmq-server
715e2fb741b8   mongo:latest                     "docker-entrypoint.s…"   10 hours ago   Up 11 minutes   0.0.0.0:27017->27017/tcp
          mongodb
61fb828d3941   yuvanagavenkat/servicereg        "/cnb/process/web"       10 hours ago   Up 11 minutes   0.0.0.0:8761->8761/tcp
          eureka-server
b0e396b7bfcf   dpage/pgadmin4                   "/entrypoint.sh"         7 weeks ago    Up 15 minutes   443/tcp, 0.0.0.0:5050->80/tcp
          pgadmin_container
997a8c7f9e0d   postgres                         "docker-entrypoint.s…"   7 weeks ago    Up 15 minutes   0.0.0.0:5432->5432/tcp
          postgres_container
PS C:\Movies_Project\Backend>
```

Checking all the containers Running as well as Stopped:

- docker ps -a

```
PS C:\Movies_Project\Backend> docker ps -a
CONTAINER ID   IMAGE                              COMMAND                  CREATED        STATUS                       PORTS
          NAMES
8df8b7c86b23   yuvanagavenkat/moviesuggestions    "/cnb/process/web"       10 hours ago   Exited (143) 3 minutes ago
          moviesuggestions-service
2ddc0453b5c7   yuvanagavenkat/moviedetails        "/cnb/process/web"       10 hours ago   Up 12 minutes
          moviedetails-service
d8ebd1e406a5   yuvanagavenkat/users               "/cnb/process/web"       10 hours ago   Up 12 minutes
          users-service
b121d9b49355   yuvanagavenkat/gateway             "/cnb/process/web"       10 hours ago   Up 12 minutes                0.0.0.0:8080->8
080/tcp   api-gateway
bf3653f7976f   mysql:8.0                          "docker-entrypoint.s…"   10 hours ago   Up 12 minutes                33060/tcp, 0.0.
0.0:3307->3306/tcp   mysql-db
efda1d94c3a3   rabbitmq:3-management              "docker-entrypoint.s…"   10 hours ago   Up 12 minutes                4369/tcp, 5671/
