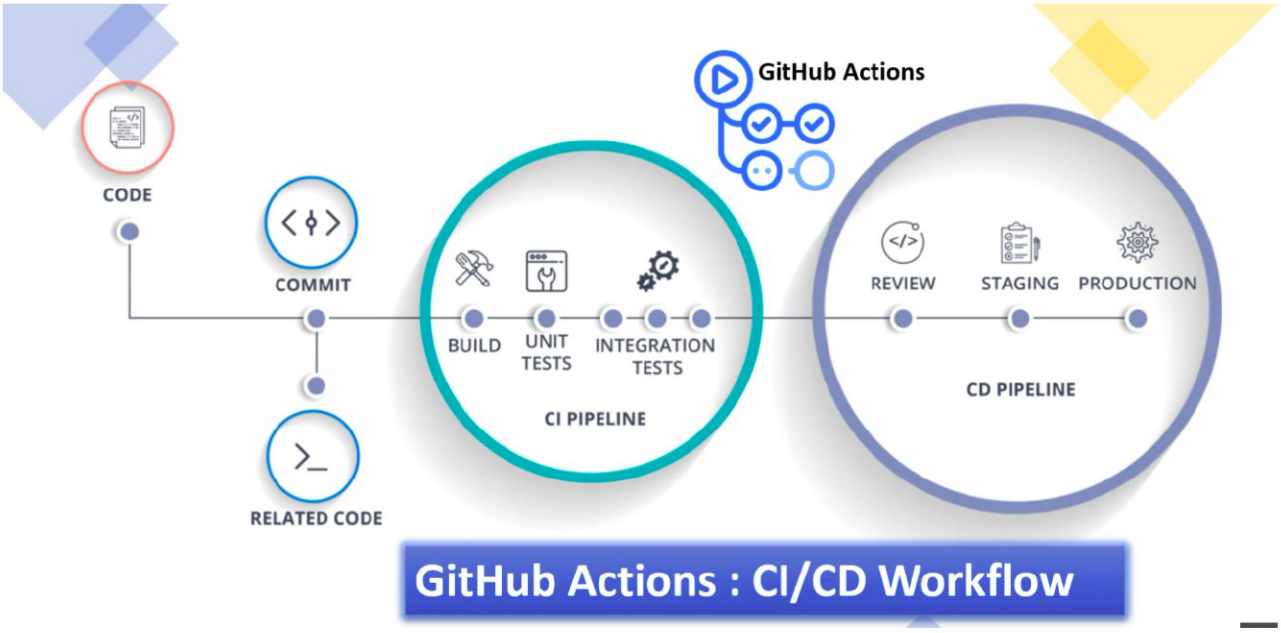
SpringBoot - Build CI/CD Pipeline Using GitHub Actions

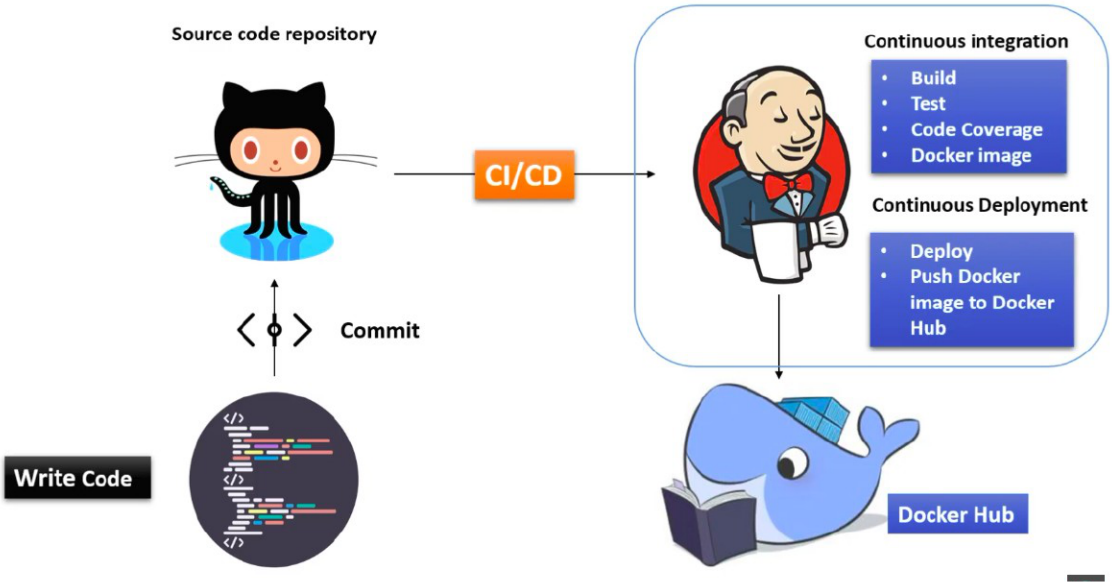
What is CI/CD?

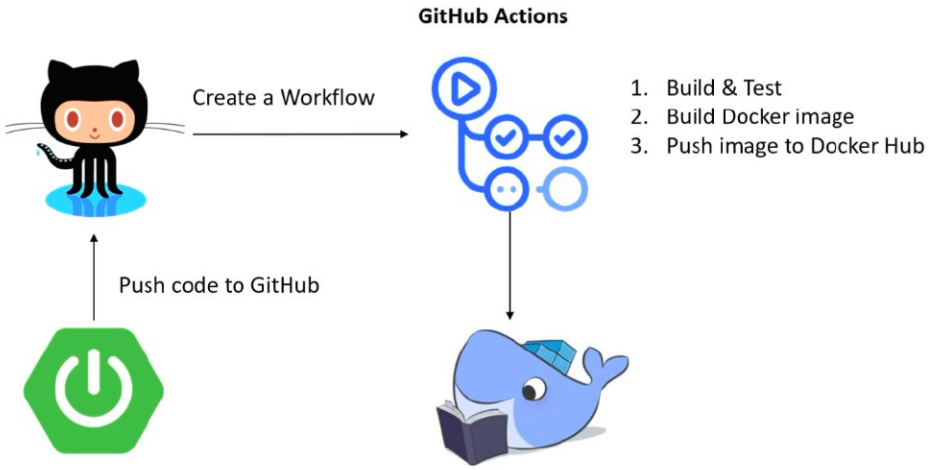
- **CI (Continuous Integration):** Automates building & testing code whenever changes are pushed to GitHub.
- **CD (Continuous Delivery/Deployment):** Automates packaging, creating Docker images, and deploying apps (to Docker Hub, Kubernetes, or Cloud).

We'll build a **Spring Boot project**, configure **GitHub Actions** for automation, and push a **Docker image** to Docker Hub.



Jenkins as CI/CD





Step 1: First create a Spring boot sample project

- Use **Spring Initializr** or **IntelliJ** to generate a simple Spring Boot application.
- Add dependencies like `Spring Web` (or others as required).
- Test locally to ensure the project runs.

Step 2: Push the spring project on Github.

Initialize Git repository:

```
git init
```

- Add files:

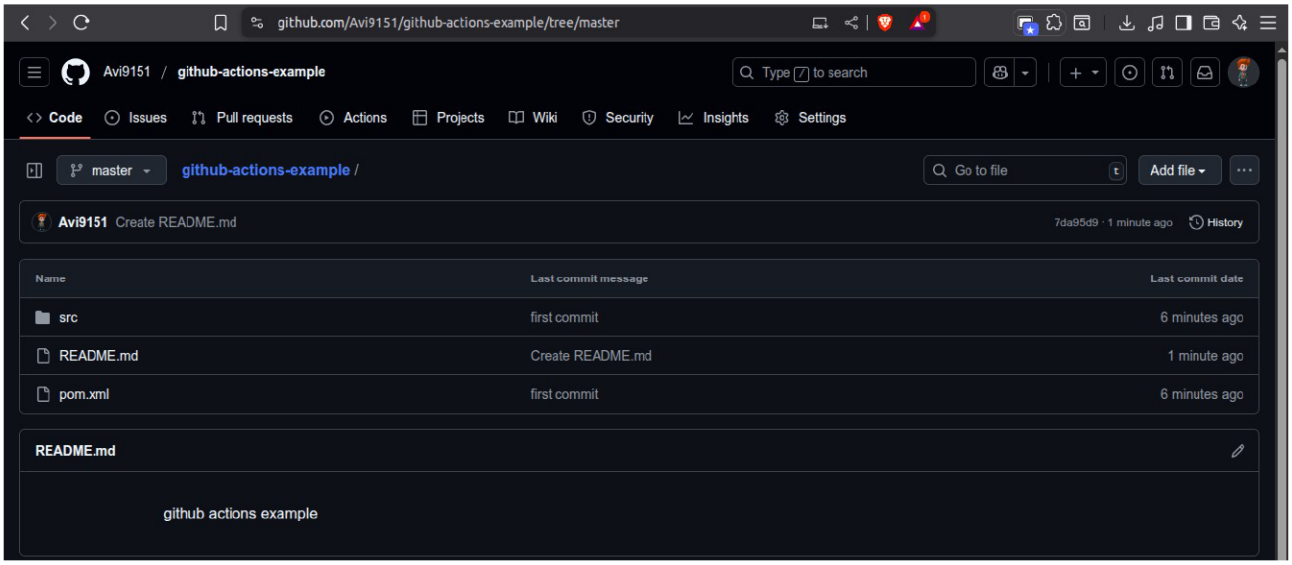
  ```
  git add .
  ```

- Commit:

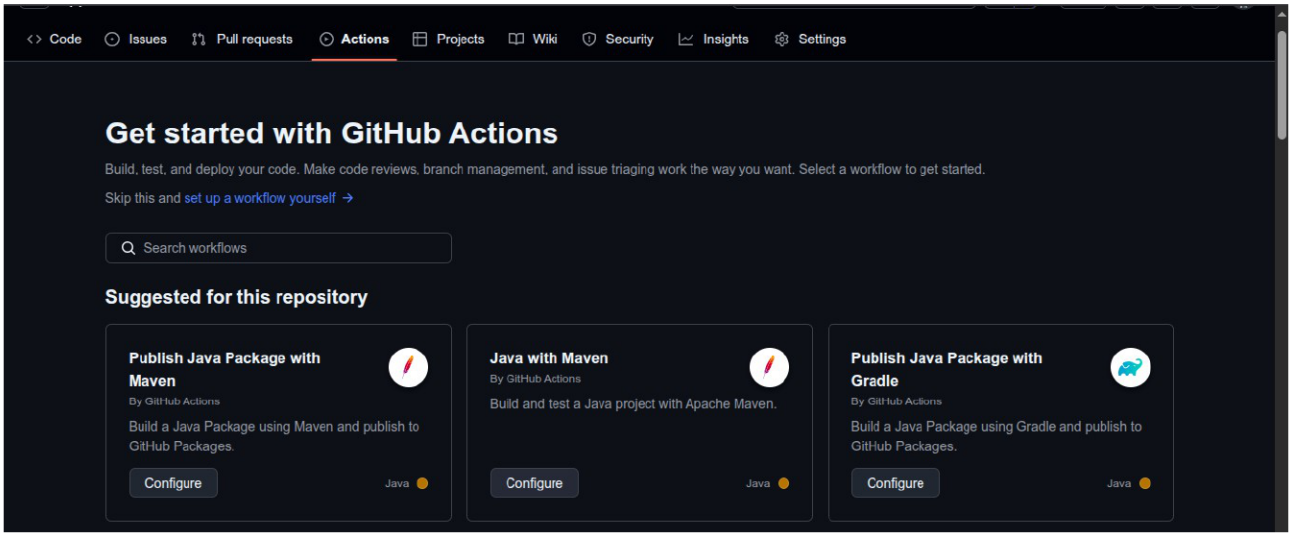
  ```
  git commit -m "Initial commit"
  ```

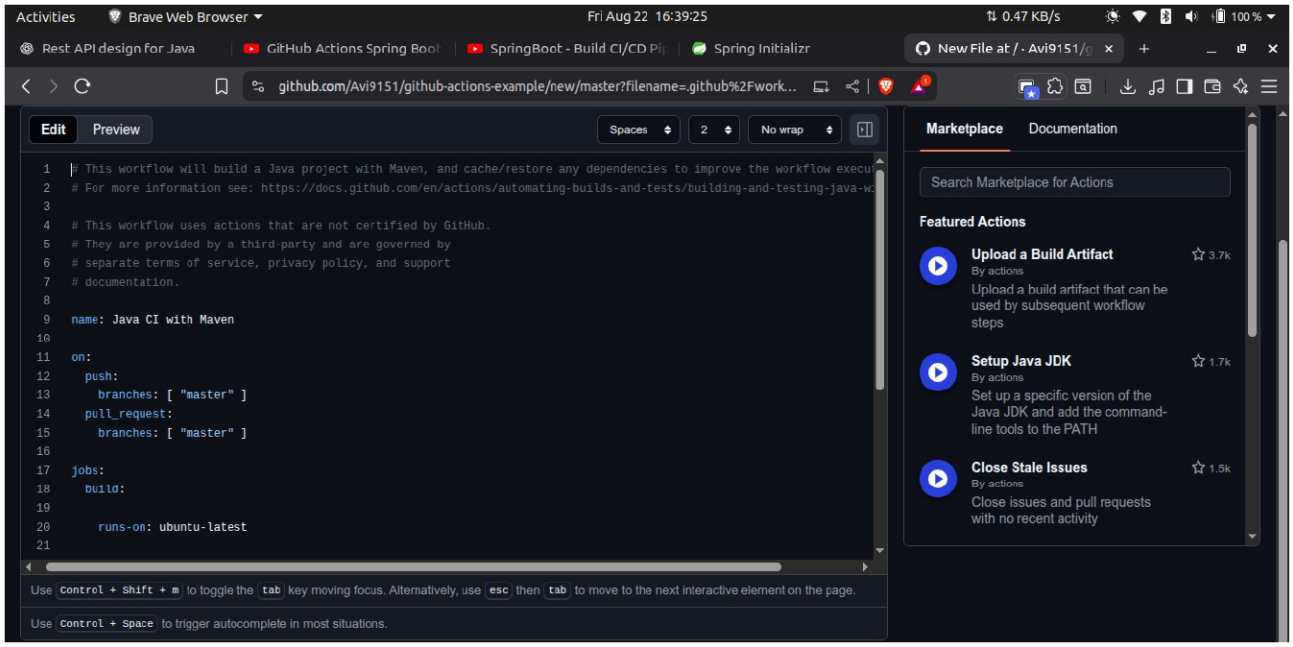
- Push to GitHub repository:

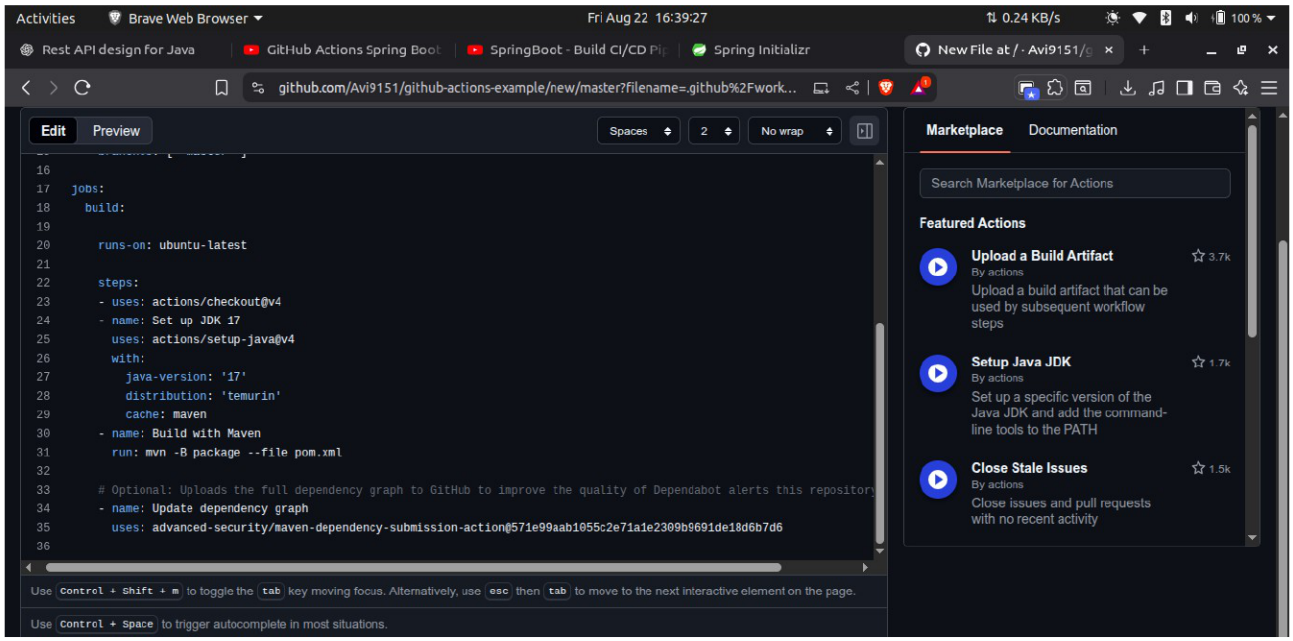
  ```
  git remote add origin <repo-url>
  git push -u origin master
  ```



Step 3: Go to the **Actions** -> **Java with Maven** then click on **Configure**







Step 4: Rename the **name** and **run**

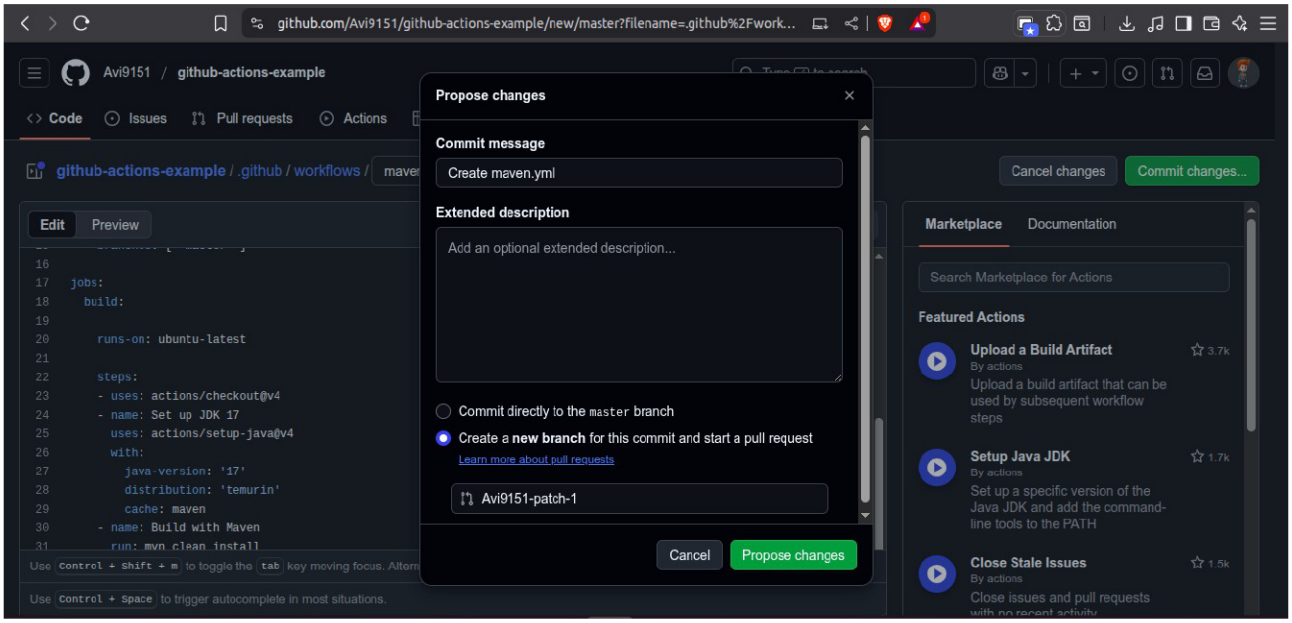
name: project Cicd flow

run: mvn clean install

Step 5: Click on the **Commit changes** button.

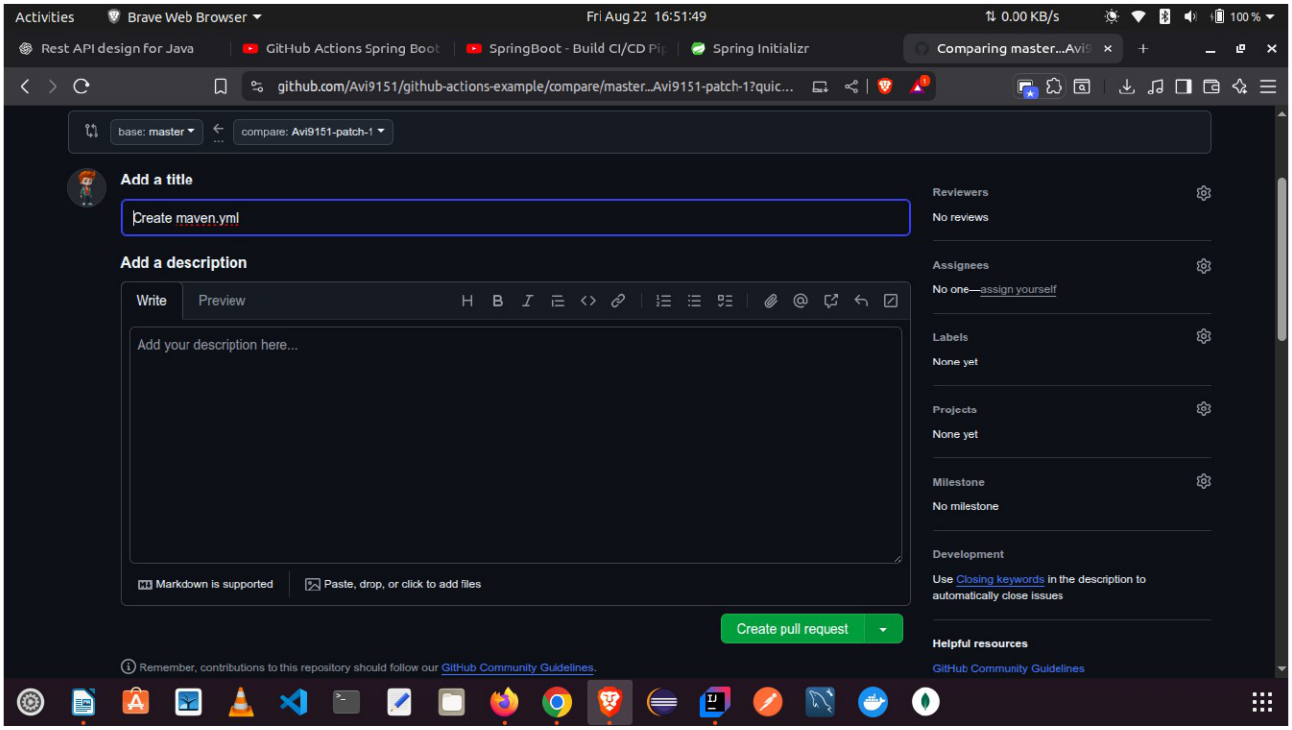
Save changes by clicking **Commit changes**.

- GitHub will suggest creating a new branch → allow it.

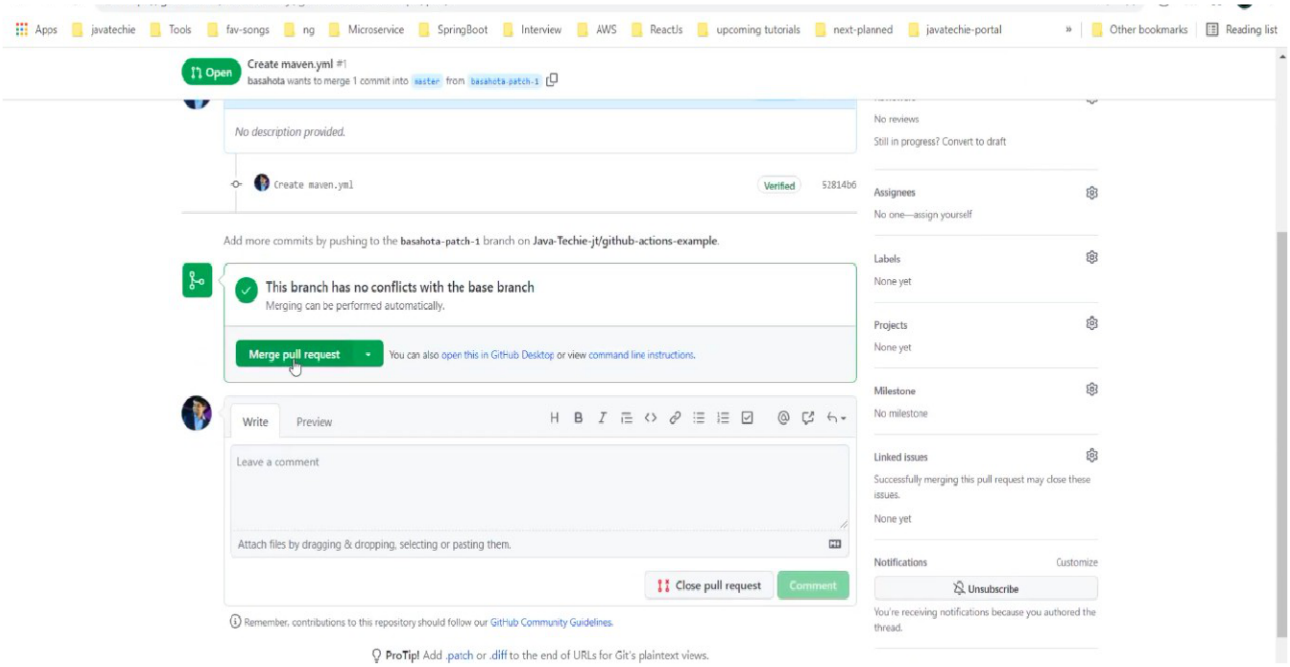


Step 6: Click on **Create a new branch for this commit and start a pull request** -> click on **Propose changes** button.

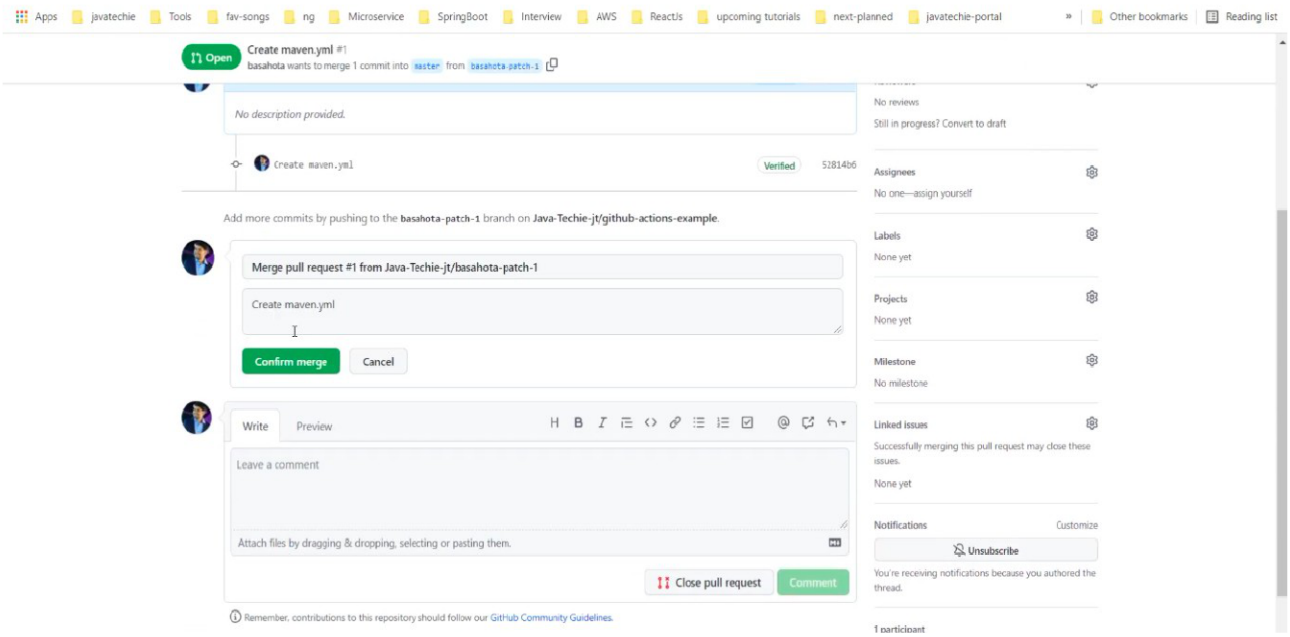
Step 7: Click on the **Create pull request** to merge these changes.