tcp, 0.0.0.0:5672->5672/tcp, 15671/tcp, 15691-15692/tcp, 25672/tcp, 0.0.0.0:15672->15672/tcp   rabbitmq-server
715e2fb741b8   mongo:latest                       "docker-entrypoint.s…"   10 hours ago   Up 12 minutes                0.0.0.0:27017->
27017/tcp   mongodb
61fb828d3941   yuvanagavenkat/servicereg          "/cnb/process/web"       10 hours ago   Up 12 minutes                0.0.0.0:8761->8
761/tcp   eureka-server
12aabd971ae1   yuvanagavenkat/servicereg          "/cnb/process/web"       10 hours ago   Exited (130) 10 hours ago
          boring_raman
```

Stopping the Containers By Docker Compose File:

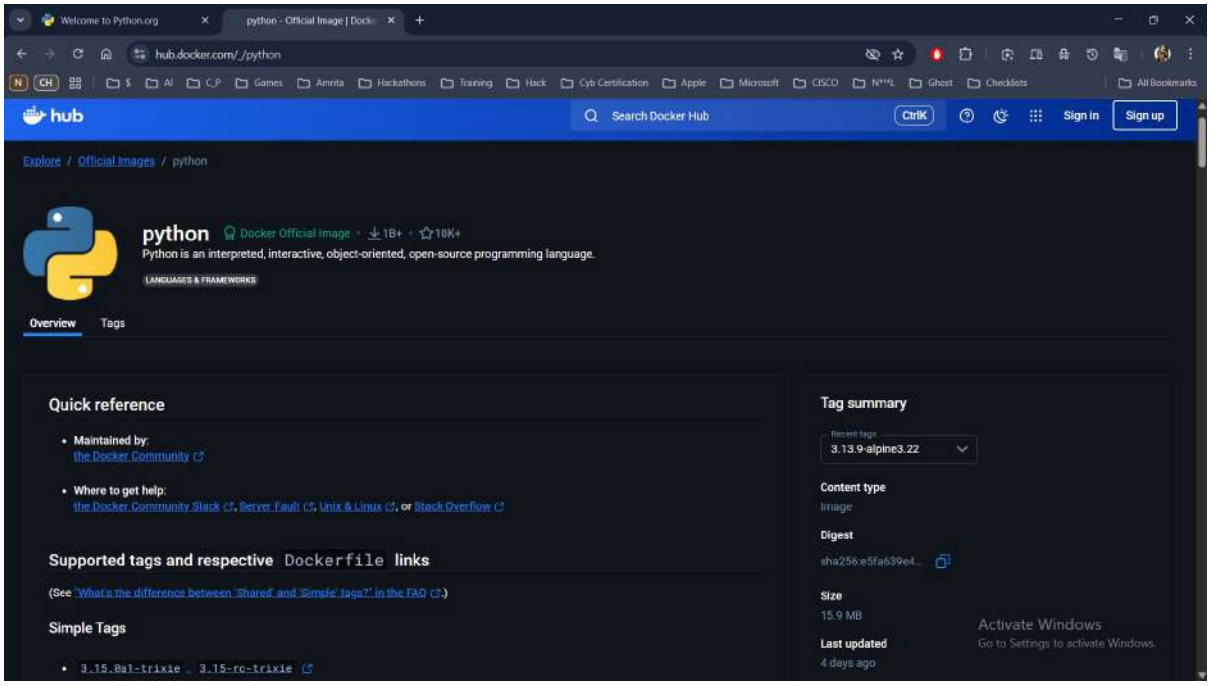
Removing a Container From Local System:

- To remove container we use the following command
- docker rm <container_id_or_name>
- Make Sure that the docker desktop is opened in the background to run all the commands.

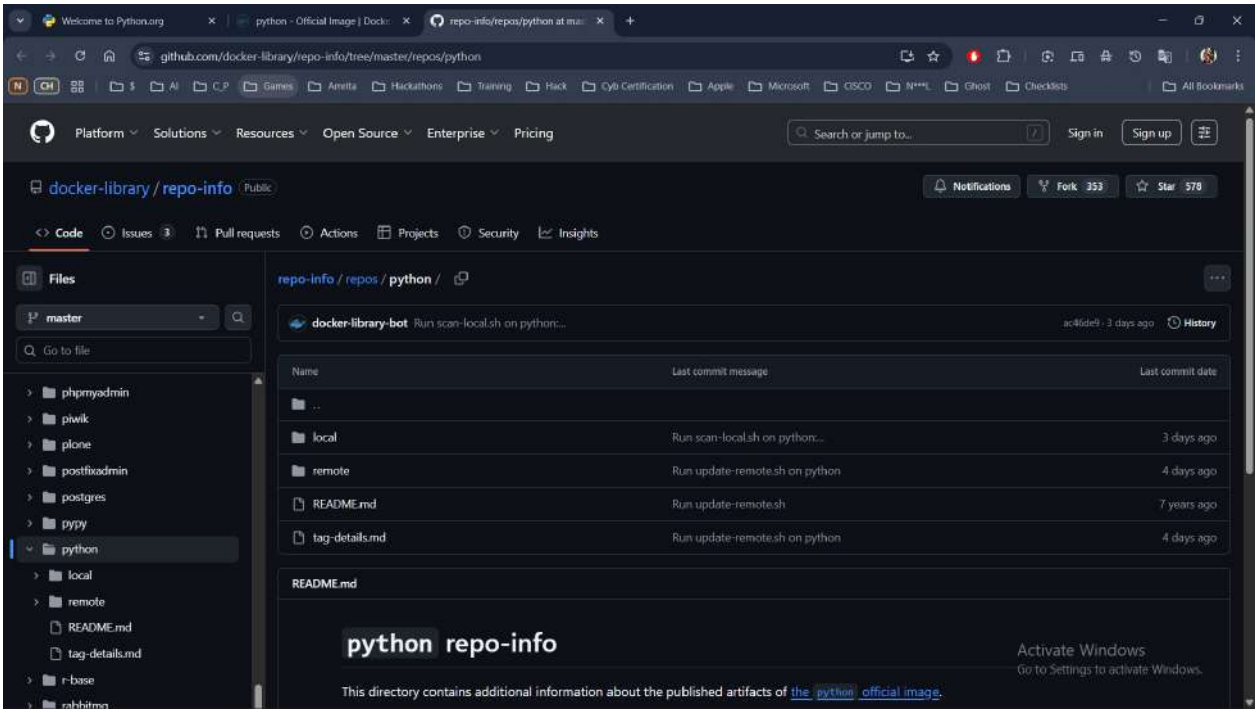
```
PS C:\Movies_Project\Backend> docker rm 997a8c7f9e0d
997a8c7f9e0d
```

Viewing the docker file of public images

- Go to the docker hub.
- Click on any one image.



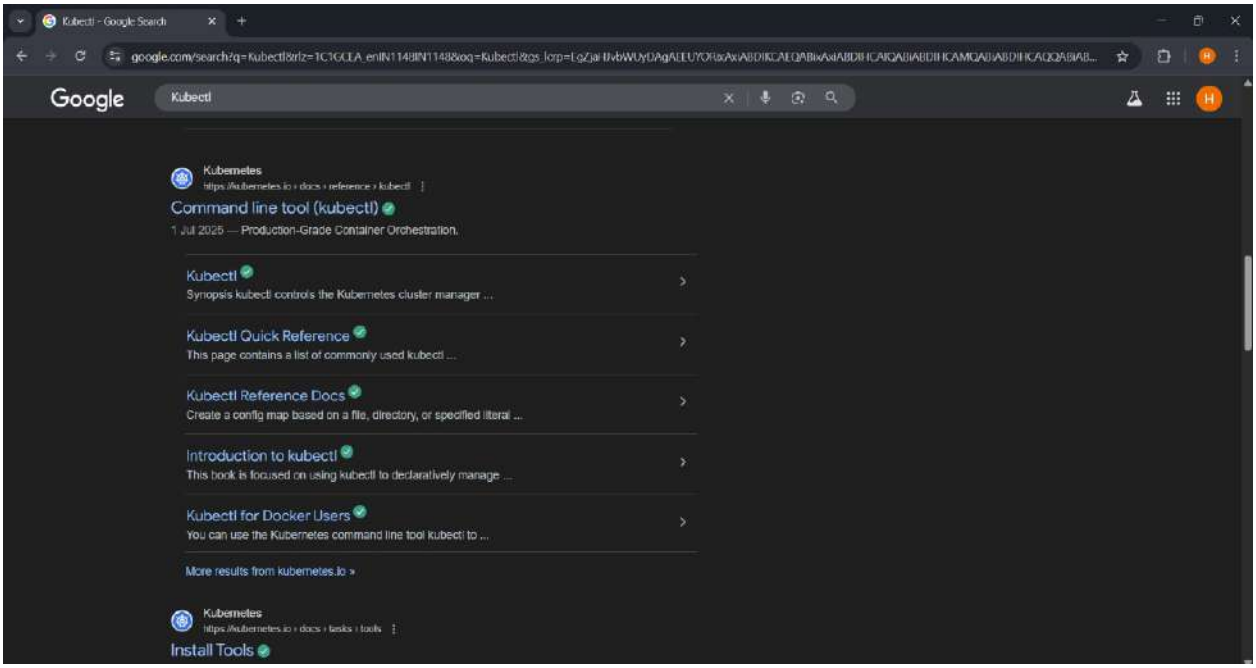
- Click on the links under Supported tags and respective Dockerfile links.
- We can view the docker files in the Github.



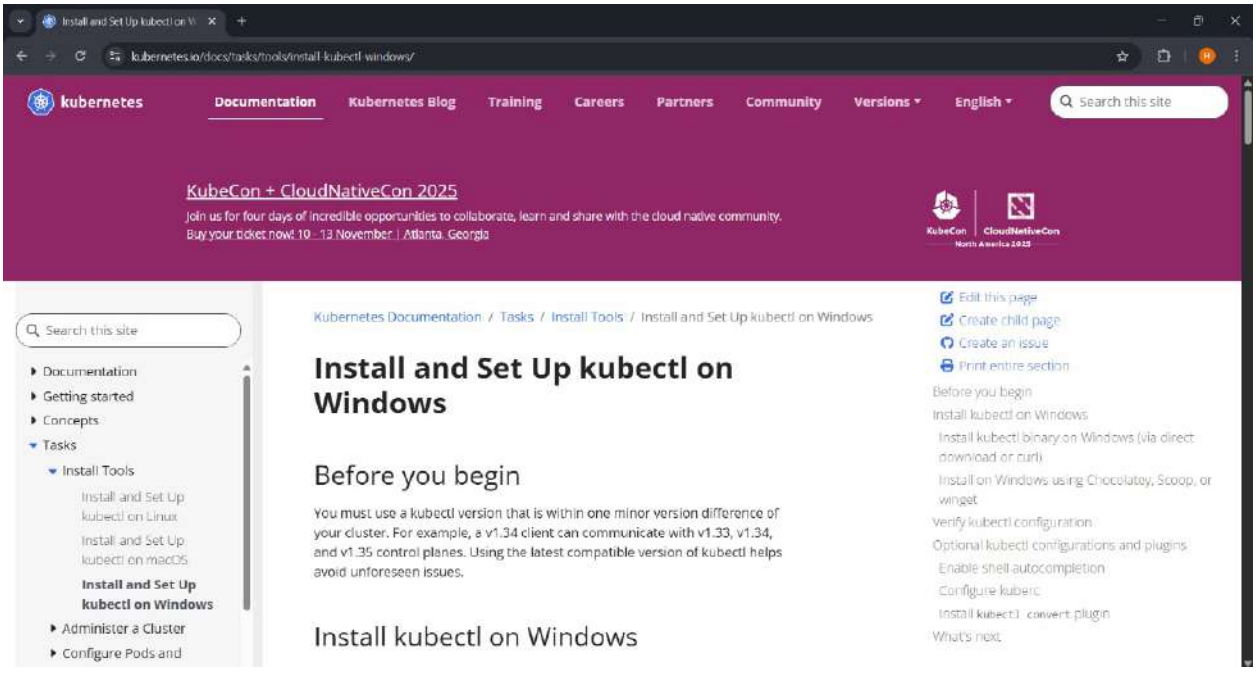
Experiment-10: Kubernetes – Container Orchestration

Aim: To demonstrate container orchestration using Kubernetes.

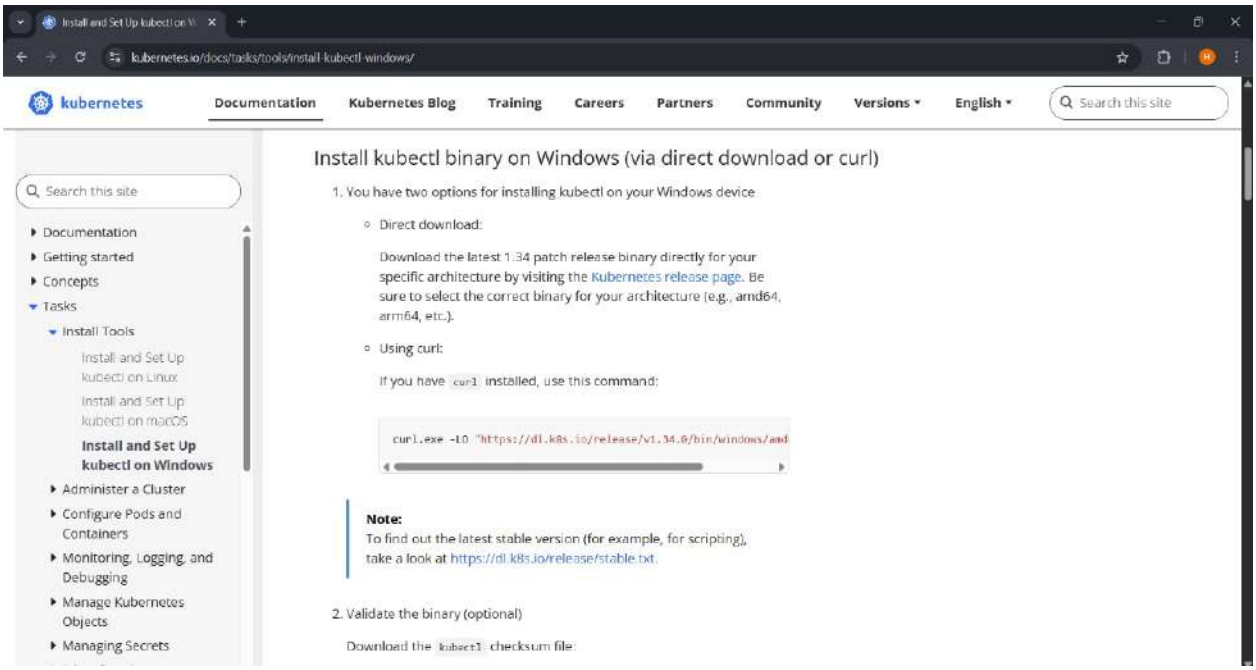
Step-1: Open Google Chrome and then Search for Kubectl.



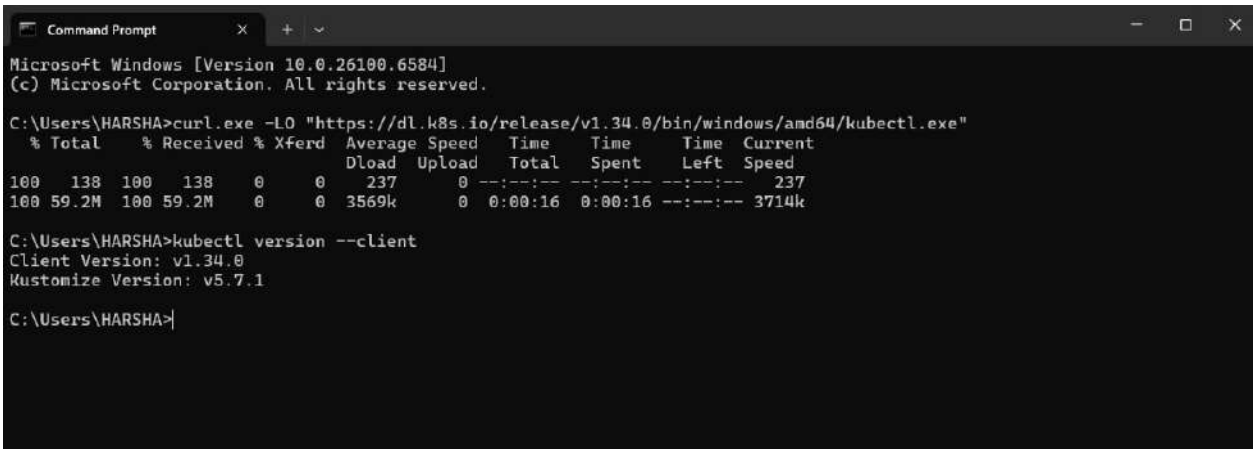
Step-2: Select the Type of Operating System like Windows,Linux or MacOS and copy the command.



Step-3: To install the kubectl RUN the command.



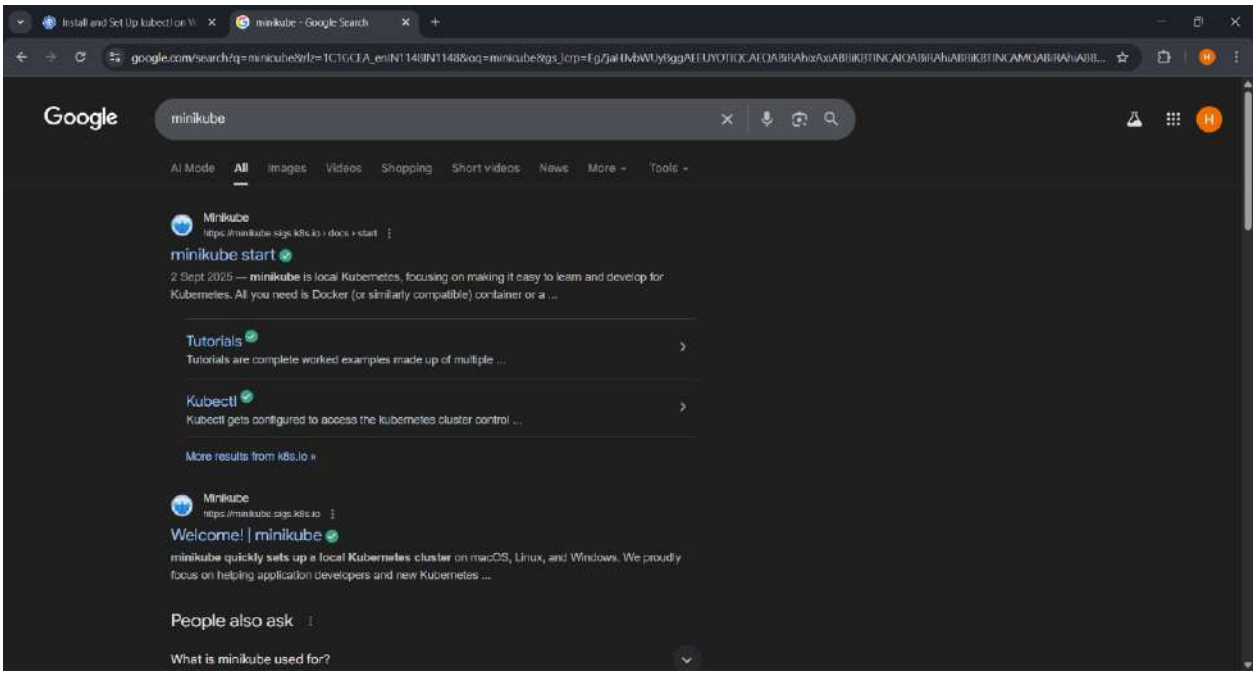
Step-4: We can see the Installation of Kubectl , Check the version of the kubectl. By using the **kubectl version.**



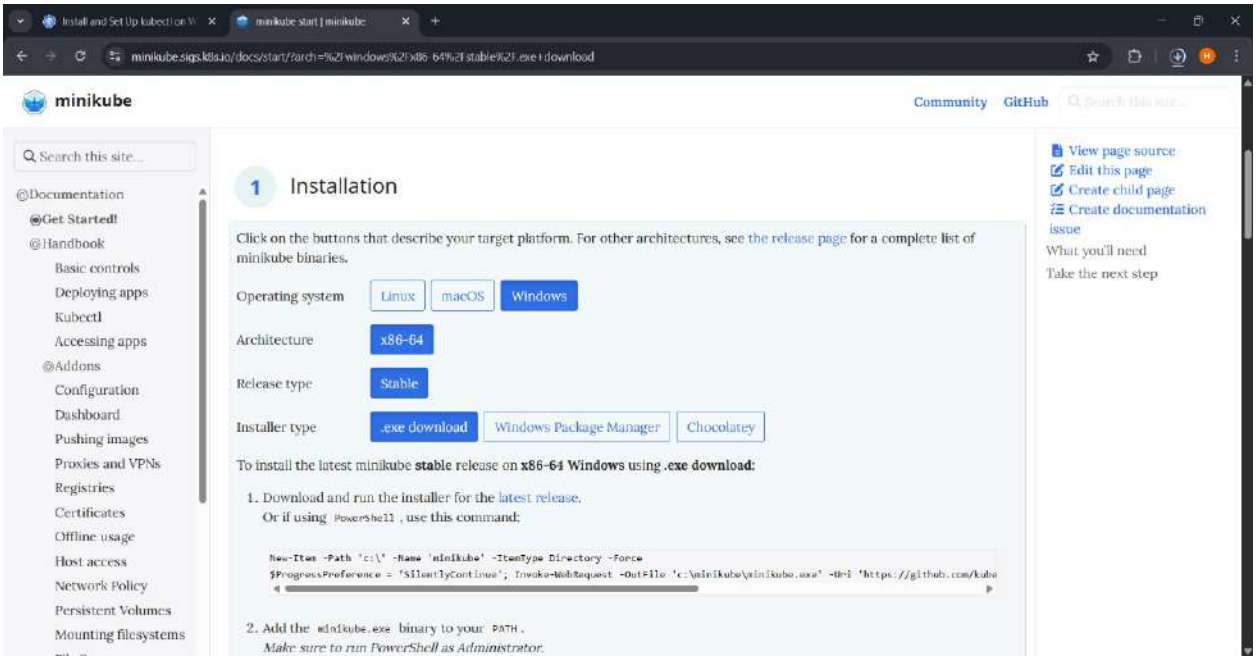
Step-5: Now we proceed with the Installation of the local kubernetes cluster.

Like : minikube,k3s,kind---------- etc.

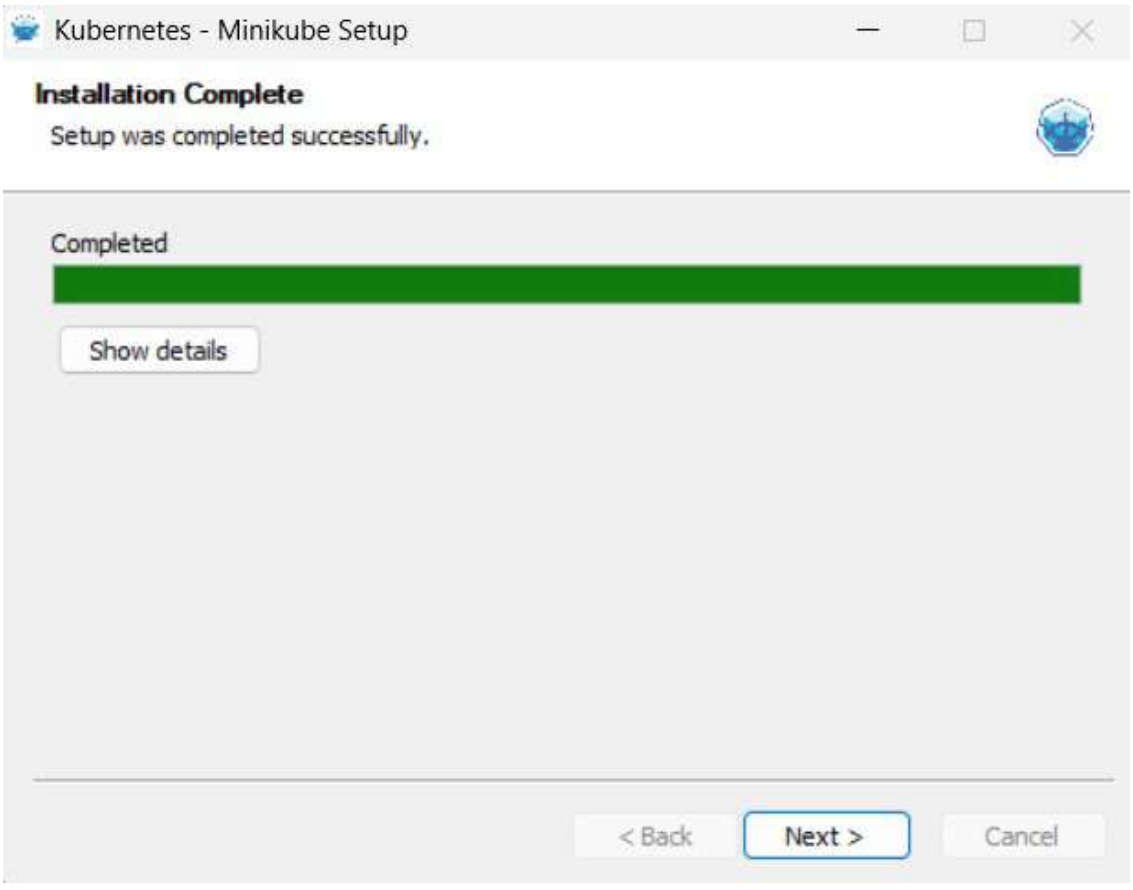
Step-6: Install the Minikube.



Step-7: Select the operating system of your computer then copy the command.



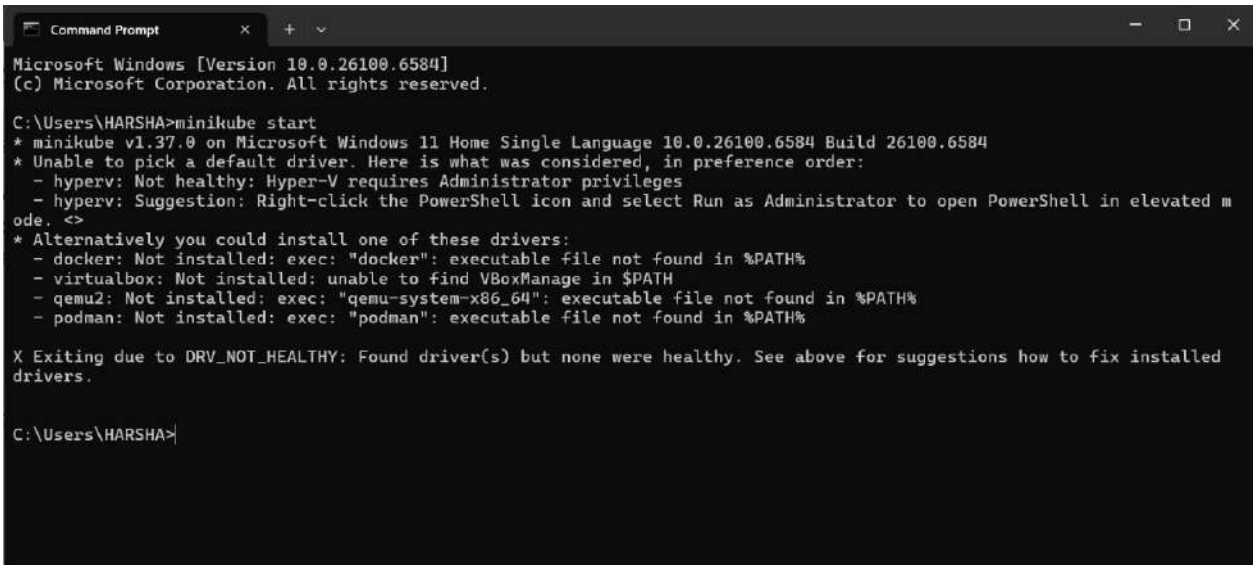
Step-8: RUN the command you can see the installation of minikube as



Step-9: we use Virtual machines for single node kubernetes cluster.

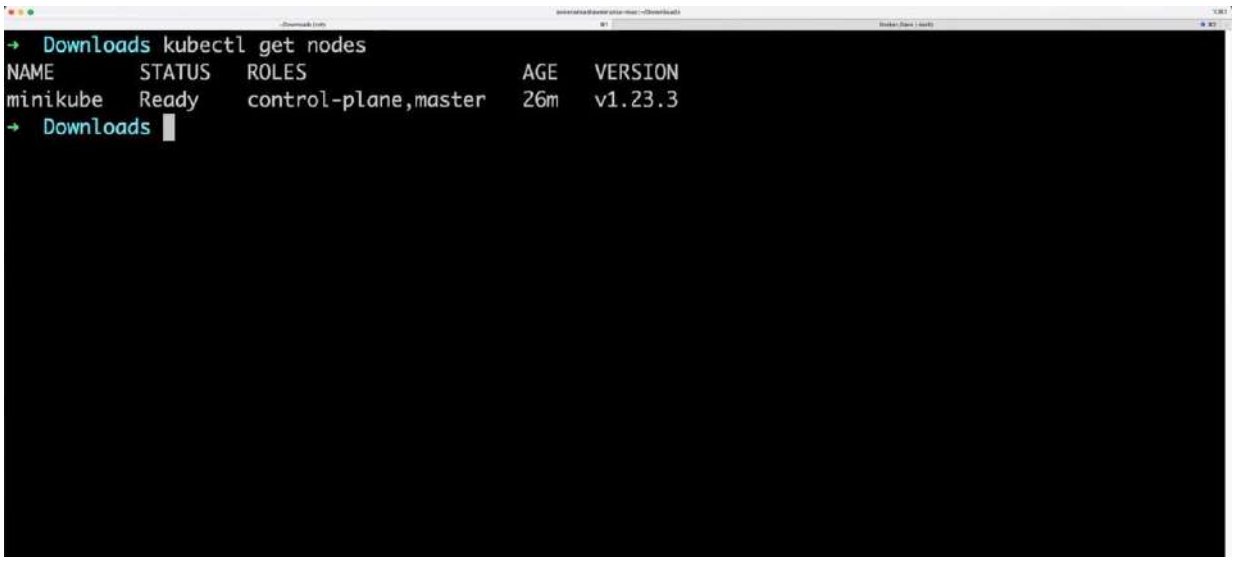
Run the command minikube start

The kubernetes cluster by default uses your Docker driver.

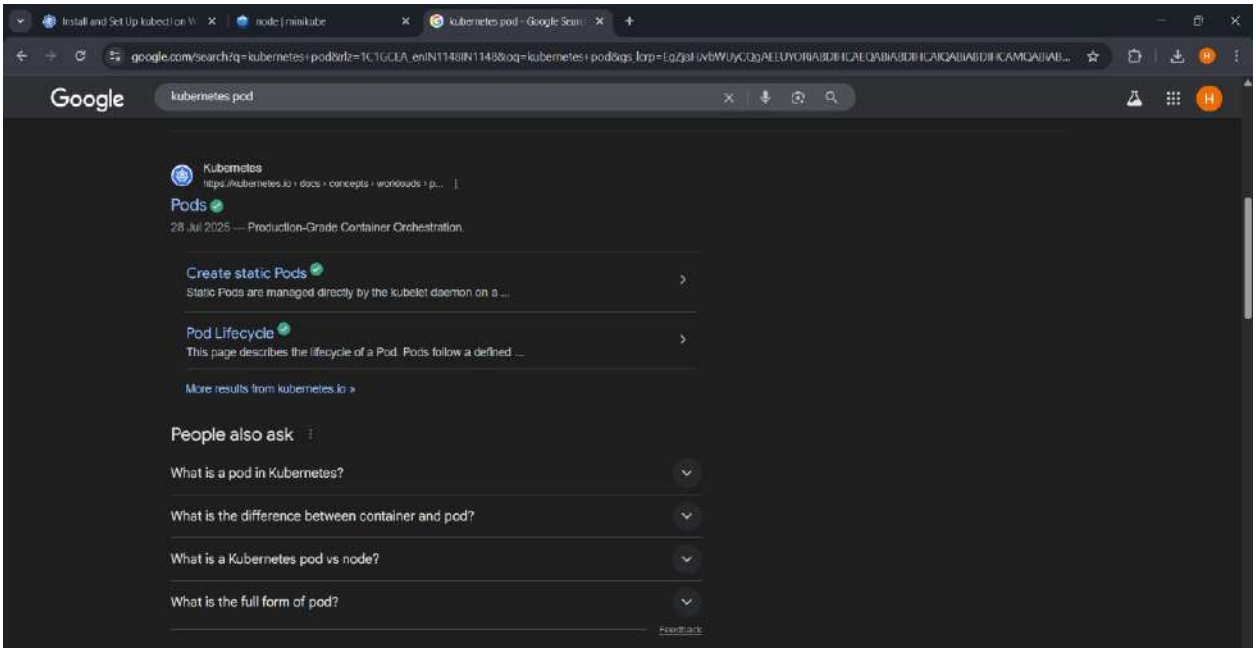


Step-10: kubectl is already connected to your kubernetes cluster and it is saying that there is one node that is running .

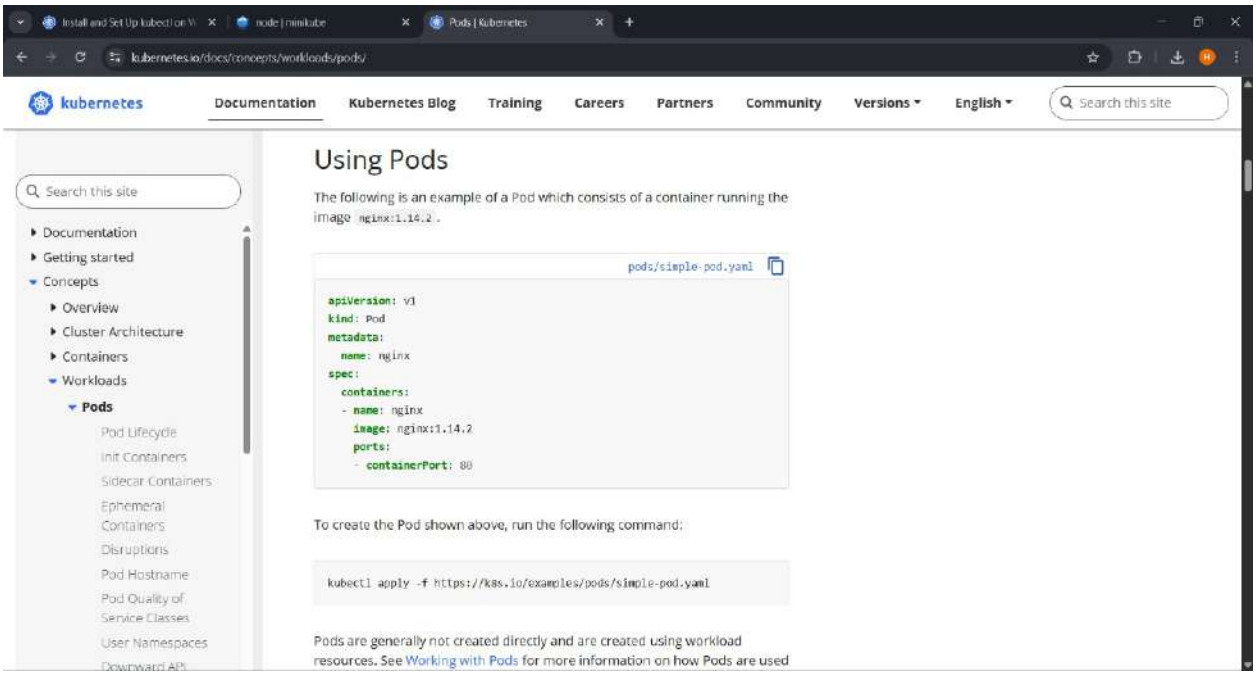
- The node is called minikube
- You can check the status of Nodes in the kubectl.



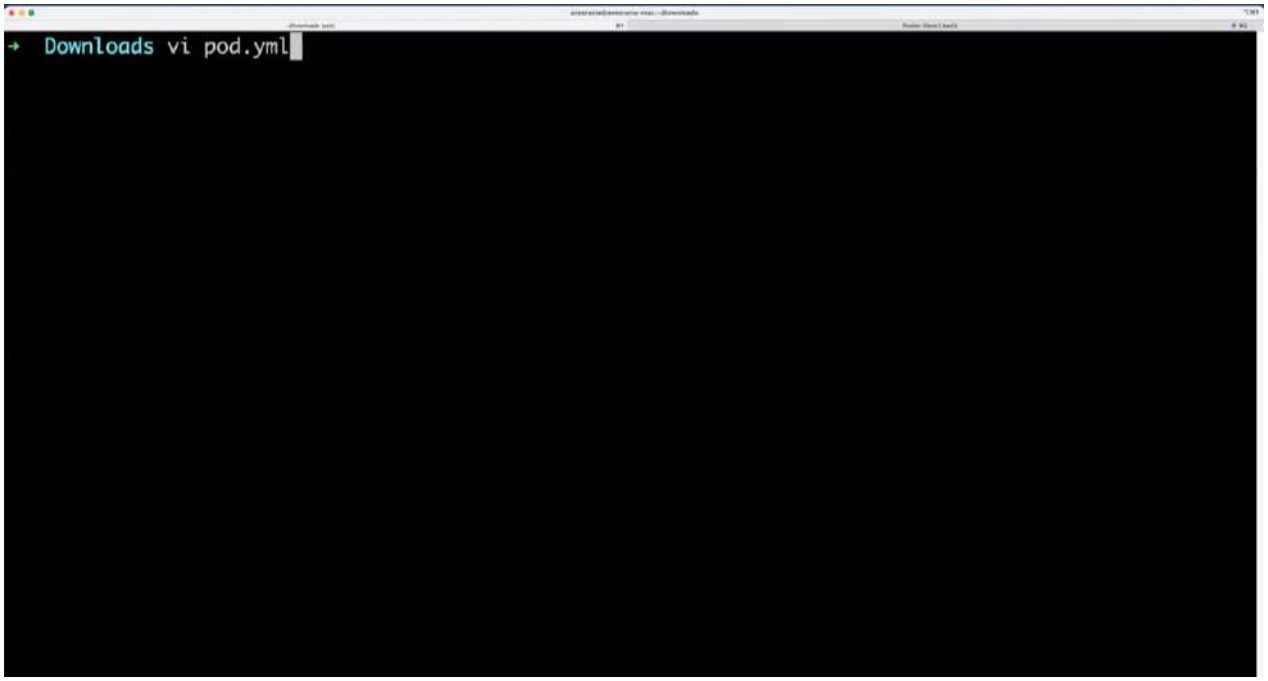
Step-11: open your google and then search for the **"Kubernetes POD"** for installation.



Step-12: To install the POD , copy the yaml file.

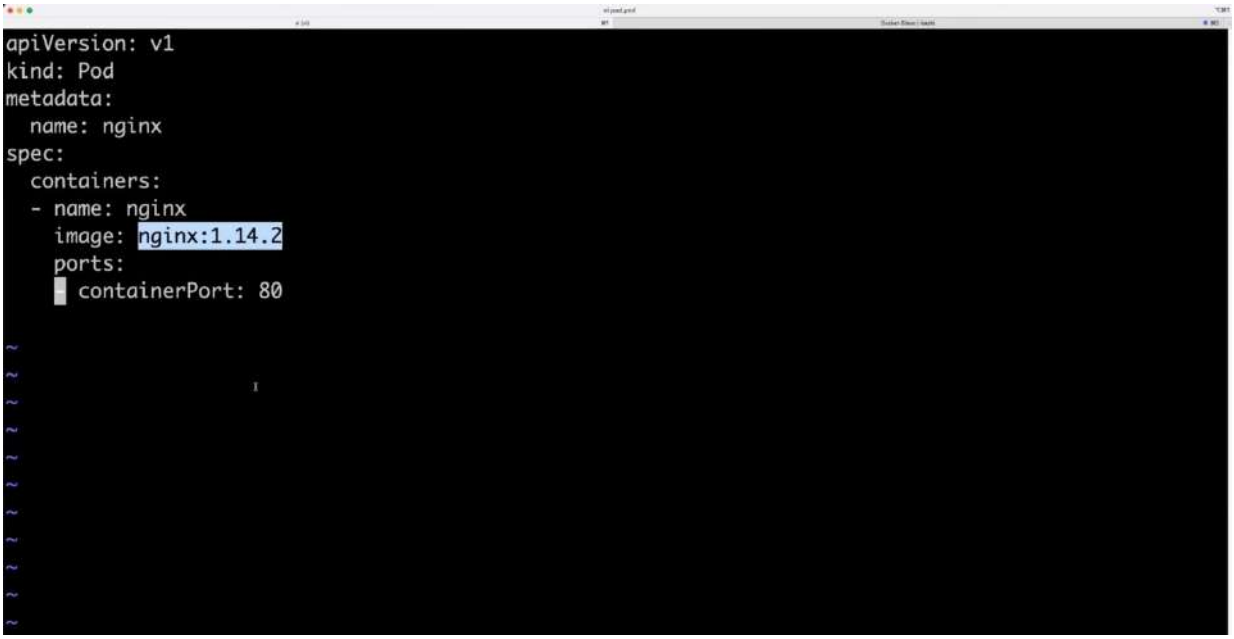


Step-13: Create a new yaml file as Shown below.



Step-14: Paste the copied yaml in this new file.

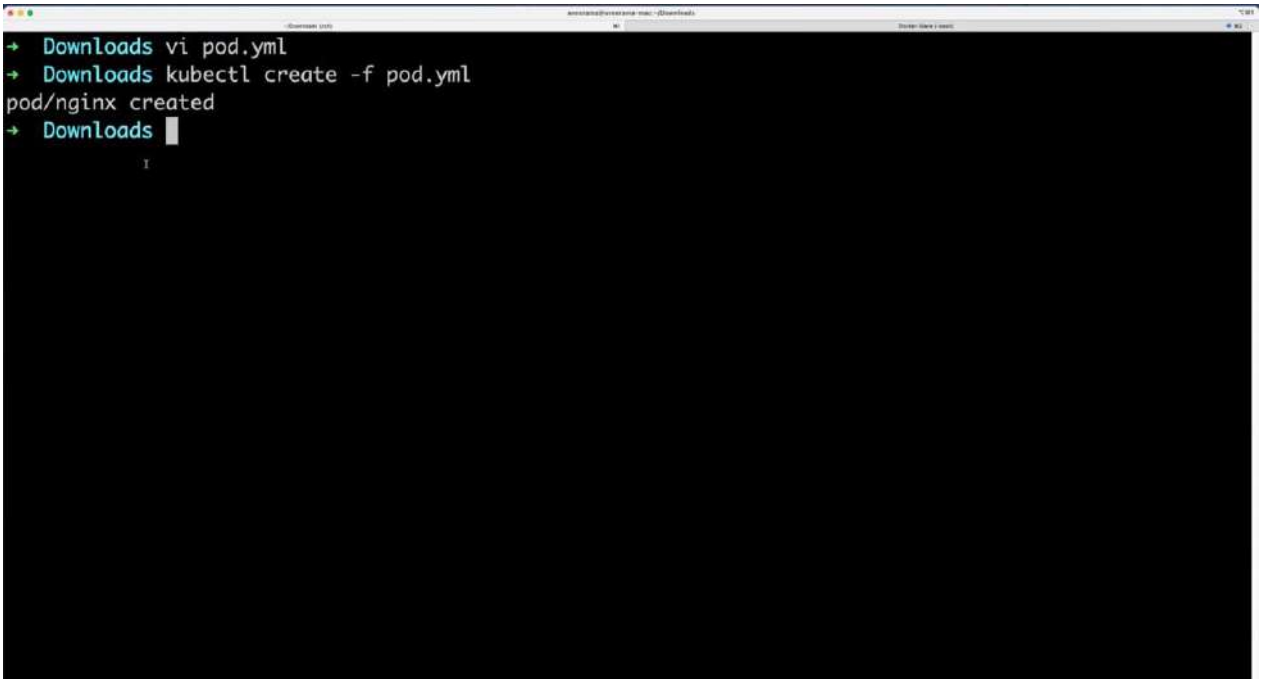
- This pod.yaml is a specification of the docker container.



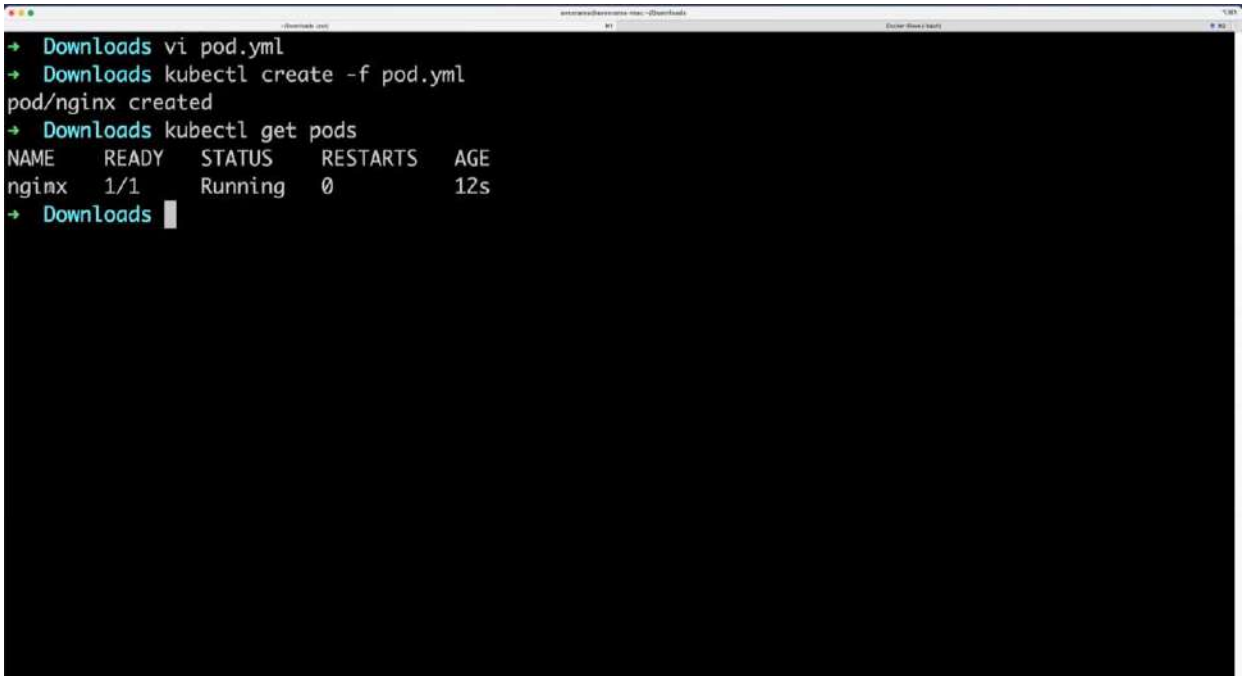
Step-15: Use this command kubectl which is similar to Docker CLI.

- Here it is kubernetes CLI as kubectl create -f pod.yml To create

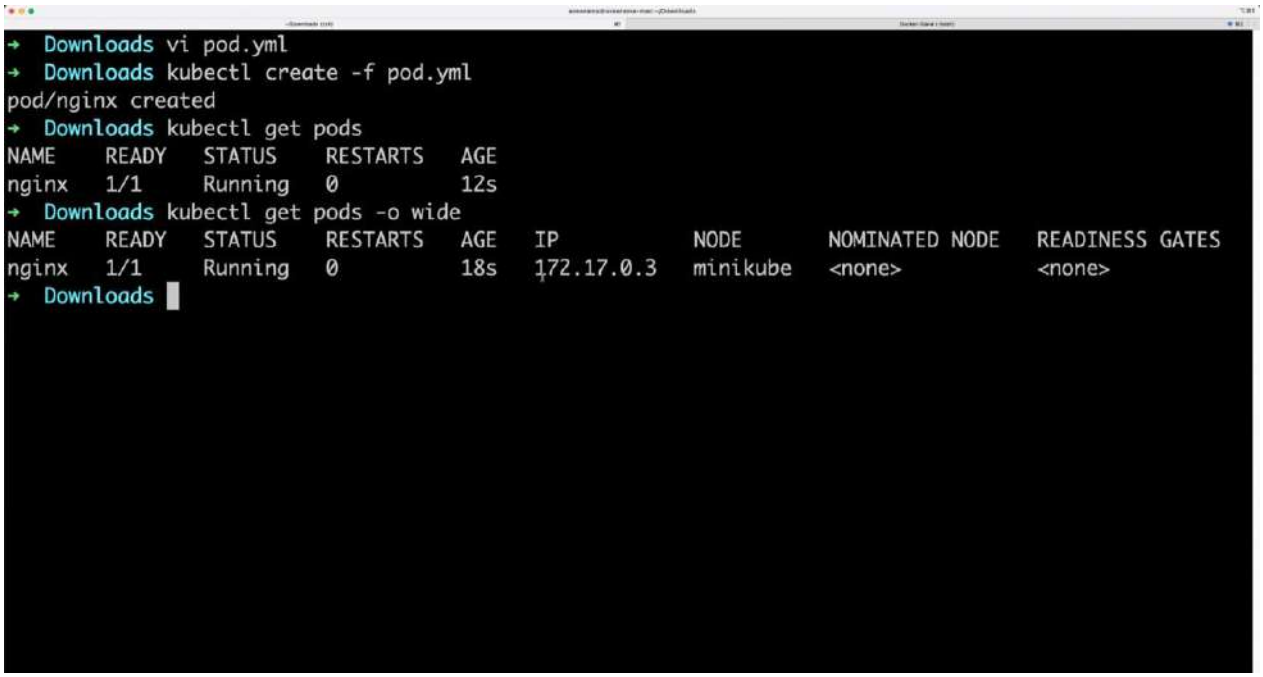
pod use this command



Step-16: To check the pods use **kubectl get pods**