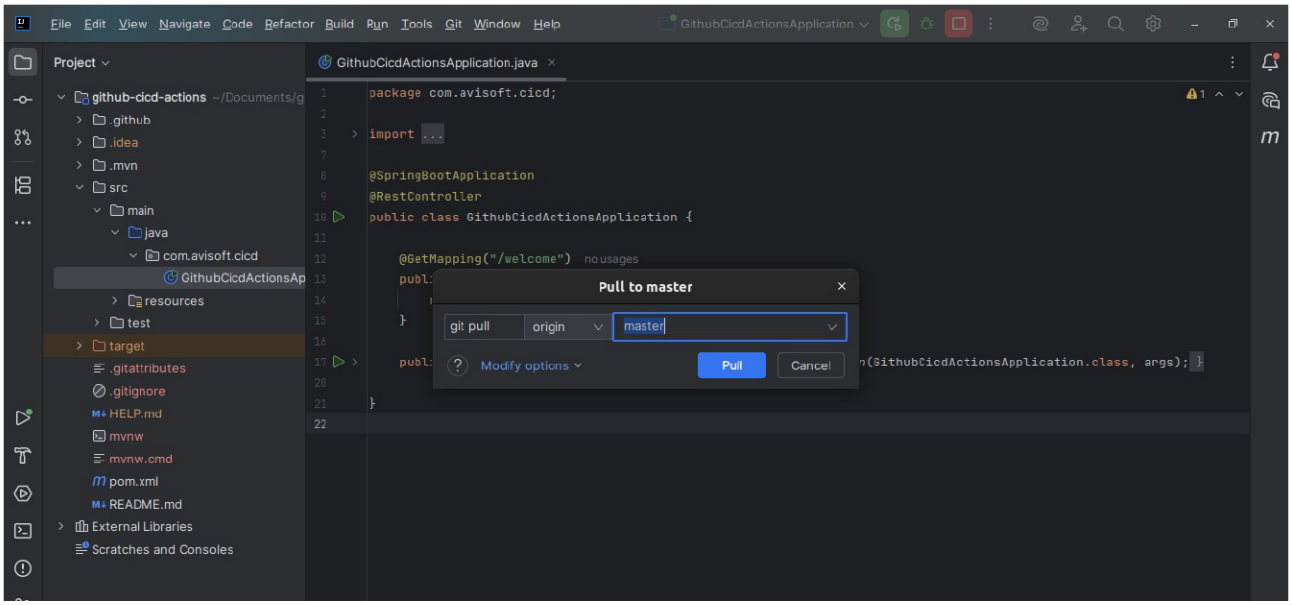
Step 8: Click on **Merge pull request**



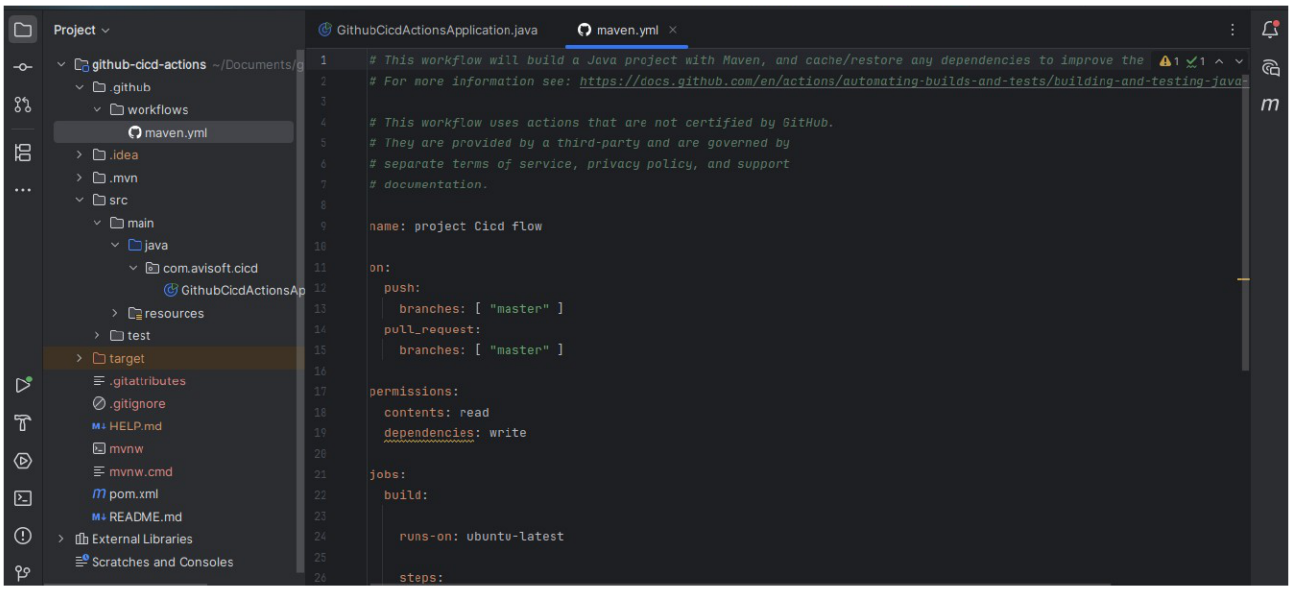
Then Click on the **Confirm merge** button.



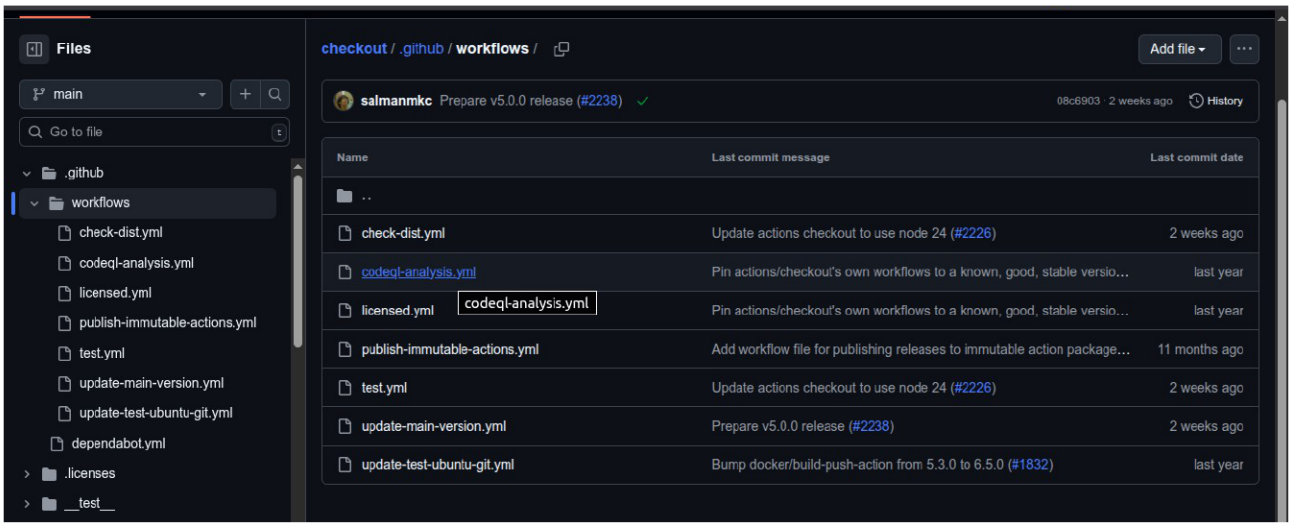
Step 9: Then Open **IntelliJ → git pull →** You'll see **.github/workflows/maven.yml** file added.



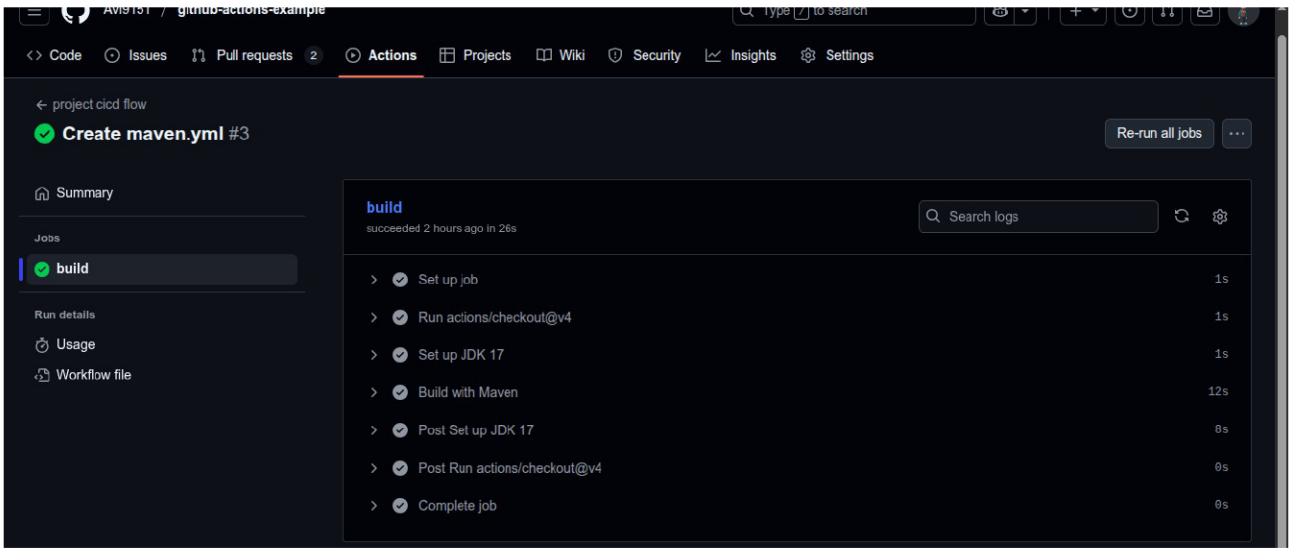
After pull open the **Github folder** -> **workflows** -> **maven.yml**

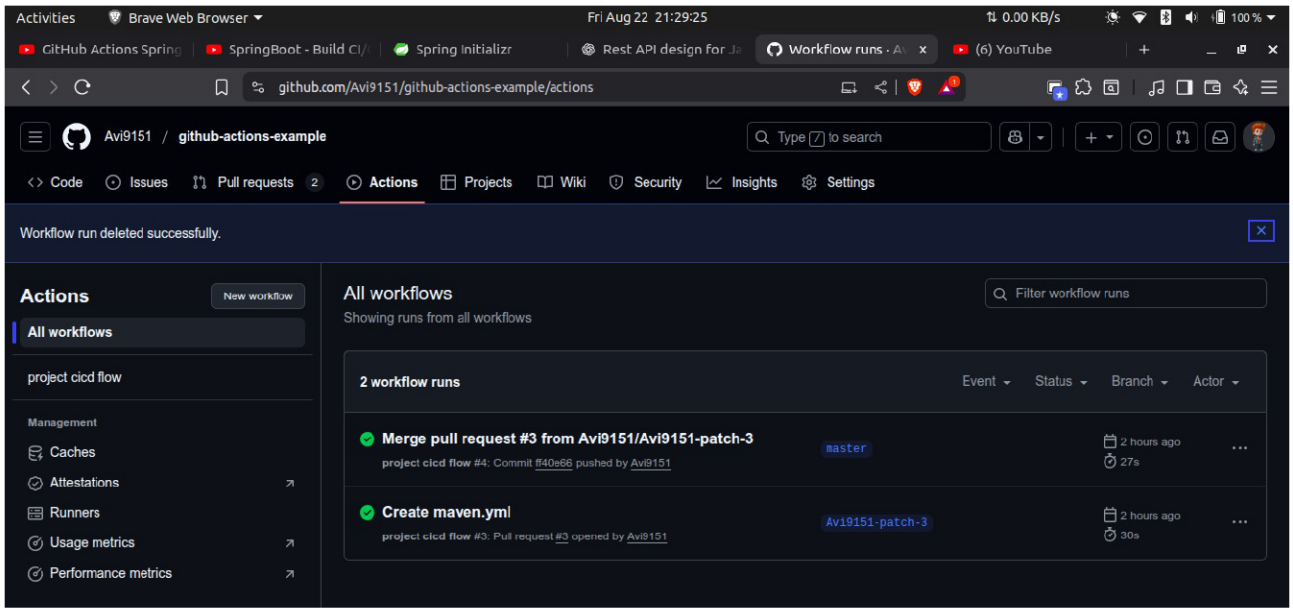


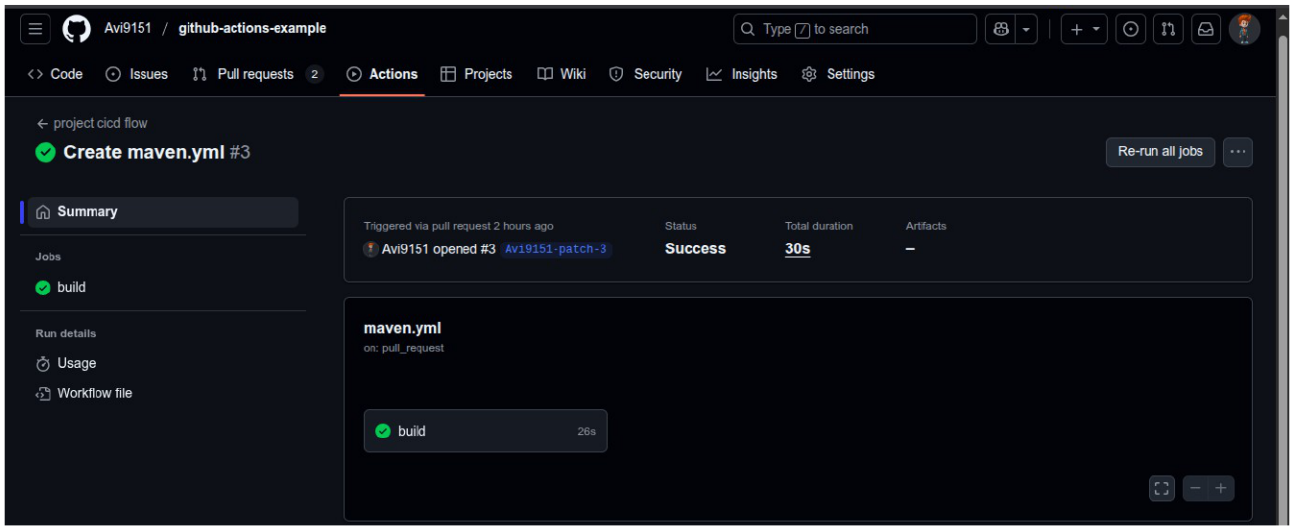
Step 10: Lets check the actions/checkouts so open url- https://github.com/actions and search.