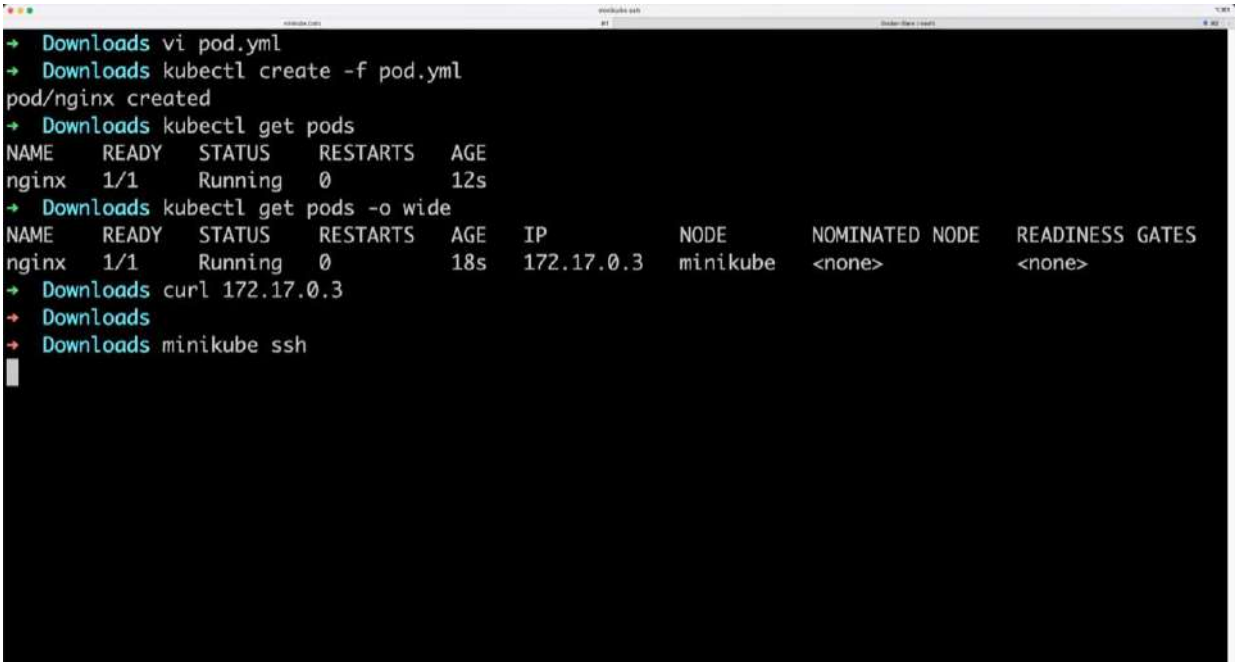


Step-17: If you do **kubectl get pods -o wide** then it prints the entire details of the Pod.

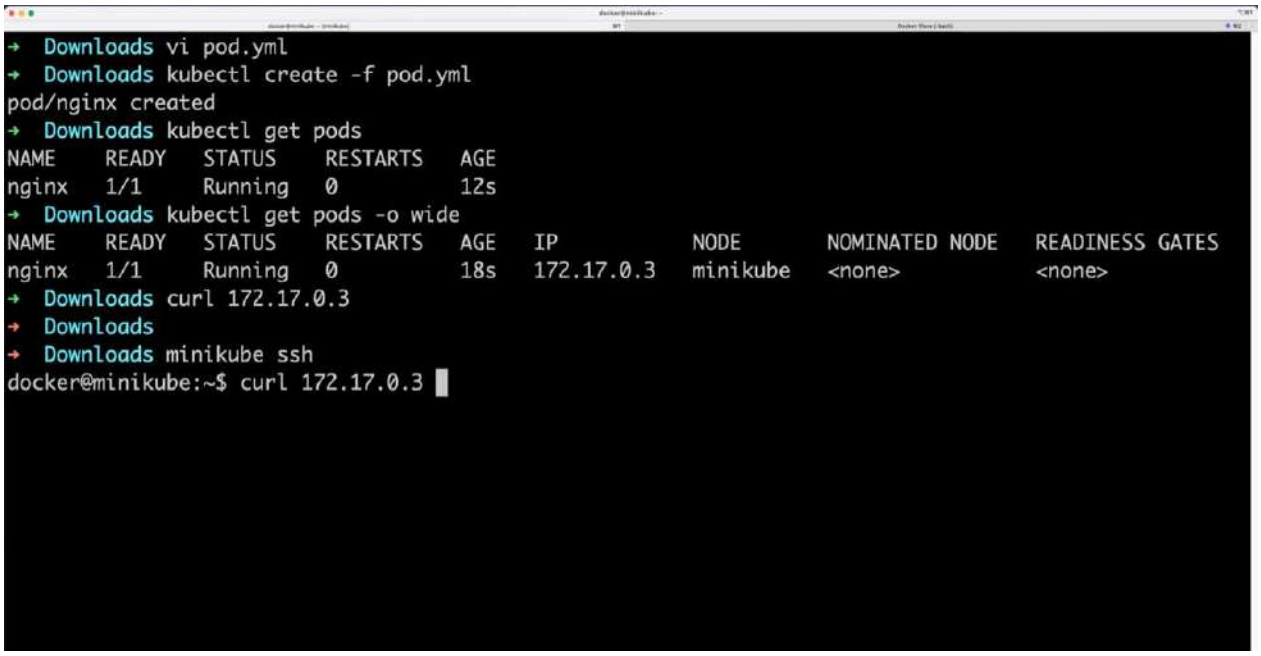


Step-17.1: If we are not exposing this Application from docker container to external application we should Login to kubernetes cluster.

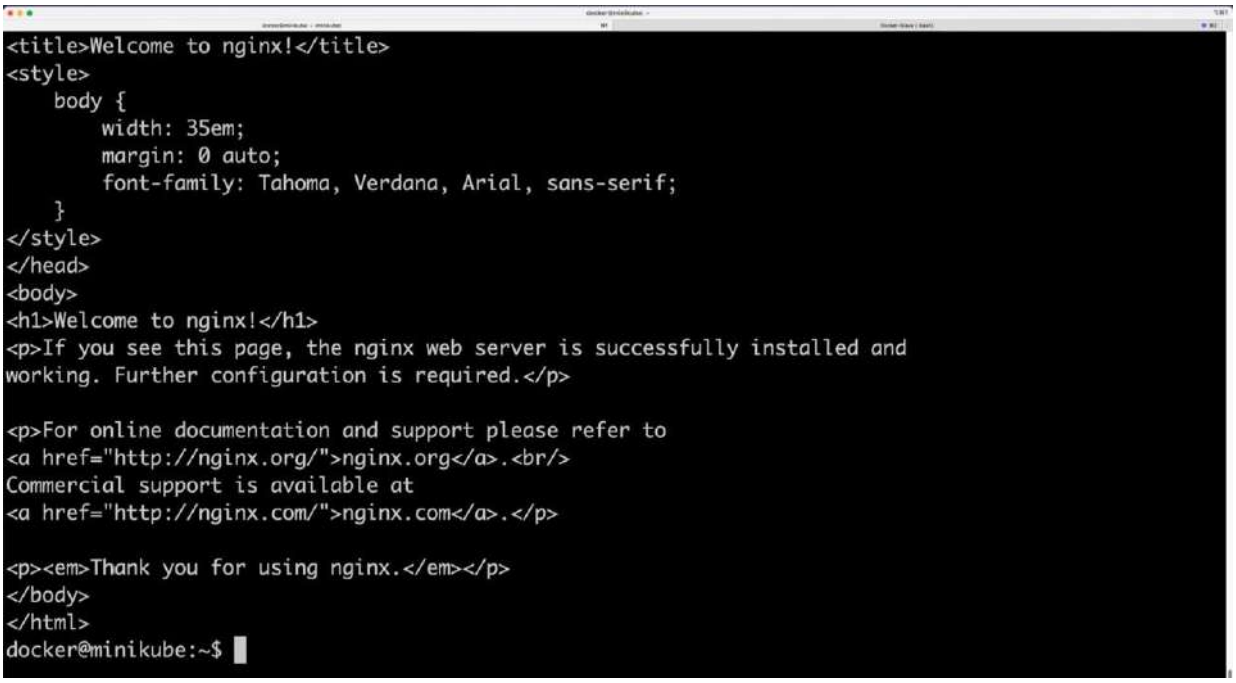
- Command is minikube ssh



Step-18: If we use realtime kubernetes cluster then , use Ip Address.

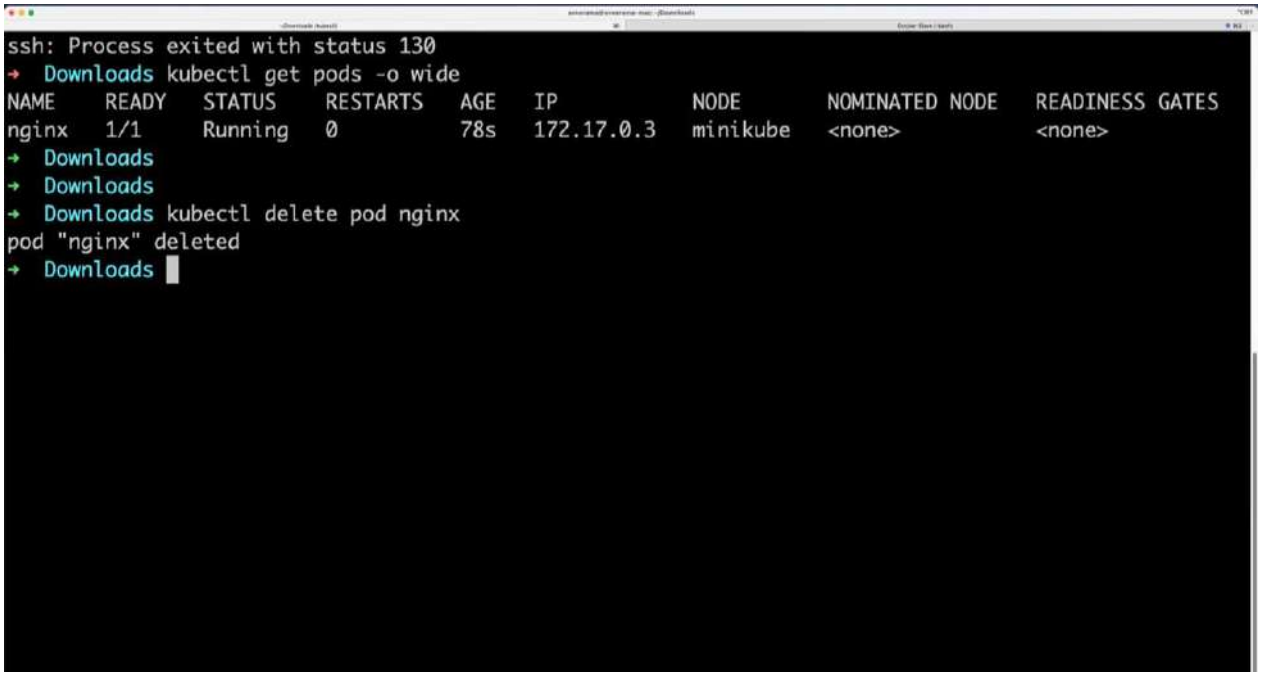


Step-19: it shows the Thank you for using the nginx , it means you have successfully created the kubernetes Application and Executed.



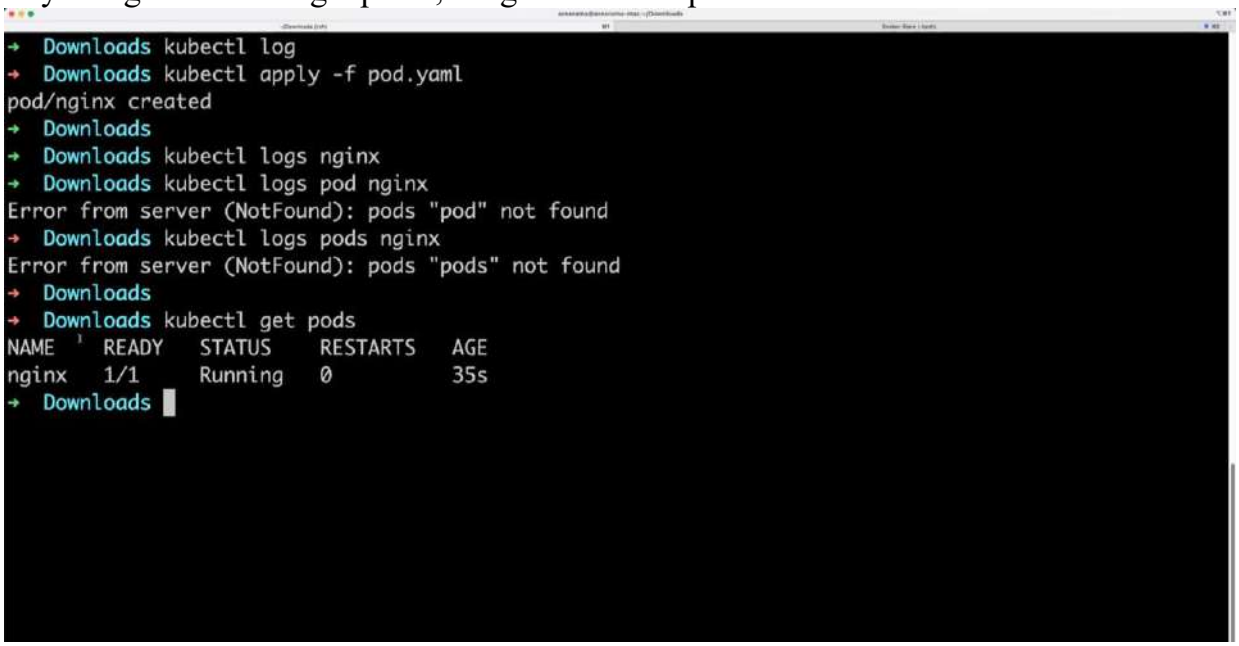
Step-20: We can check the status of the application.

- If we want to delete the Pod then we can delete the Pod by providing the name of the Pod.
- example:nginx



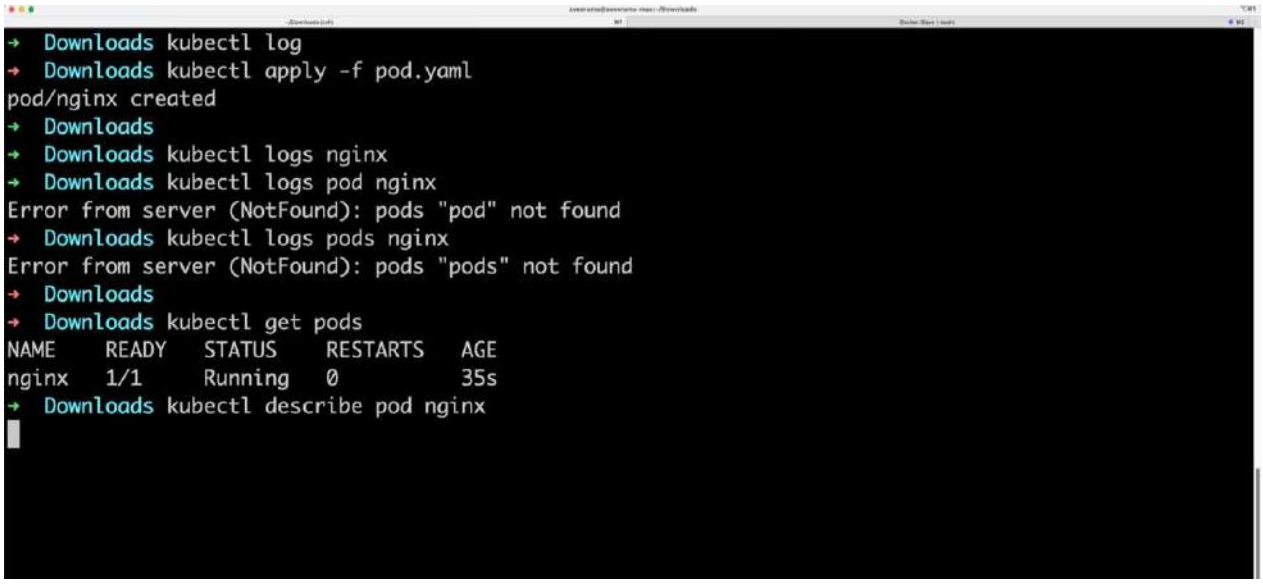
Step-21: we can also check the logs of the kubernetes Pod.

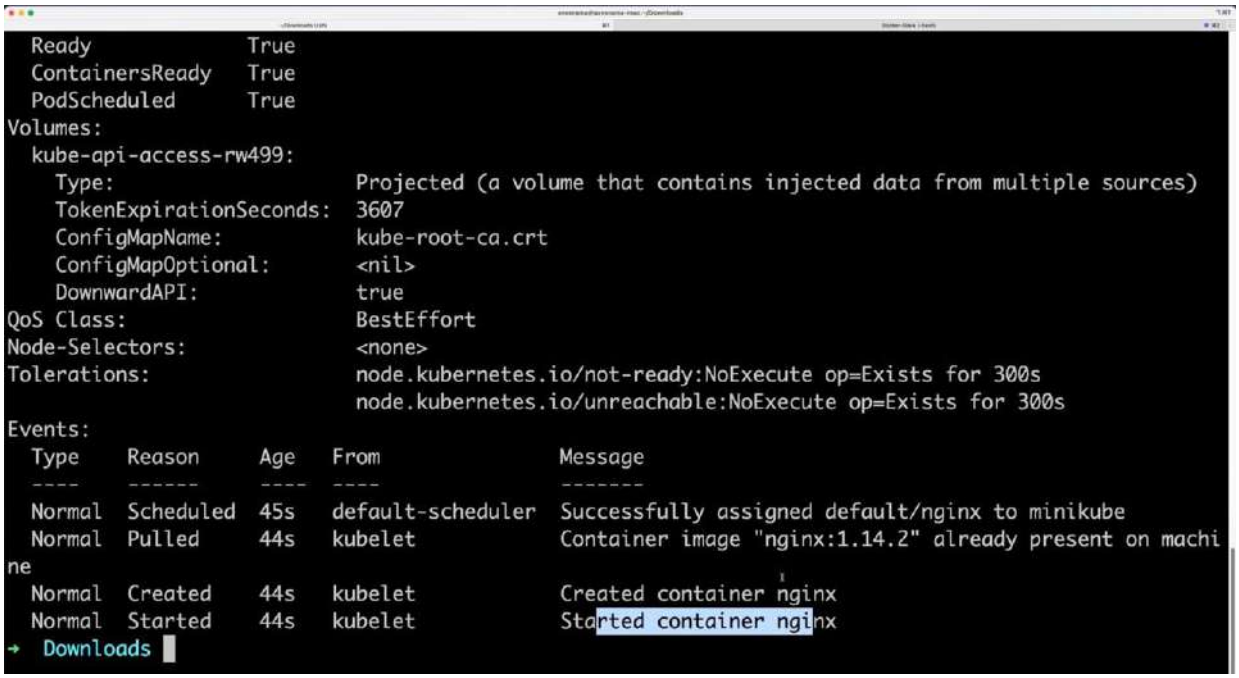
- By using the kubectl get pods , we get the entire pod information.



Step-22: if you run this command **kubectl describe pod nginx.**

- Here nginx is the name of the Pod.
- it will print the all the information of your Pod.

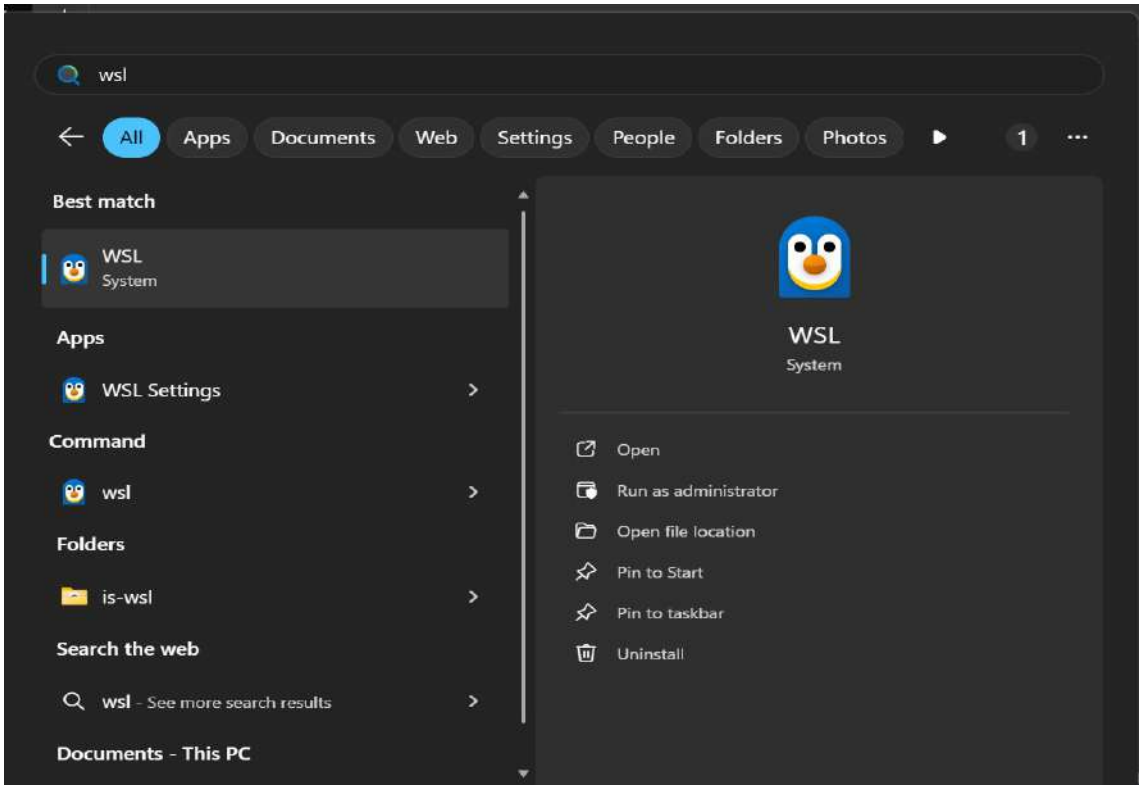




Experiment11: Ansible – Configuration Management

Aim: To automate configuration management using Ansible.

1. **Install WSL (Windows Subsystem for Linux)**
   Run the following commands in cmd
   wsl --install
   wsl --update
   wsl --set-default-version 2
   open the wsl after installing



2. **Update Ubuntu and Install Dependencies**

   Run the following commands in linux shell

   sudo apt update && sudo apt upgrade -y
   sudo apt install python3 python3-pip python3-venv -y
   sudo apt install unzip wget -y

```
harsha@LAPTOP-DR0MMK6C:~$ sudo apt update && sudo apt upgrade -y
[sudo] password for harsha:
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1222 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [204 kB]
Get:6 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [21.6 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [8968 B]
Get:8 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [892 kB]
Get:9 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [198 kB]
Get:11 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.2 kB]
Get:12 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [18.3 kB]
Get:13 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [1978 kB]
Get:14 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [450 kB]
Get:15 http://archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:16 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [208 B]
Get:17 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [520 B]
Get:18 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [28.0 kB]
Get:19 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [5844 B]
Get:20 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:21 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [384 B]
Get:22 http://archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:23 http://archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:24 http://archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:25 http://archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:26 http://archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:27 http://archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:28 http://archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
```

3. **Install AWS CLI**

   Run the following commands in linux shell

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

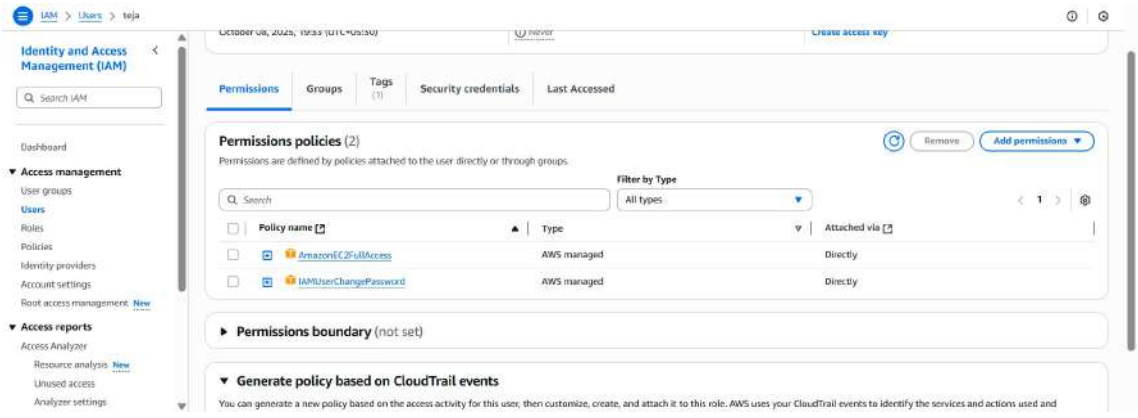
   unzip awscliv2.zip

   sudo ./aws/install

   aws –version