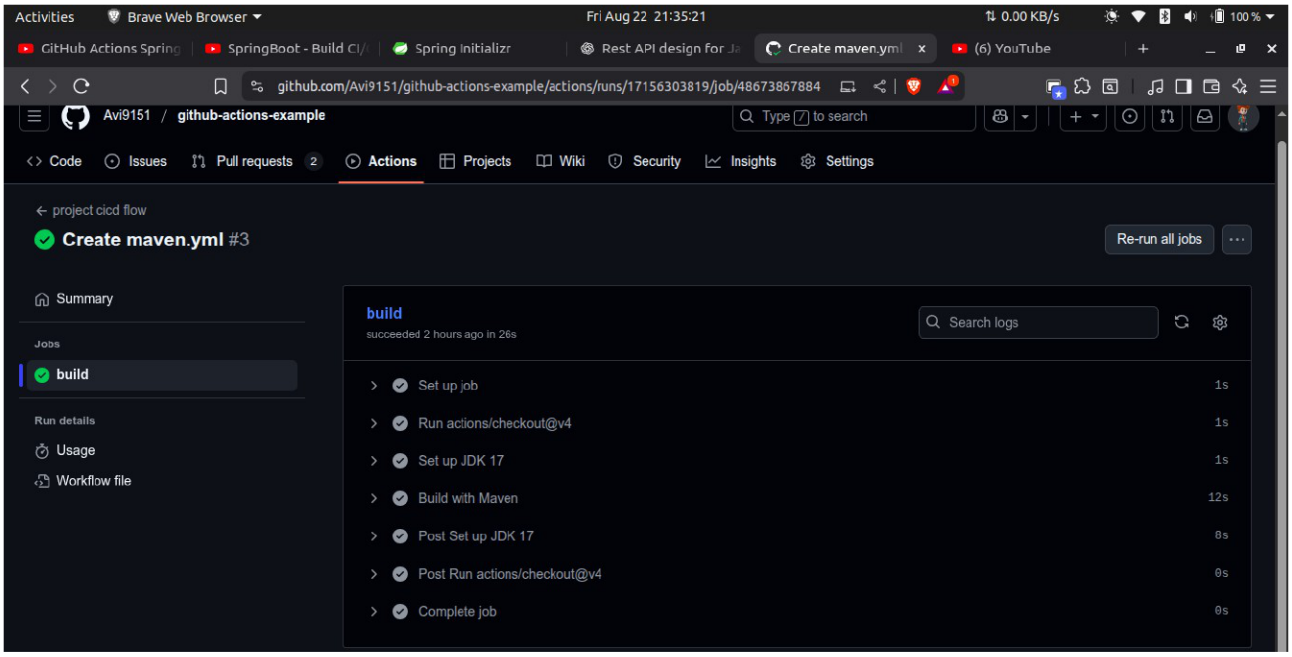
Step 11: Go to the **Actions** and click on the Create **maven.yml**



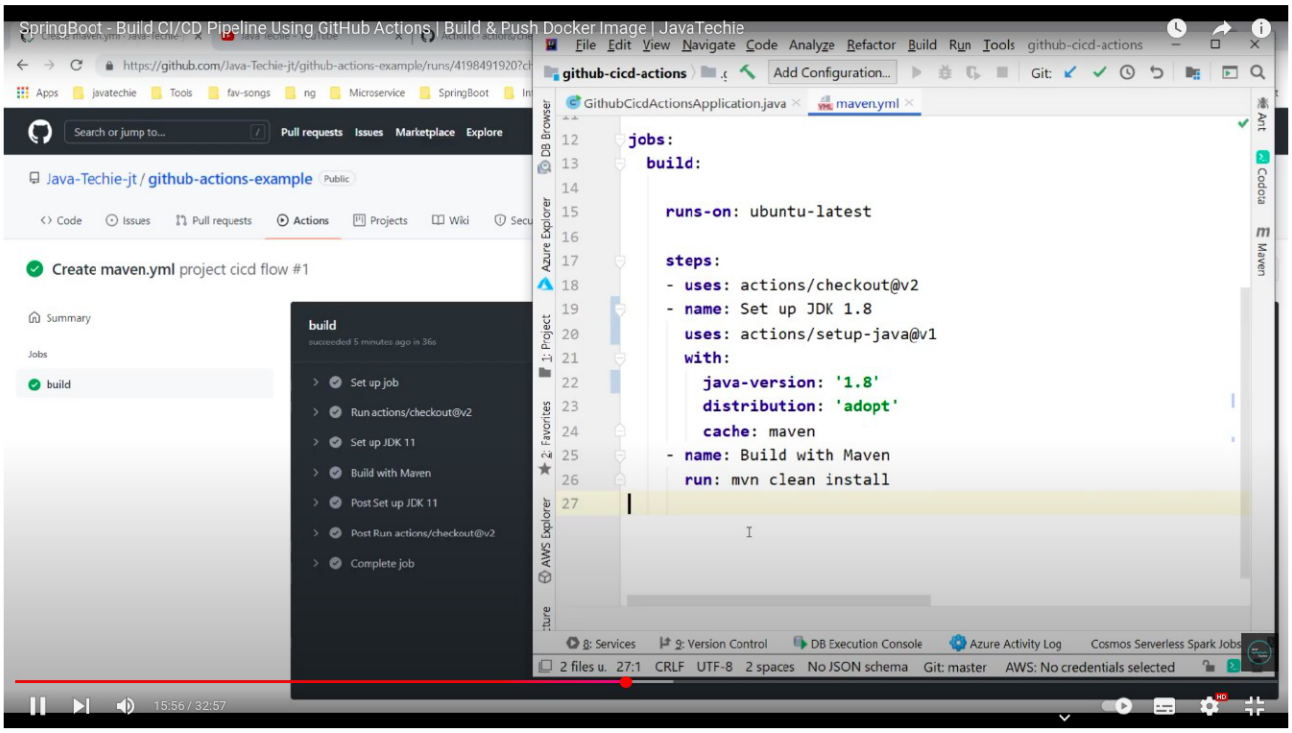


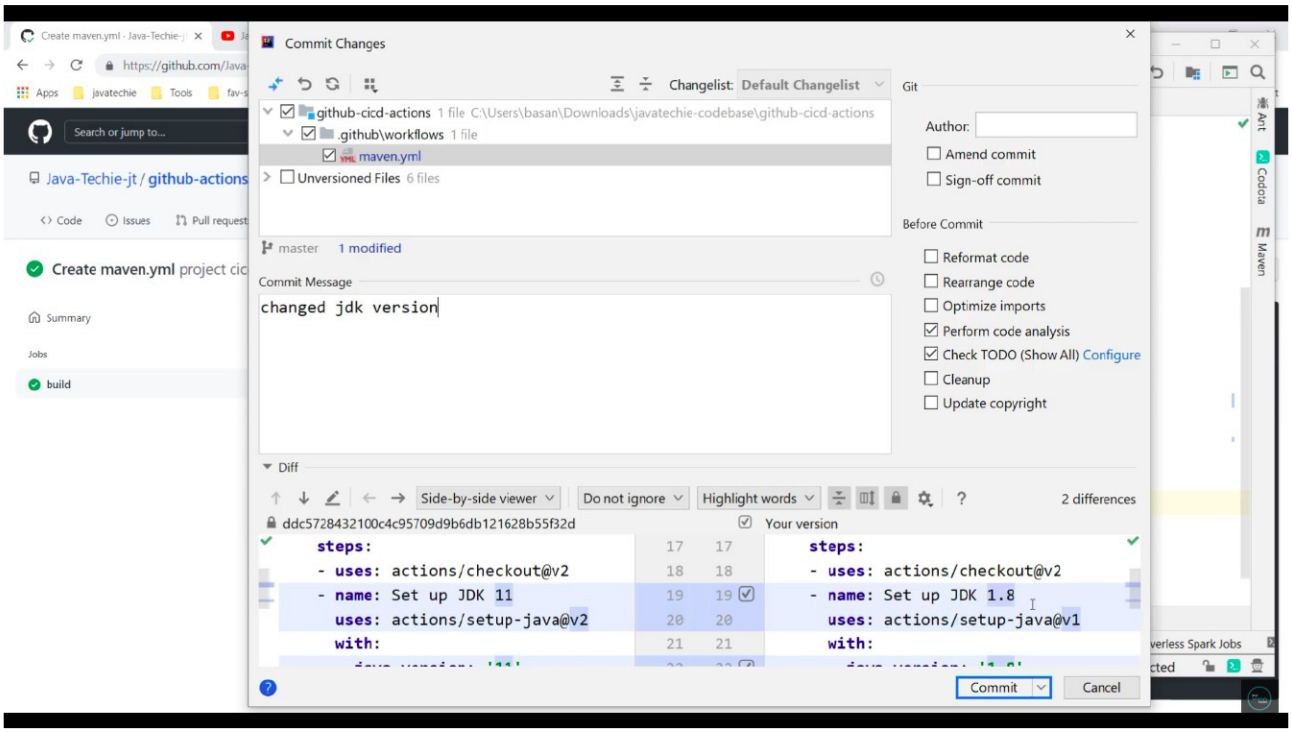


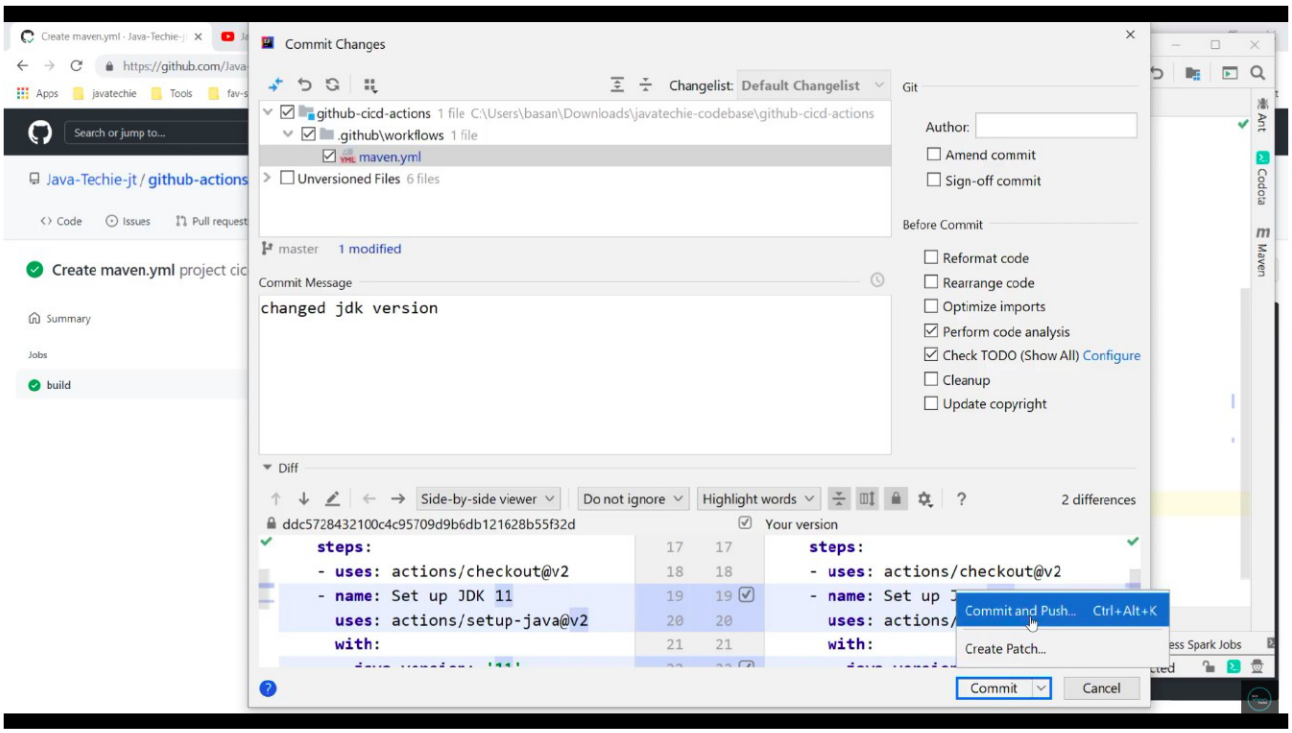
Click on **build**

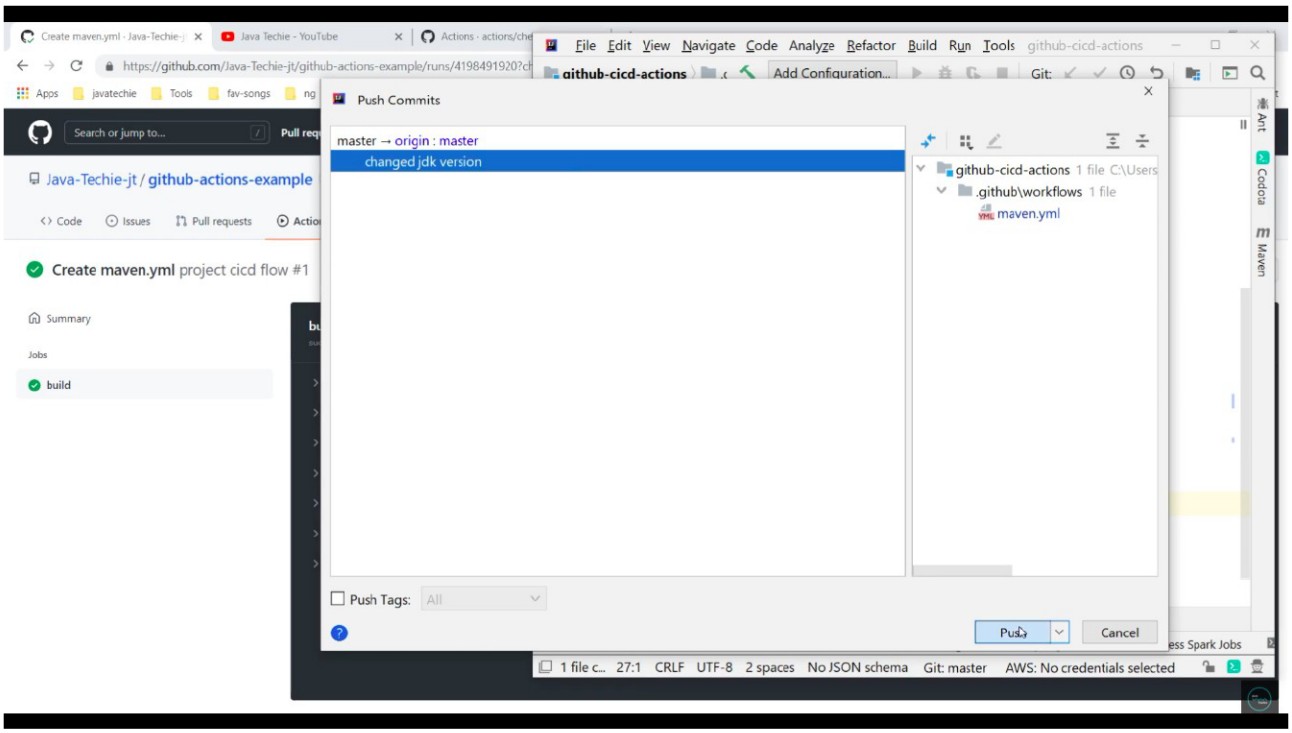

Activities
Brave Web Browser
Fri Aug 22 21:35:21
0.00 KB/s
100 %
GitHub Actions Spring
SpringBoot - Build CI/
Spring Initializr
Rest API design for Ja
Create maven.yml
(6) YouTube
github.com/Avi9151/github-actions-example/actions/runs/17156303819/job/48673867884
Avi9151 / github-actions-example
Type / to search
Code
Issues
Pull requests 2
Actions
Projects
Wiki
Security
Insights
Settings
project cicd flow
Create maven.yml #3
Re-run all jobs
Summary
Jobs
build
Run details
Usage
Workflow file
build
succeeded 2 hours ago in 26s
Search logs
Set up job 1s
Run actions/checkout@v4 1s
Set up JDK 17 1s
Build with Maven 12s
Post Set up JDK 17 8s
Post Run actions/checkout@v4 0s
Complete job 0s

Note : If you want to change jdk version and checkout@v(version).

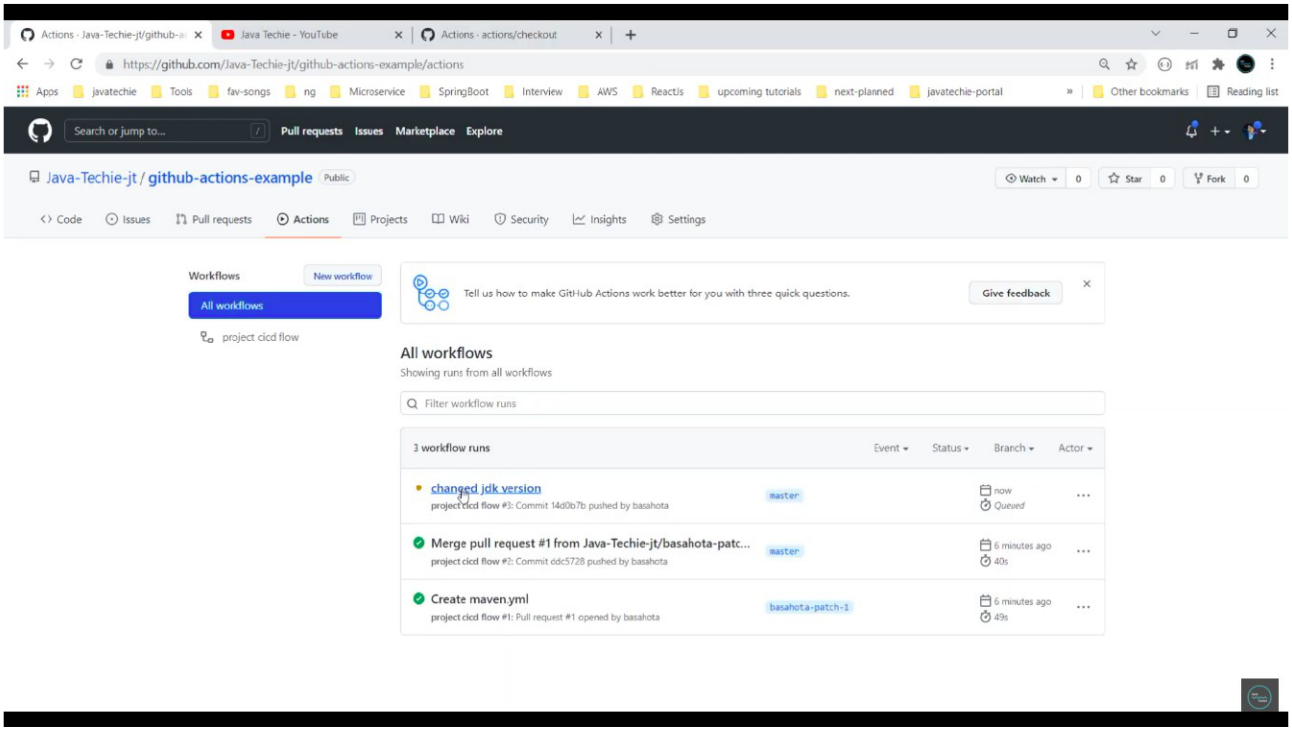




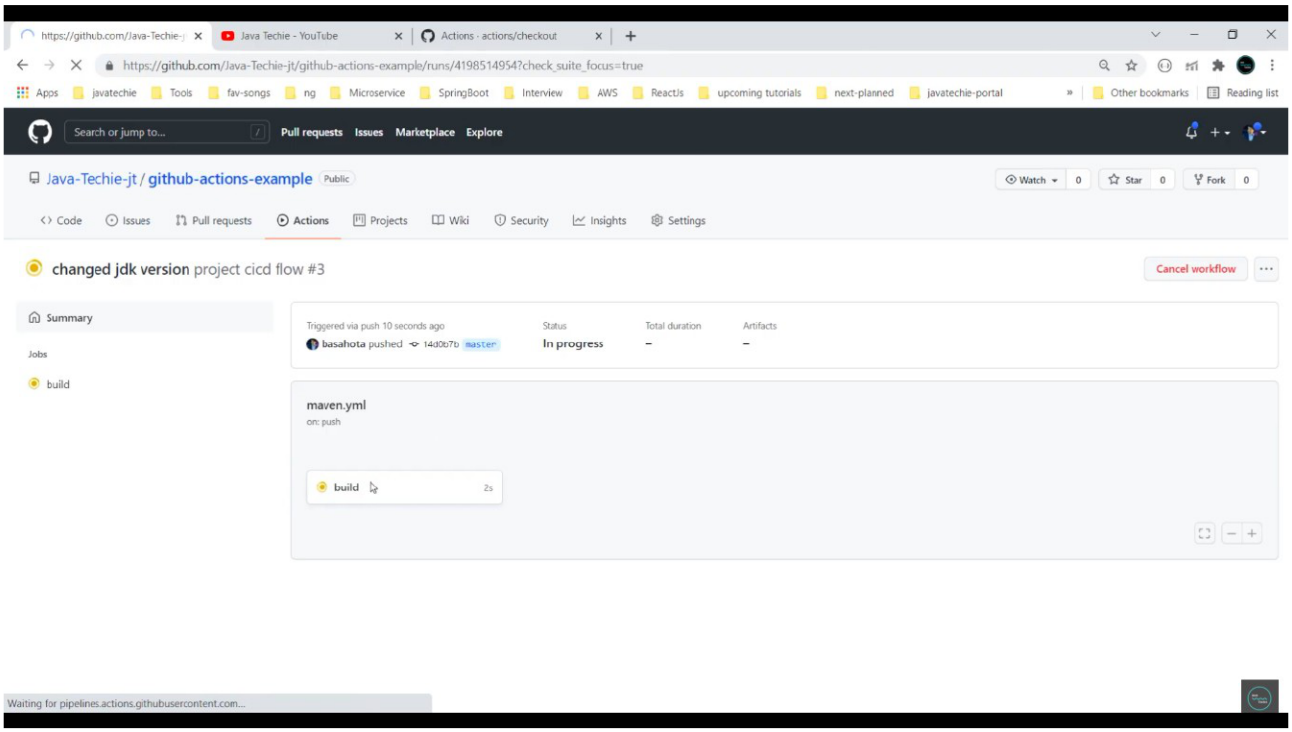

Commit Changes
Changelist: Default Changelist
github-cicd-actions 1 file C:\Users\basan\Downloads\javatechie-codebase\github-cicd-actions
.github\workflows 1 file
maven.yml
Unversioned Files 6 files
master 1 modified
Commit Message
changed jdk version
Git
Author:
Amend commit
Sign-off commit
Before Commit
Reformat code
Rearrange code
Optimize imports
Perform code analysis
Check TODO (Show All) Configure
Cleanup
Update copyright
Diff
Side-by-side viewer
Do not ignore
Highlight words
2 differences
ddc5728432100c4c95709d9b6db121628b55f32d
Your version
steps:
- uses: actions/checkout@v2
- name: Set up JDK 11
uses: actions/setup-java@v2
with:
Commit and Push... Ctrl+Alt+K
Create Patch...
Commit
Cancel
Java-Techie-jt / github-actions
Create maven.yml project cic
Summary
Jobs
build


Push Commits
master → origin : master
changed jdk version
github-cicd-actions 1 file C:\Users
.github\workflows 1 file
maven.yml
Push Tags: All
Push
Cancel
File Edit View Navigate Code Analyze Refactor Build Run Tools github-cicd-actions
Add Configuration...
Git:
Java-Techie-jt / github-actions-example
Create maven.yml project cicd flow #1
Summary
Jobs
build
1 file c... 27:1 CRLF UTF-8 2 spaces No JSON schema Git: master AWS: No credentials selected

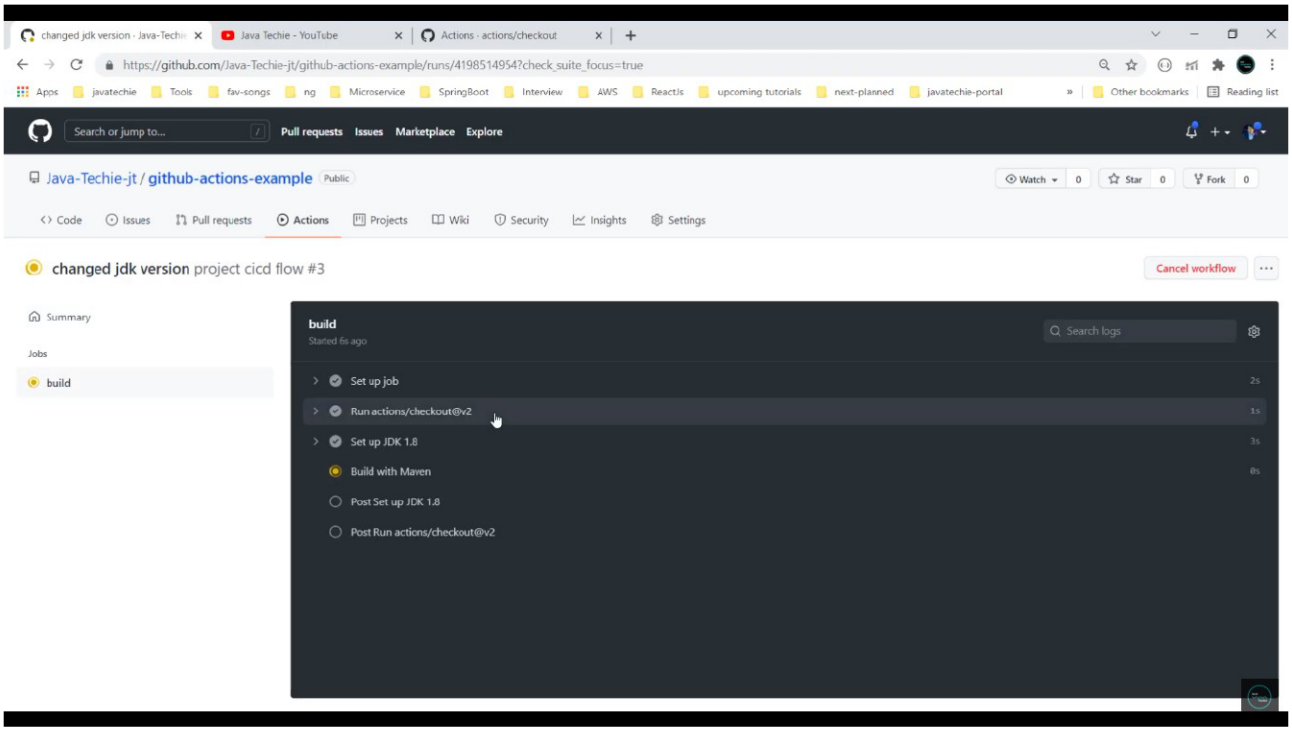
Click on **changed jdk version**



Click on **build**



You can see your all jdk version change, etc(if you want to change) it is doing setup.