```
harsha@LAPTOP-DR0MMK6C:~$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip".
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 59.3M  100 59.3M    0     0  3758k      0  0:00:16  0:00:16 --:--:-- 3855k
harsha@LAPTOP-DR0MMK6C:~$
```

4. **Create a User in AWS IAM and create Access Key with EC2 access**



5. **Configure AWS CLI**

   Run : aws configure --profile terraform

   Enter the details from the previous step in the shell

6. **Set up Ansible**

   6.1 Create a Python virtual environment

   Run the following commands in Linux shell

   python3 -m venv ~/ansible-env

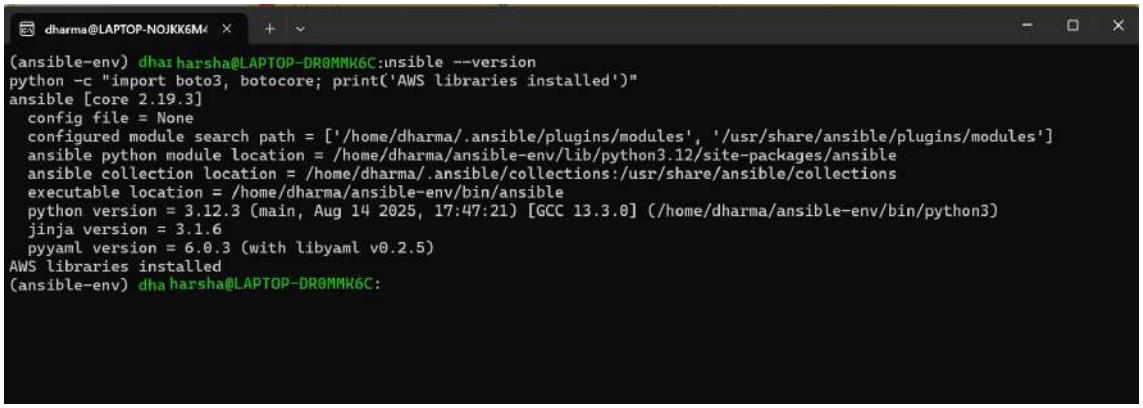
   source ~/ansible-env/bin/activate

   6.2 Install Ansible and AWS Python libraries

   Run the following commands in Linux shell

   pip install --upgrade pip

   pip install ansible boto3 botocore amazon.aws



7. **Create an Ansible hosts file**
   Run **echo "localhost ansible_connection=local" > ~/hosts**

8. **Create Ansible playbook to launch EC2**