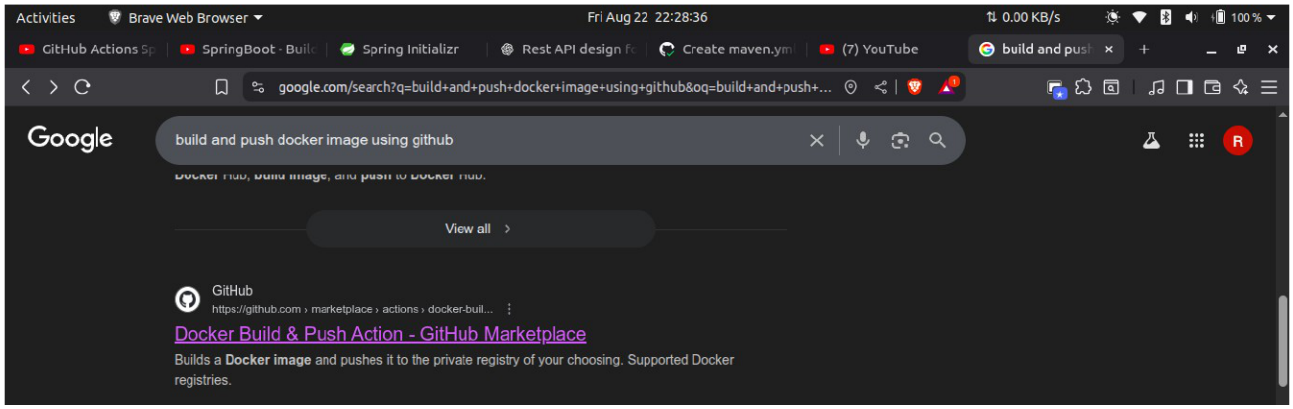


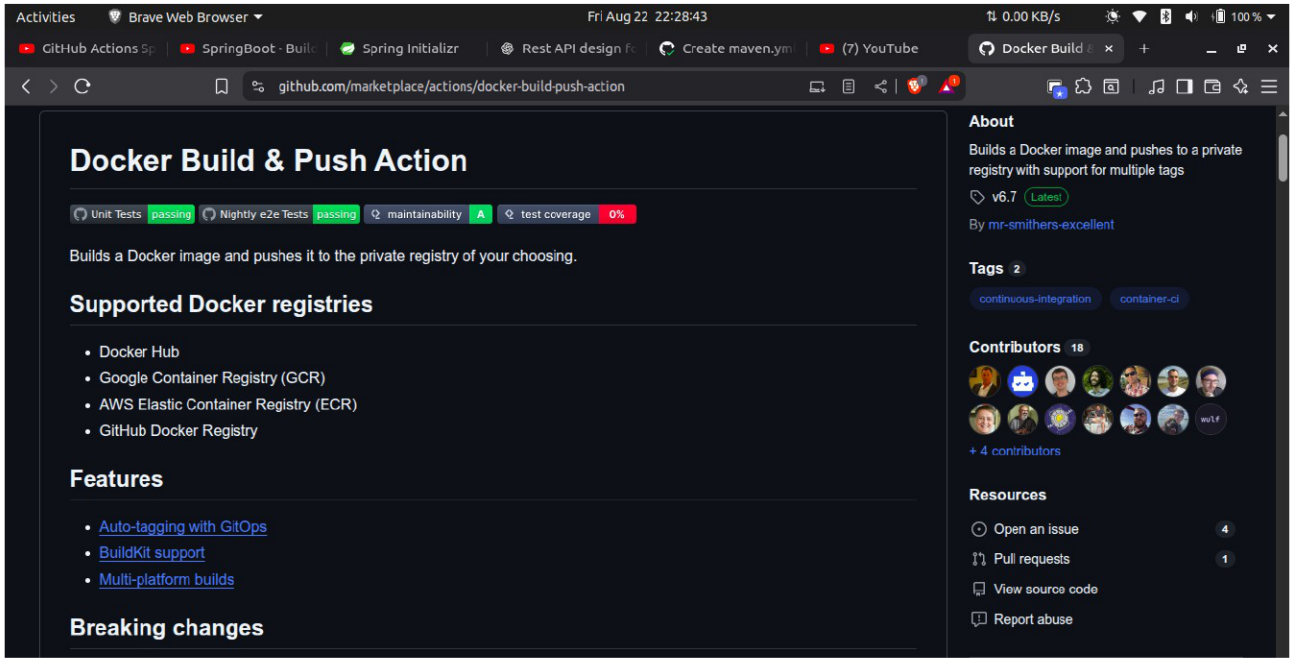
Now we tell to the Actions to build docker image

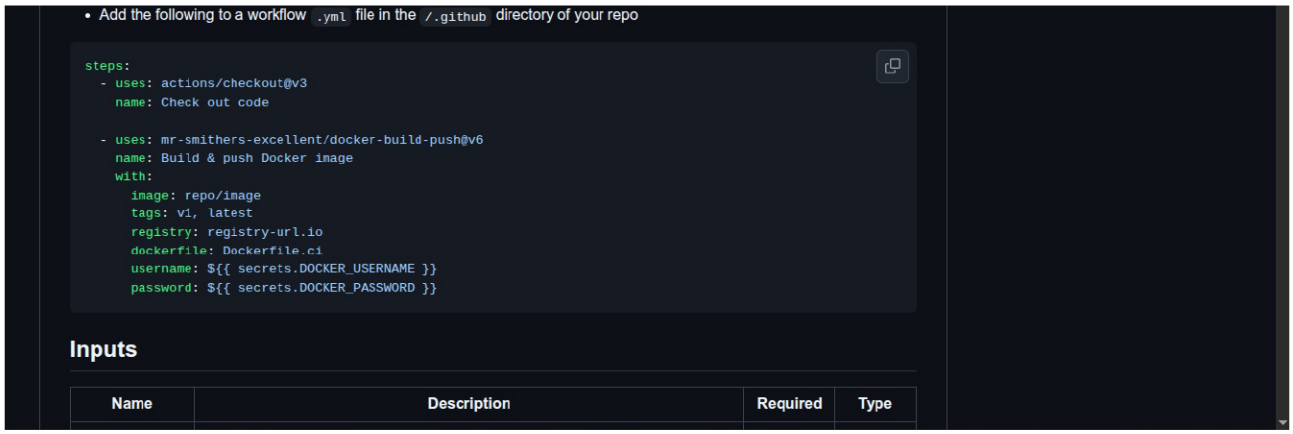
1. Login to your docker hub
2. Build a docker image
3. Push that image to docker hub

Note : “Github provide readymade Actions which can create docker image and push automatically without manually configuration.”

Step 12: Now search this - “build and push docker image using github” on browser.

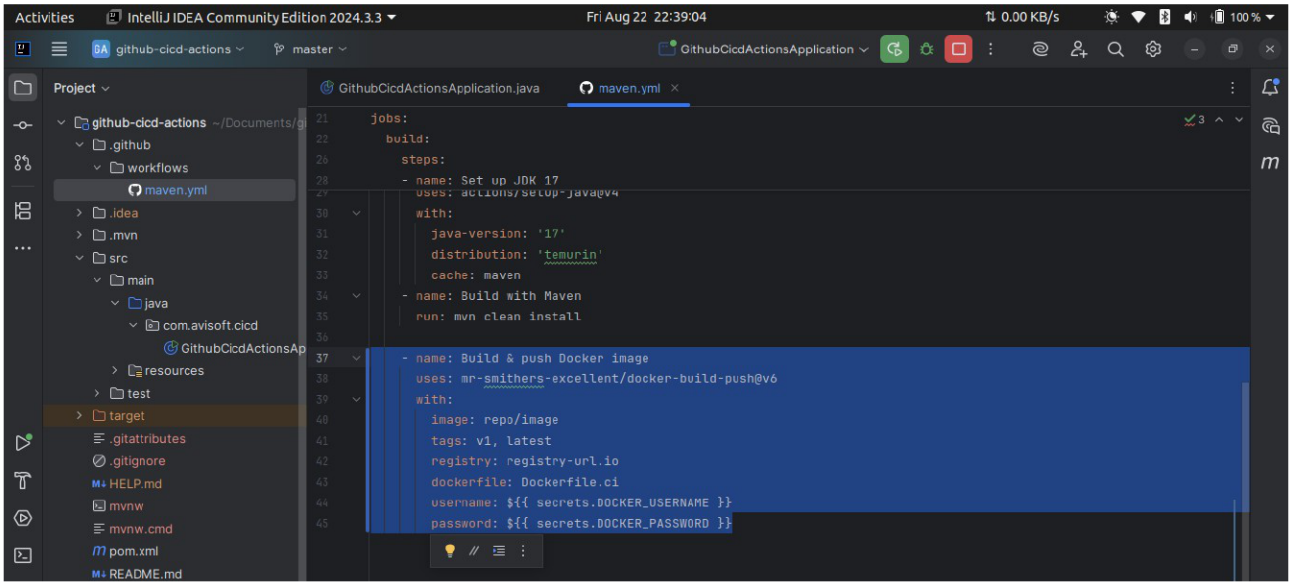






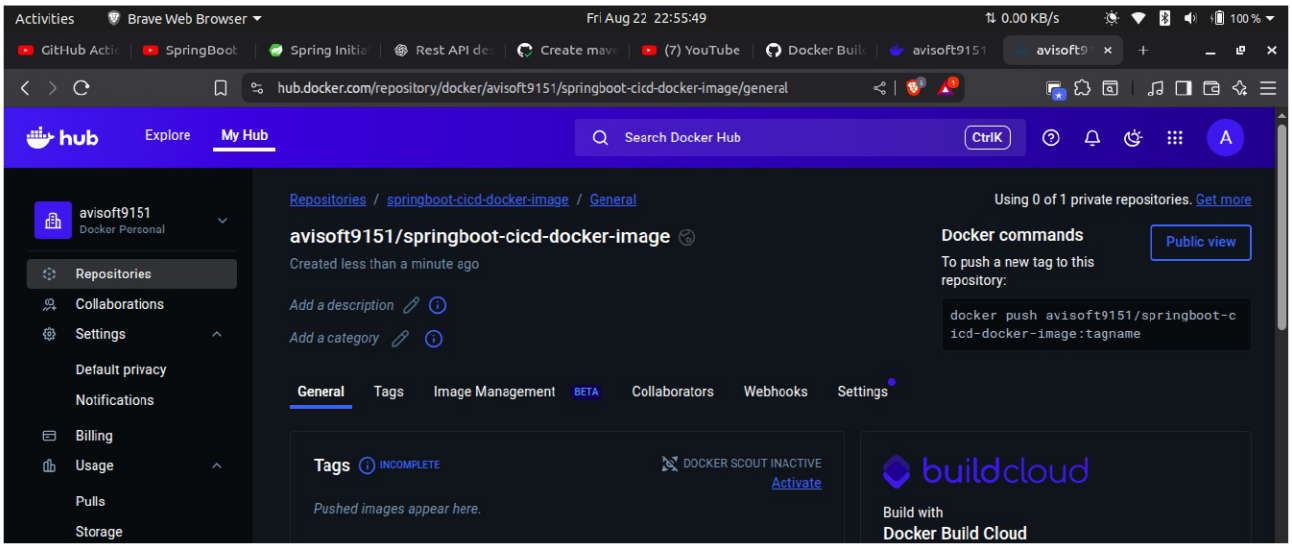
Step 13: Add this yml code in maven.yml.

```
- name: Build & push Docker image
  uses: mr-smithers-excellent/docker-build-push@v6
  with:
    image: repo/image
    tags: v1, latest
    registry: registry-url.io
    dockerfile: Dockerfile
    username: ${{ secrets.DOCKER_USERNAME }}
    password: ${{ secrets.DOCKER_PASSWORD }}
```



Step 14: Open **Docker hub** on browser and login.

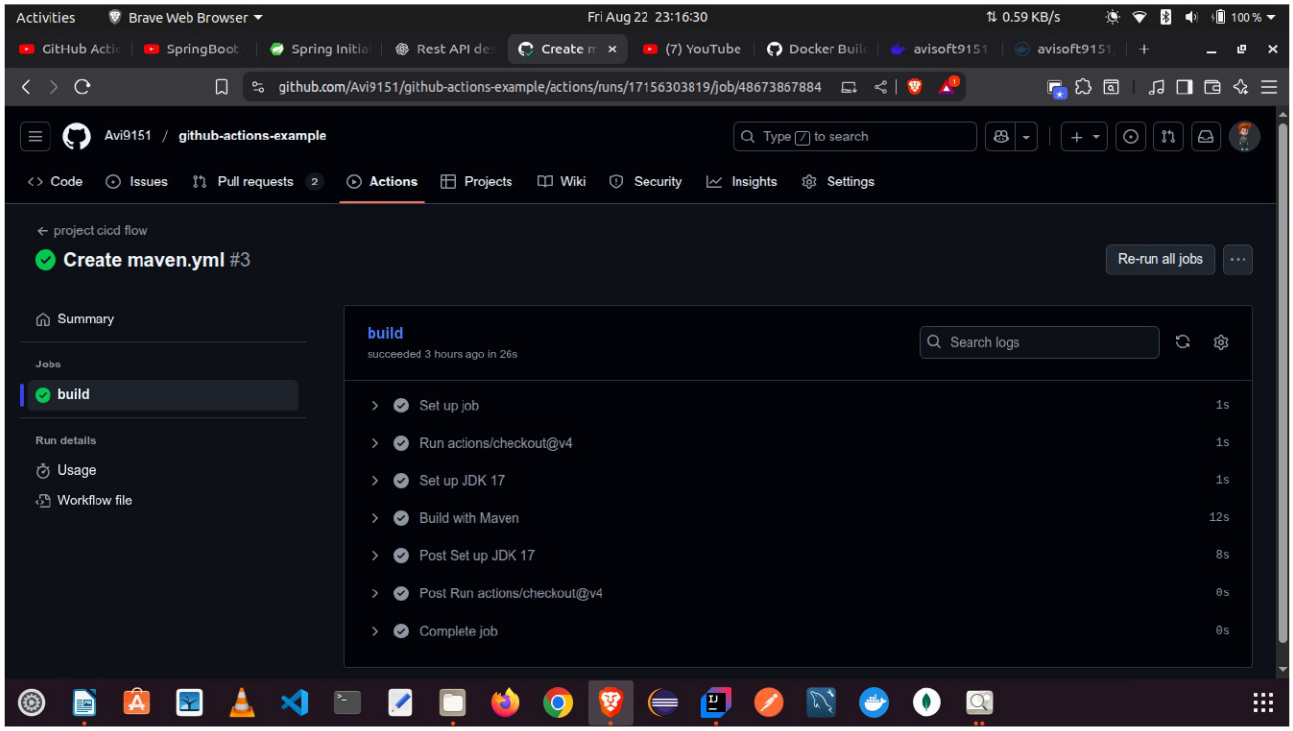
Step 15: Create new Repository.

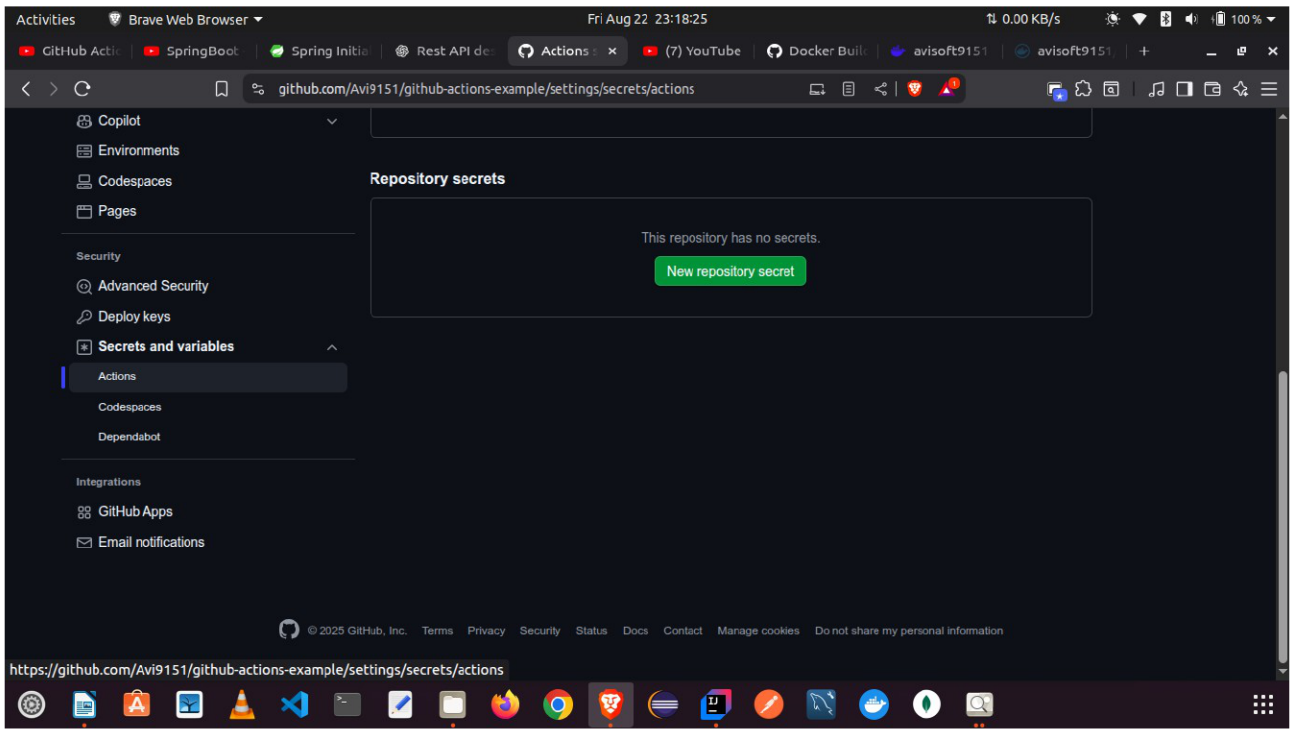


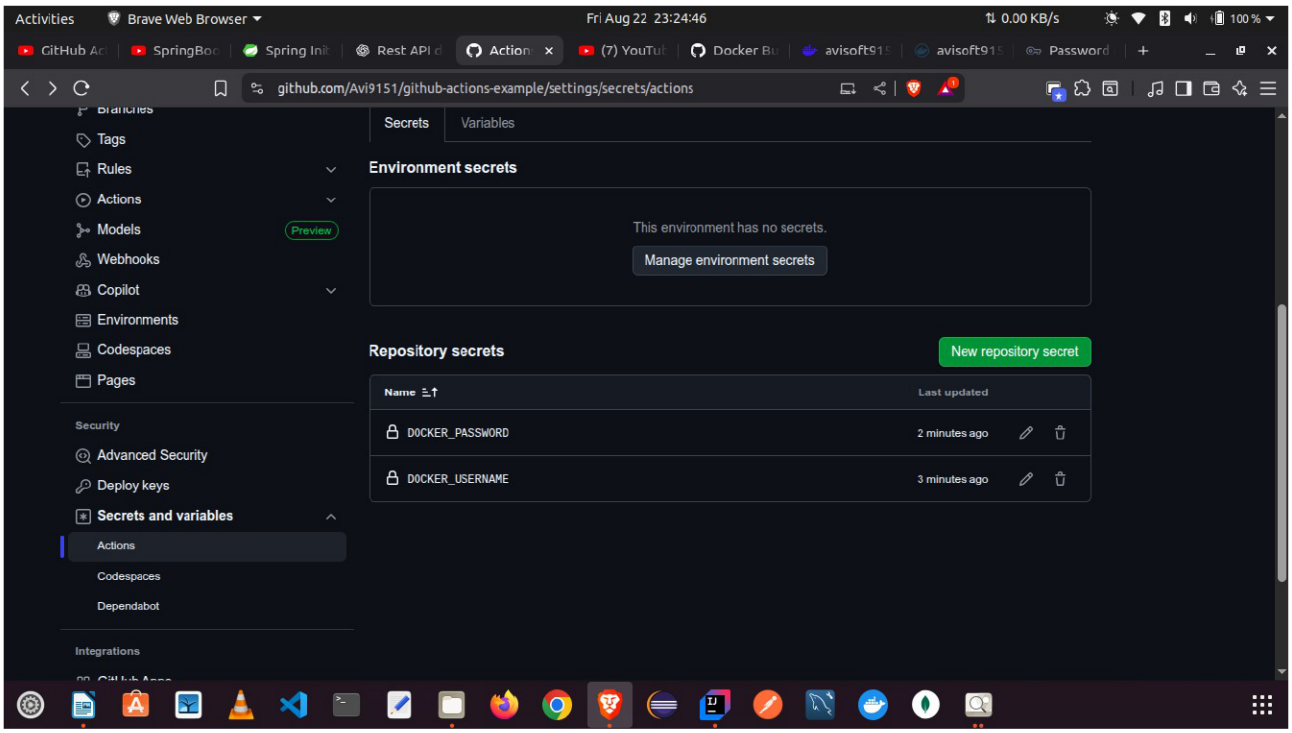
Step 16: Now update maven.yml code.

```yaml
name: project cicd flow

on:
  push:
    branches: [ "master" ]
  pull_request:
    branches: [ "master" ]

permissions:
  contents: write
  security-events: write

jobs:
  build:

    runs-on: ubuntu-latest

    steps:
    - uses: actions/checkout@v4
    - name: Set up JDK 17
      uses: actions/setup-java@v4
      with:
        java-version: '17'
        distribution: 'temurin'
        cache: maven
    - name: Build with Maven
      run: mvn clean install

    # Update from here
    - name: Build & push Docker image
      uses: mr-smithers-excellent/docker-build-push@v6
      with:
        image: avisoft9151/springboot-cicd-docker-image
        tags: latest
        registry: docker.io
        dockerfile: Dockerfile
        username: ${{ secrets.DOCKER_USERNAME }}
        password: ${{ secrets.DOCKER_PASSWORD }}
```

Step 17: Go to the github repo and click on settings -> Secrets and Variables -> Actions -> New Repository Secrets -> Add Secrets for DOCKER_USERNAME and DOCKER_PASSWORD







Step 18: Create docker file, click on github-cicid-actions -> create file Dockerfile and write code.

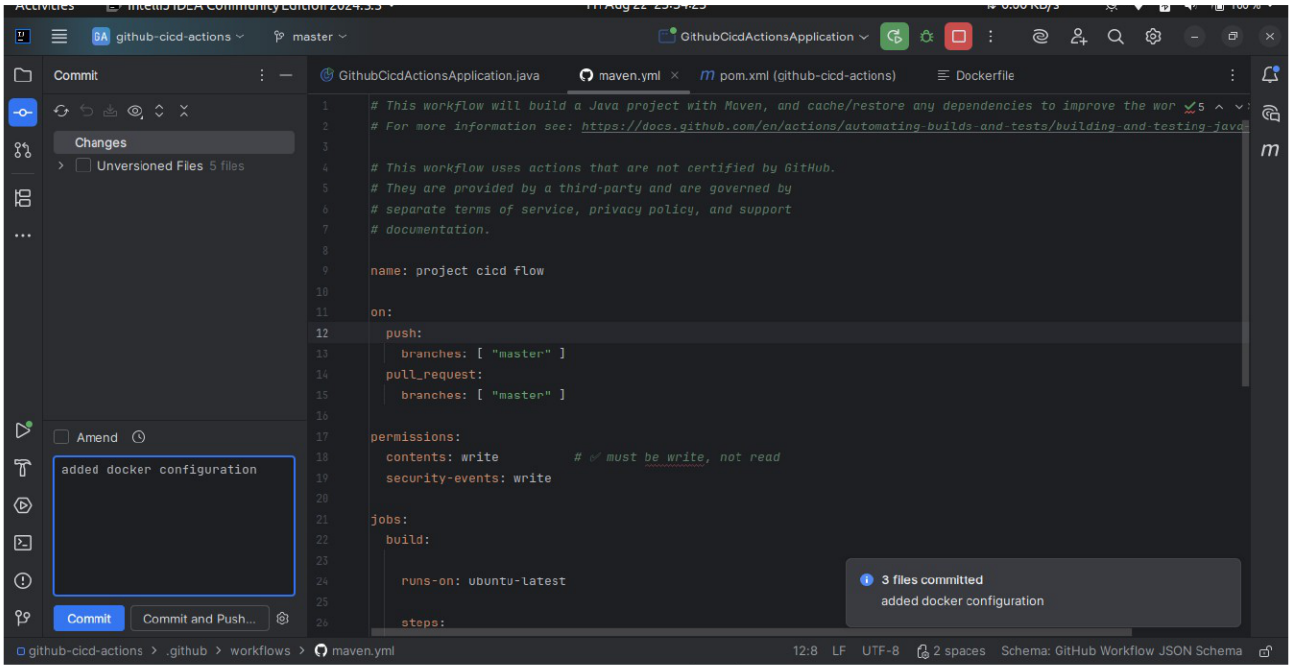
```
FROM openjdk:17
EXPOSE 8080
ADD target/springboot-cicd-docker-image.jar springboot-cicd-docker-image.jar
ENTRYPOINT ["java", "-jar", "/springboot-cicd-docker-image.jar"]
```

Step 19: Update pom.xml code and add this <finalName>.

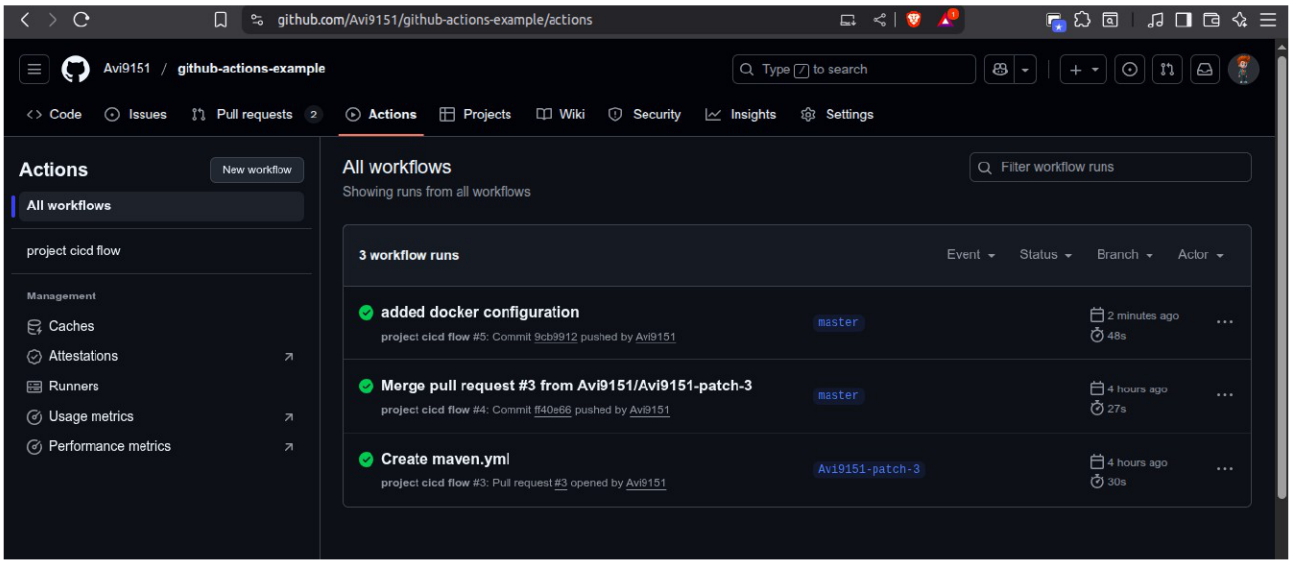
```
<build>
        <plugins>
                <plugin>
                        <groupId>org.springframework.boot</groupId>
                        <artifactId>spring-boot-maven-plugin</artifactId>
                </plugin>
        </plugins>
    <finalName>springboot-cicd-docker-image</finalName>
</build>
```

Note : It will create docker image and push to the docker hub.

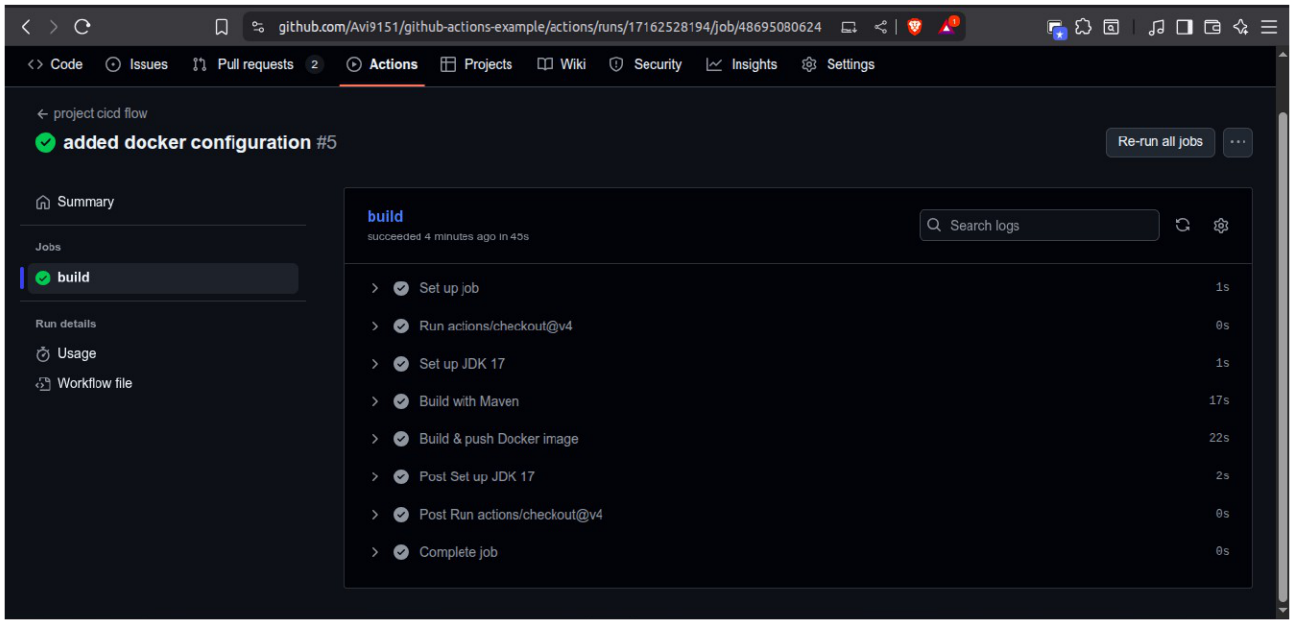
Step 20: Now open Intellij Idea and git commit and push.