```
(ansible-env) harsha@LAPTOP-DR0MMK6C:~$ cat > ~/launch_ec2.yml << 'EOF'
- name: Launch EC2 instance
  hosts: localhost
  connection: local
  gather_facts: false
  vars:
    ansible_python_interpreter: ~/ansible-env/bin/python3
    key_name: my-keypair
    instance_type: t2.micro
    image_id: ami-0f5ee92e2d63afc18  # Ubuntu 22.04 Mumbai
    region: ap-south-1

  tasks:
    - name: Launch EC2
      amazon.aws.ec2_instance:
        name: "ansible-demo"
        key_name: "{{ key_name }}"
        instance_type: "{{ instance_type }}"
        image_id: "{{ image_id }}"
        region: "{{ region }}"
        aws_profile: terraform
        wait: yes
      register: ec2

    - name: Show instance IP
      debug:
        msg: "Public IP: {{ ec2.instances[0].public_ip_address }}"
EOF
```

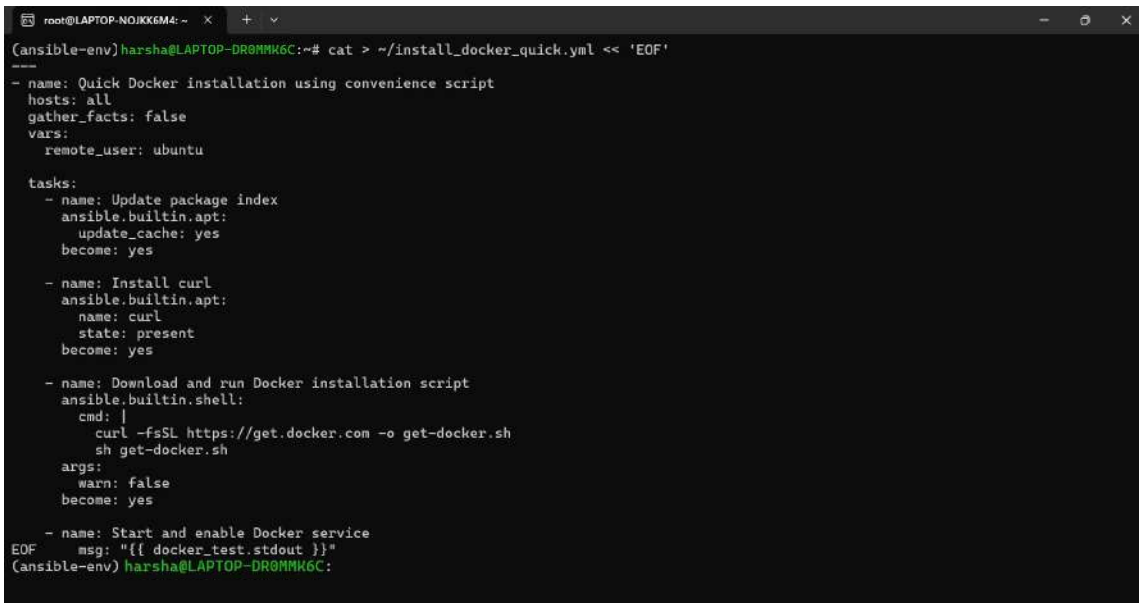
9. **Launch the EC2 instance**
   Run **ansible-playbook -i ~/hosts launch_ec2.yml**

```
(ansible-env) harsha@LAPTOP-DR0MMK6C:~$ ansible-playbook -i ~/hosts launch_ec2.yml

PLAY [Launch EC2 instance] ****************************************************************************************

TASK [Launch EC2] *************************************************************************************************
changed: [localhost]

TASK [Show instance IP] *******************************************************************************************
ok: [localhost] => {
    "msg": "Public IP: 13.201.9.51"
}

PLAY RECAP ********************************************************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

(ansible-env) harsha@LAPTOP-DR0MMK6C:~$
```

10. Ansible playbook to install Docker on your EC2 instance

```
---
- name: Quick Docker installation using convenience script
  hosts: all
  gather_facts: false
  vars:
    remote_user: ubuntu

  tasks:
    - name: Update package index
      ansible.builtin.apt:
        update_cache: yes
      become: yes

    - name: Install curl
      ansible.builtin.apt:
        name: curl
        state: present

      become: yes
    - name: Download and run Docker installation script
      ansible.builtin.shell:
        cmd: |
          curl -fsSL https://get.docker.com -o get-docker.sh
          sh get-docker.sh
      args:
        warn: false
      become: yes

    - name: Start and enable Docker service
      ansible.builtin.systemd:
        name: docker
        state: started
        enabled: yes
      become: yes
```

```
- name: Add user to docker group
    ansible.builtin.user:
        name: ubuntu
        groups: docker
        append: yes
    become: yes

- name: Test Docker installation
    ansible.builtin.command:
        cmd: docker run hello-world
    register: docker_test
    become: yes

- name: Show test result
    debug:
        msg: "{{ docker_test.stdout }}"
```



11. Run the Docker installation

Run **ansible-playbook -i ~/ec2_inventory_new.ini ~/install_docker_quick.yml**

```
(ansible-env)harsha@LAPTOP-DR0MMK6C:~# ansible-playbook -i ~/ec2_inventory_new.ini ~/install_docker_corrected.yml

PLAY [Install Docker on EC2 instance] ******************************************************************************************

TASK [Wait for SSH connection] *************************************************************************************************
[WARNING]: Host '3.111.214.107' is using the discovered Python interpreter at '/usr/bin/python3.10', but future installation of anoth
er Python interpreter could cause a different interpreter to be discovered. See https://docs.ansible.com/ansible-core/2.19/reference_
appendices/interpreter_discovery.html for more information.
ok: [3.111.214.107]

TASK [Update package cache] ****************************************************************************************************
ok: [3.111.214.107]

TASK [Install curl] ************************************************************************************************************
ok: [3.111.214.107]

TASK [Download Docker installation script] *************************************************************************************
changed: [3.111.214.107]

TASK [Install Docker] **********************************************************************************************************
changed: [3.111.214.107]

TASK [Start and enable Docker service] *****************************************************************************************
ok: [3.111.214.107]

TASK [Add ubuntu user to docker group] *****************************************************************************************
changed: [3.111.214.107]

TASK [Verify Docker installation] **********************************************************************************************
changed: [3.111.214.107]
```

12. Test Docker on the instance

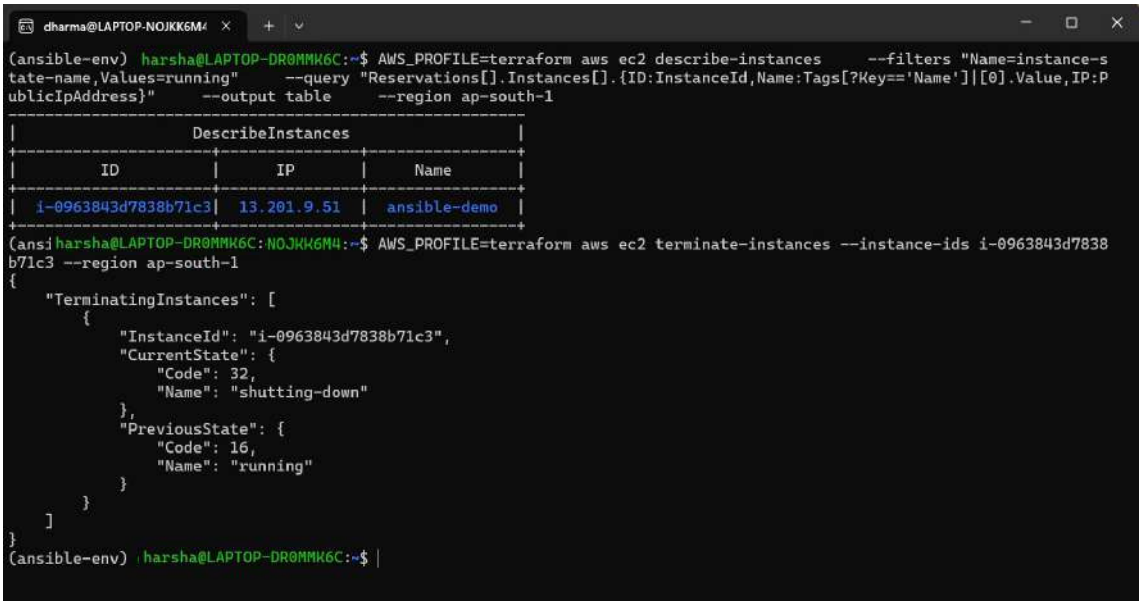
Run **ansible -i ~/ec2_inventory_new.ini all -m command -a "docker ps" -o**

ansible -i ~/ec2_inventory_new.ini all -m command -a "docker images" -o

```
(ansible-env)harsha@LAPTOP-DR0MMK6C:~# ansible -i ~/ec2_inventory_new.ini all -m command -a "docker ps" -o
        ansible -i ~/ec2_inventory_new.ini all -m command -a "docker images" -o
[WARNING]: Deprecation warnings can be disabled by setting `deprecation_warnings=False` in ansible.cfg.
[DEPRECATION WARNING]: The '-o' argument is deprecated. This feature will be removed from ansible-core version 2.23.
[DEPRECATION WARNING]: The oneline callback plugin is deprecated. This feature will be removed from ansible-core version 2.23.
3.111.214.107 | CHANGED | rc=0 | (stdout) CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
[WARNING]: Deprecation warnings can be disabled by setting `deprecation_warnings=False` in ansible.cfg.
[DEPRECATION WARNING]: The '-o' argument is deprecated. This feature will be removed from ansible-core version 2.23.
[DEPRECATION WARNING]: The oneline callback plugin is deprecated. This feature will be removed from ansible-core version 2.23.
3.111.214.107 | CHANGED | rc=0 | (stdout) REPOSITORY    TAG       IMAGE ID       CREATED        SIZE\nhello-world   latest    1b44b5a
3e06a   2 months ago   10.1kB
(ansible-env) harsha@LAPTOP-DR0MMK6C:
```

13. Terminate (destroy) the EC2 instance

Run **AWS_PROFILE=terraform aws ec2 terminate-instances --instance-ids <INSTANCE_ID> --region ap-south-1**



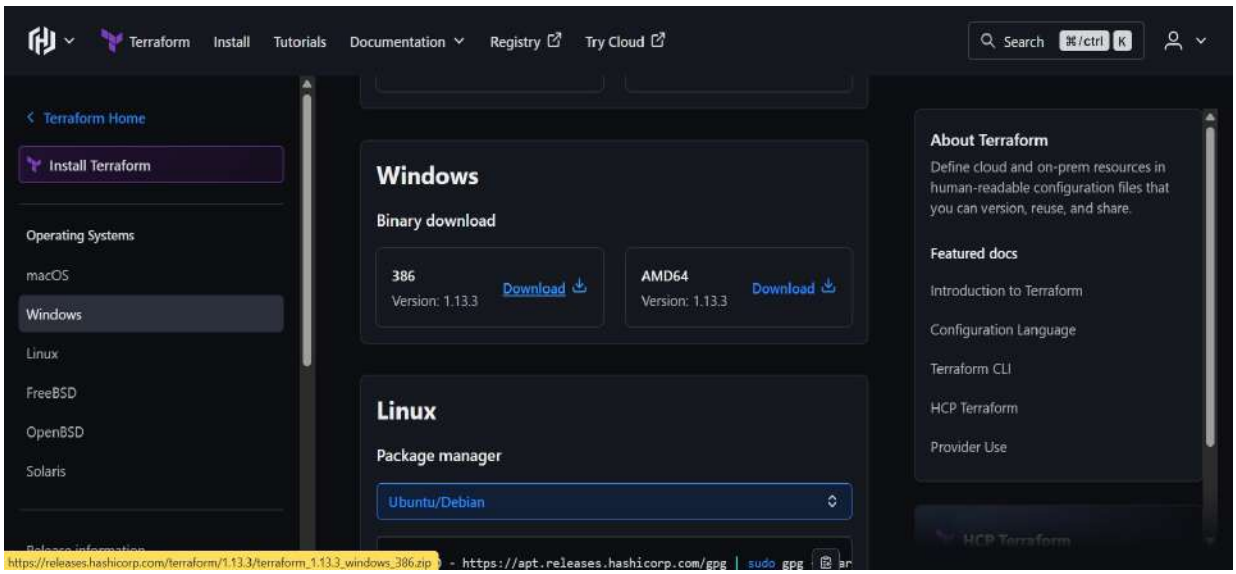
Experiment-12: Terraform – Infrastructure as Code

Aim: To demonstrate infrastructure provisioning using Terraform.

1. Install Terraform

1.1 Goto this website : https://developer.hashicorp.com/terraform/install

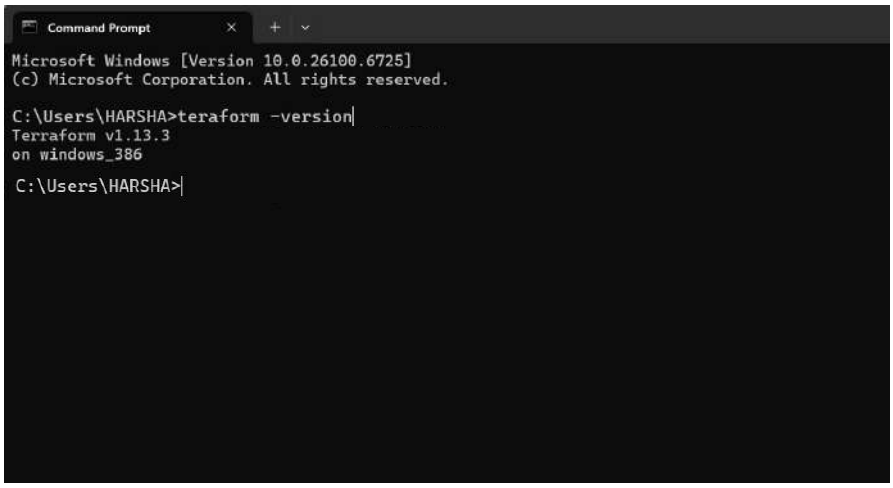
And download the zip for your OS/arch from HashiCorp releases.



1.2 Unzip and move the terraform (or terraform.exe) to a directory on your PATH and add the path to Environment variables in Windows.

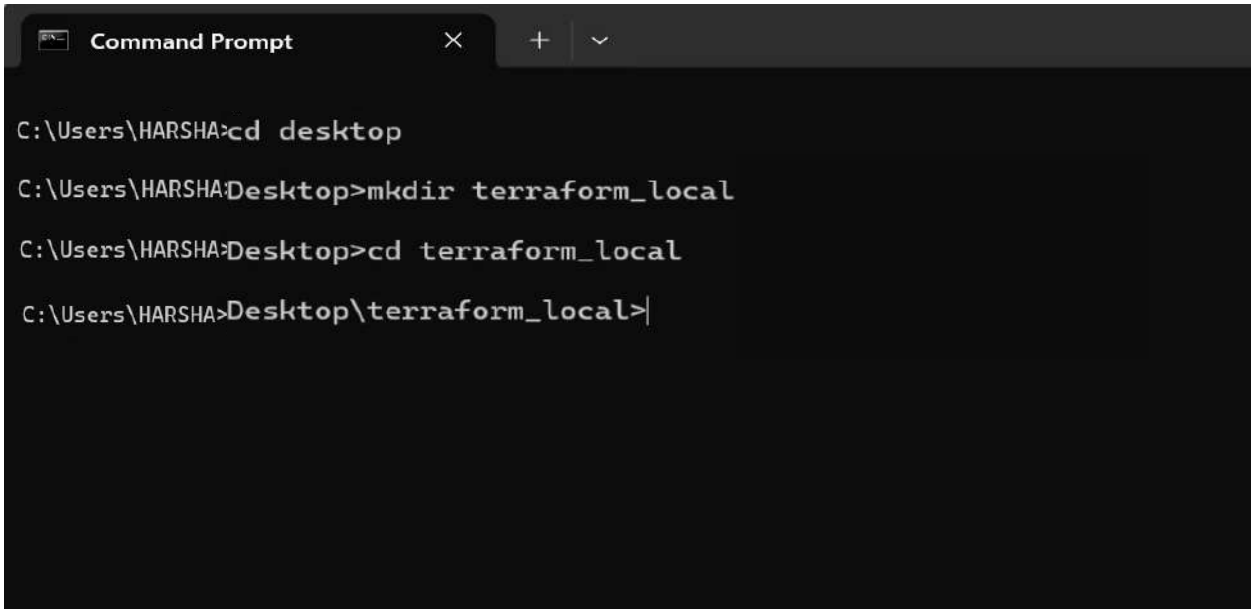
2. Verify Terraform

Run terraform -version in cmd

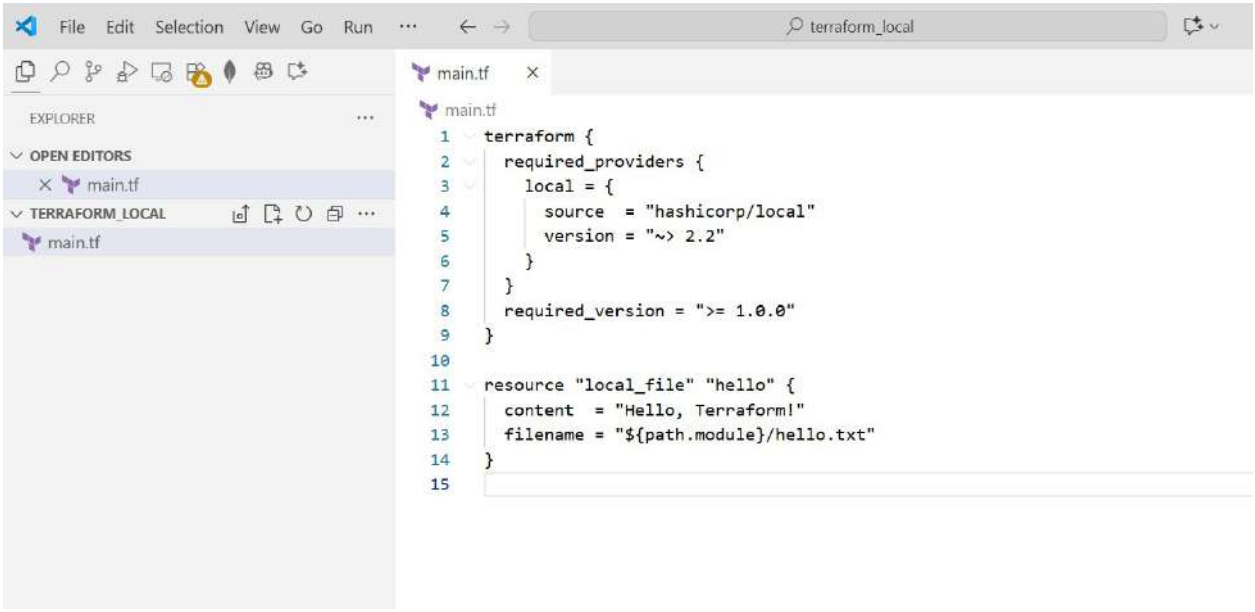


3. Running Terraform Locally

3.1 Create a project directory



3.2 Create a main.tf file in this folder



3.3 Run the standard Terraform workflow:

- terraform init
  - initialize - downloads provider plugins, creates .terraform/
- terraform validate
  - validate config
- terraform plan -out=tfplan
  - see what will changete rraform init

- terraform apply "tfplan"
  - apply the saved plan (no interactive prompt if you prefer: terraform apply -auto-approve)
- type hello.txt
  - verify file created and view its content
- terraform destroy -auto-approve
  - when done, destroy created resources

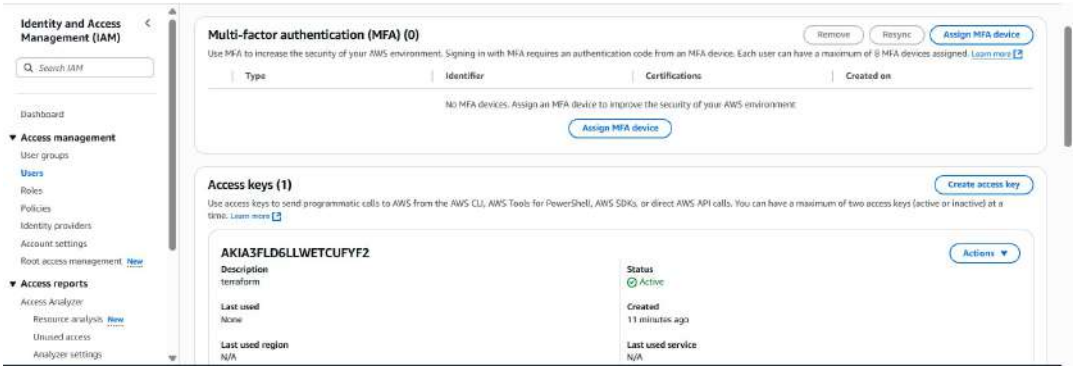
```
C:\Users\Desktop\terraform_local>terraform init  cal>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/local versions matching "~> 2.2"...
- Installing hashicorp/local v2.5.3...
- Installed hashicorp/local v2.5.3 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

C:\Users\HARSHA>\Desktop\terraform_local>terraform validate
Success! The configuration is valid.
```

```
No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.

Apply complete! Resources: 0 added, 0 changed, 0 destroyed.

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # local_file.hello will be destroyed
  - resource "local_file" "hello" {
      - content              = "Hello, Terraform!" -> null
      - content_base64sha256 = "ncWp9Vop6B6qYn8aIPKuPMIkWRzvO4mgxV9N1jscpf8=" -> null
      - content_base64sha512 = "PGmx7i5FR7nzPdG+RGISMuSUYKhuZDyPcbdZJm/HaGM81NLdWuoe3Uc5AALNtsDSIkdQWhjdnSYtfa64UdSk/w==" -> null
      - content_md5          = "08a67bbe3ffb550ec09d204035d4dd49" -> null
      - content_sha1         = "42086c02e03bf671ddf621ed9922f52f2c7a605c" -> null
      - content_sha256       = "9dc5a9f55a29eb4eaa627d1a20f2ae3cc224591cef3b89a0c55f4dd63b1ca5fd" -> null
      - content_sha512       = "3c69b1ee2e4547b9f33dd1be44621232e49460a86e643c8f71b759266fc768633cd4d2dd5aeaa1edd47390002cdb6c0d22247505a18dd9d262d7daeb851d4
a4ff" -> null
      - directory_permission = "0777" -> null
      - file_permission      = "0777" -> null
      - filename             = "./hello.txt" -> null
      - id                   = "42086c02e03bf671ddf621ed9922f52f2c7a605c" -> null
    }

Plan: 0 to add, 0 to change, 1 to destroy.
local_file.hello: Destroying... [id=42086c02e03bf671ddf621ed9922f52f2c7a605c]
local_file.hello: Destruction complete after 0s

Destroy complete! Resources: 1 destroyed.
```

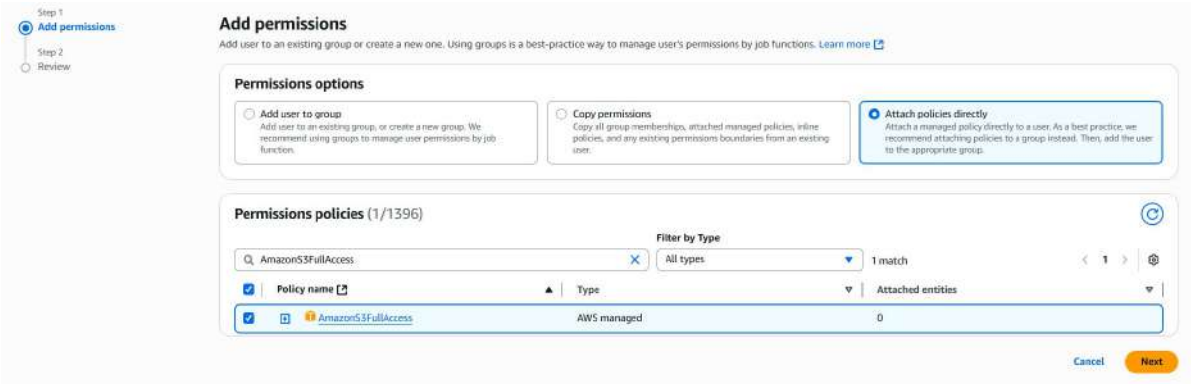
4. Creating Cloud Resources in AWS using Terraform

4.1 Install AWS CLI from the following link : https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html and set up AWS Credentails

4.2 In the IAM Console, create a User and Access key



4.3 Add S3 Permissions



4.4 Verify AWS CLI configuration

Run aws sts get-caller-identity --profile terraform in cmd to verify

4.5 Create Terraform configuration file

main.tf

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
  required_version = ">= 1.5.0"
}

provider "aws" {
  region  = "ap-south-1"
  profile = "terraform"
}

resource "aws_s3_bucket" "my_bucket" {
  bucket = "my-unique-terraform-bucket-12345"  # change this to a globally unique name
}

resource "aws_s3_bucket_acl" "my_bucket_acl" {
  bucket = aws_s3_bucket.my_bucket.id
  acl    = "private"
}
```

4.6 Initialize Terraform

```
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.100.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

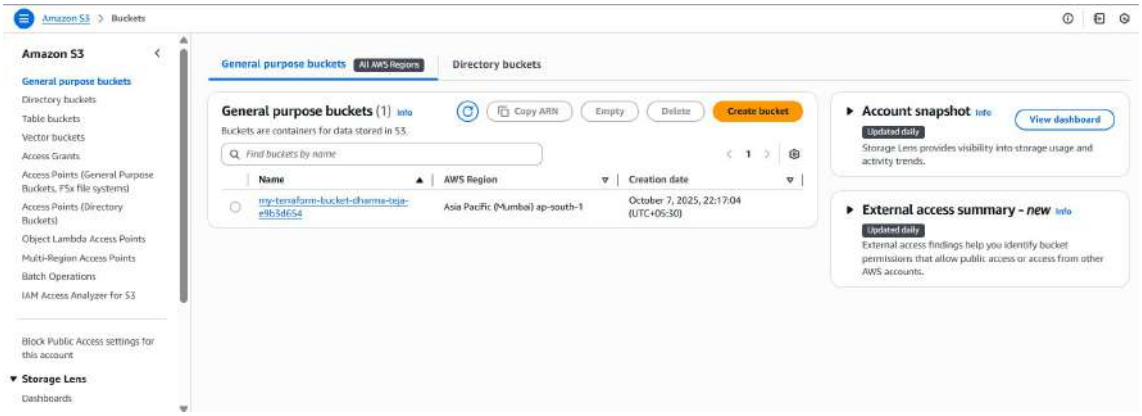
4.7 Plan your changes

```
Terraform used the selected providers to generate the following execution plan. Resource actions are
indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_s3_bucket.my_bucket will be created
  + resource "aws_s3_bucket" "my_bucket" {
      + acceleration_status         = (known after apply)
      + acl                         = (known after apply)
      + arn                         = (known after apply)
      + bucket                      = "my-unique-terraform-bucket-12345"
      + bucket_domain_name          = (known after apply)
      + bucket_prefix               = (known after apply)
      + bucket_regional_domain_name = (known after apply)
      + force_destroy               = false
      + hosted_zone_id              = (known after apply)
      + id                          = (known after apply)
      + object_lock_enabled         = (known after apply)
      + policy                      = (known after apply)
      + region                      = (known after apply)
      + request_payer               = (known after apply)
      + tags_all                    = (known after apply)
```

4.7 Apply the plan

```
You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
random_id.bucket_suffix: Refreshing state... [id=6bPWVA]
aws_s3_bucket.my_bucket: Refreshing state... [id=my-terraform-bucket-dharma-teja-e9b3d654]

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no
changes are needed.

Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
```

4.8 Verify the S3 Bucket in AWS



4.9 Clean up

```
    }

  # random_id.bucket_suffix will be destroyed
  - resource "random_id" "bucket_suffix" {
      - b64_std     = "6bPWVA==" -> null
      - b64_url     = "6bPWVA" -> null
      - byte_length = 4 -> null
      - dec         = "3920877140" -> null
      - hex         = "e9b3d654" -> null
      - id          = "6bPWVA" -> null
    }

Plan: 0 to add, 0 to change, 2 to destroy.

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_s3_bucket.my_bucket: Destroying... [id=my-terraform-bucket-dharma-teja-e9b3d654]
aws_s3_bucket.my_bucket: Destruction complete after 1s
random_id.bucket_suffix: Destroying... [id=6bPWVA]
random_id.bucket_suffix: Destruction complete after 0s

Destroy complete! Resources: 2 destroyed.
```