Step 21: Now go to the github and click on Actions and you can see started the build.

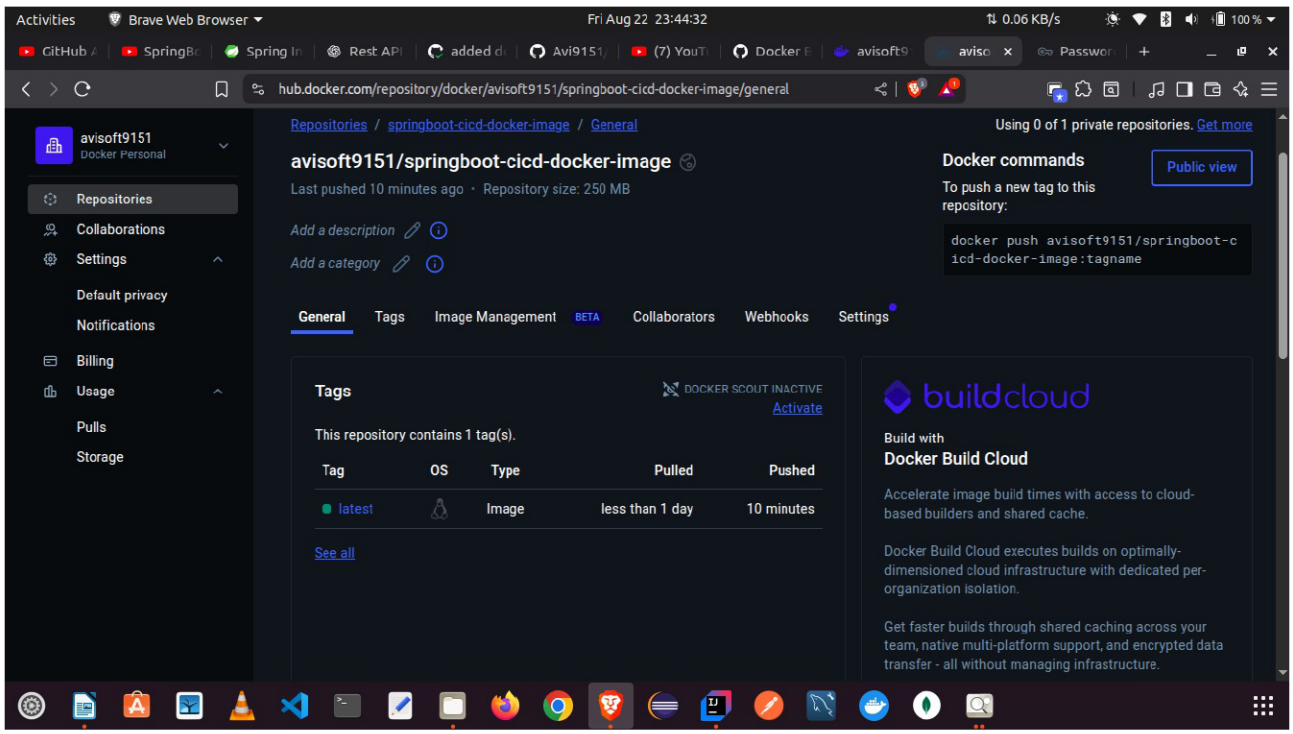


Step 22: Click on the added docker configuration -> build and you can see start build with maven



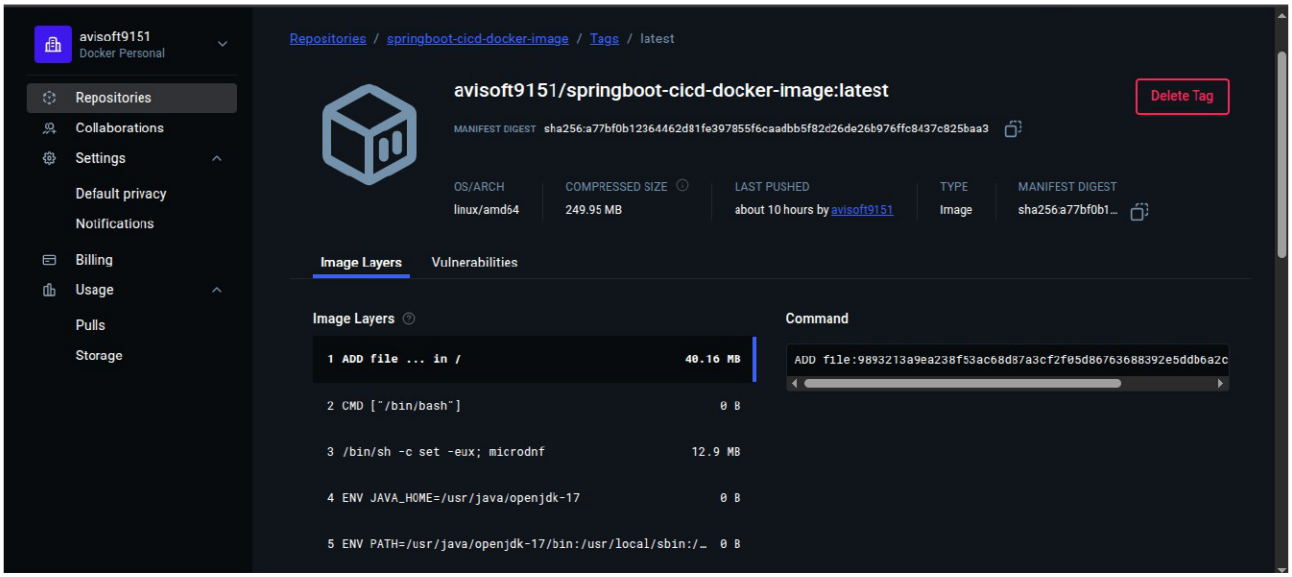
Note : Now you can see all build process completed and it should perticular image to docker hub

Step 23: Now open the Docker hub on browser and refresh the page.

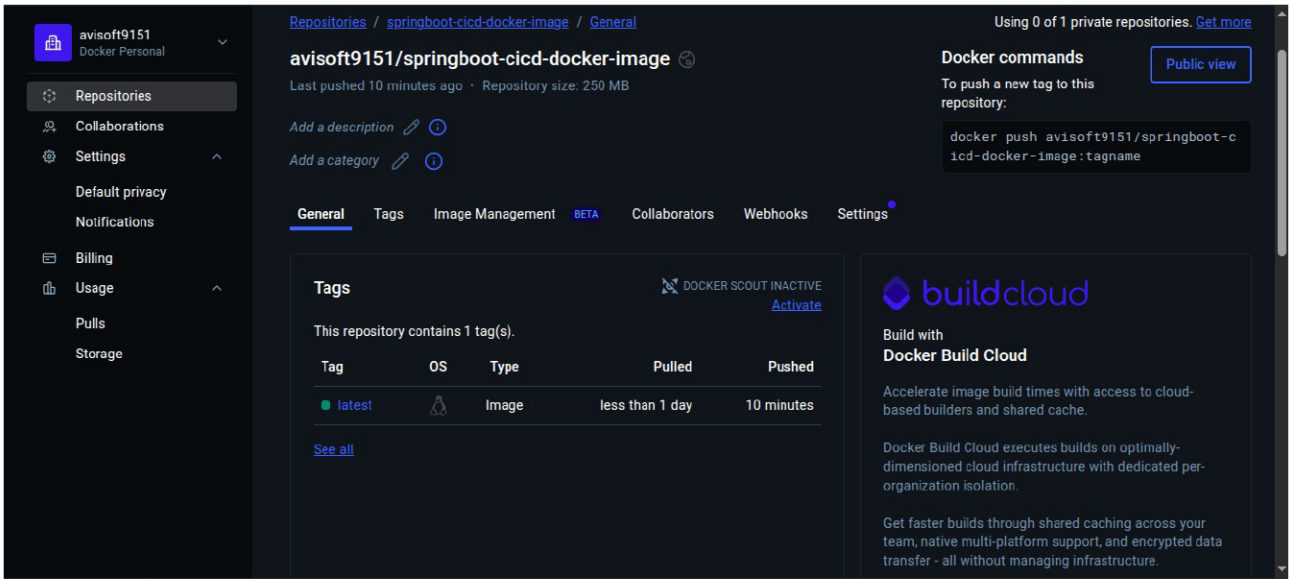


Note : Now you can see image got pushed on the docker hub.

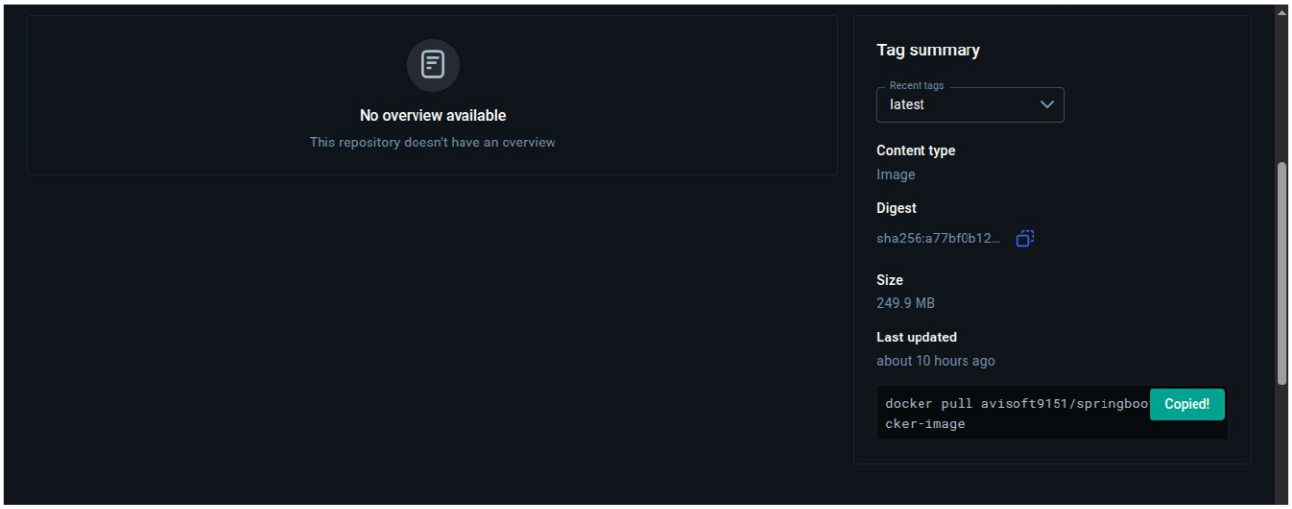
And you can click on latest tag and see.

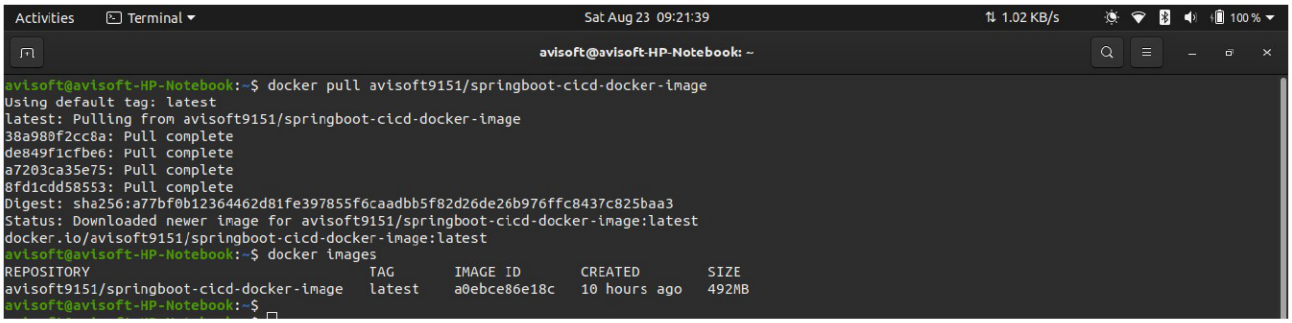


Step 24: We can pull this image and run on locally then click on **Public view**.

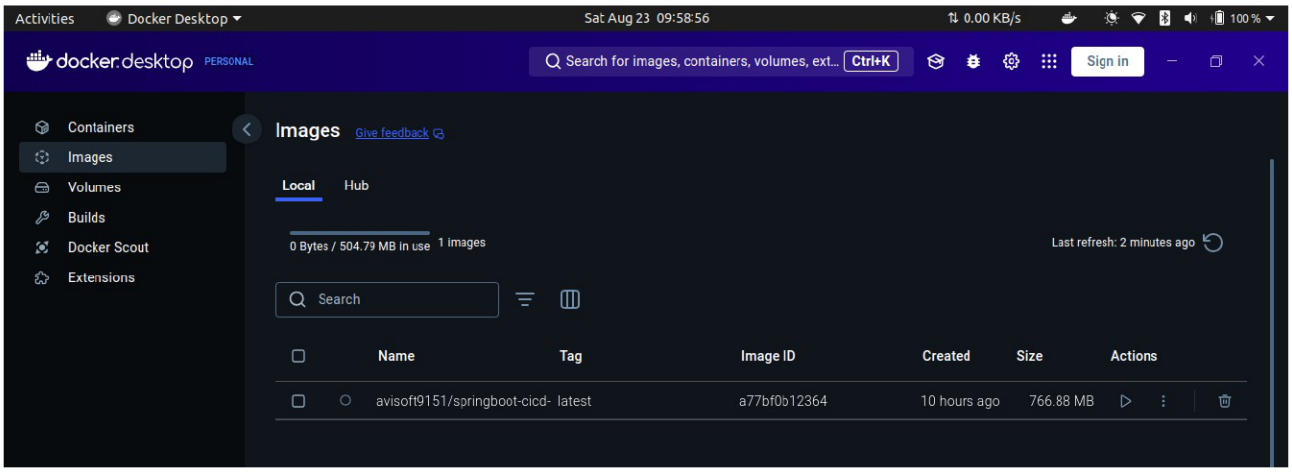


Copy this command and run it on terminal for local run.



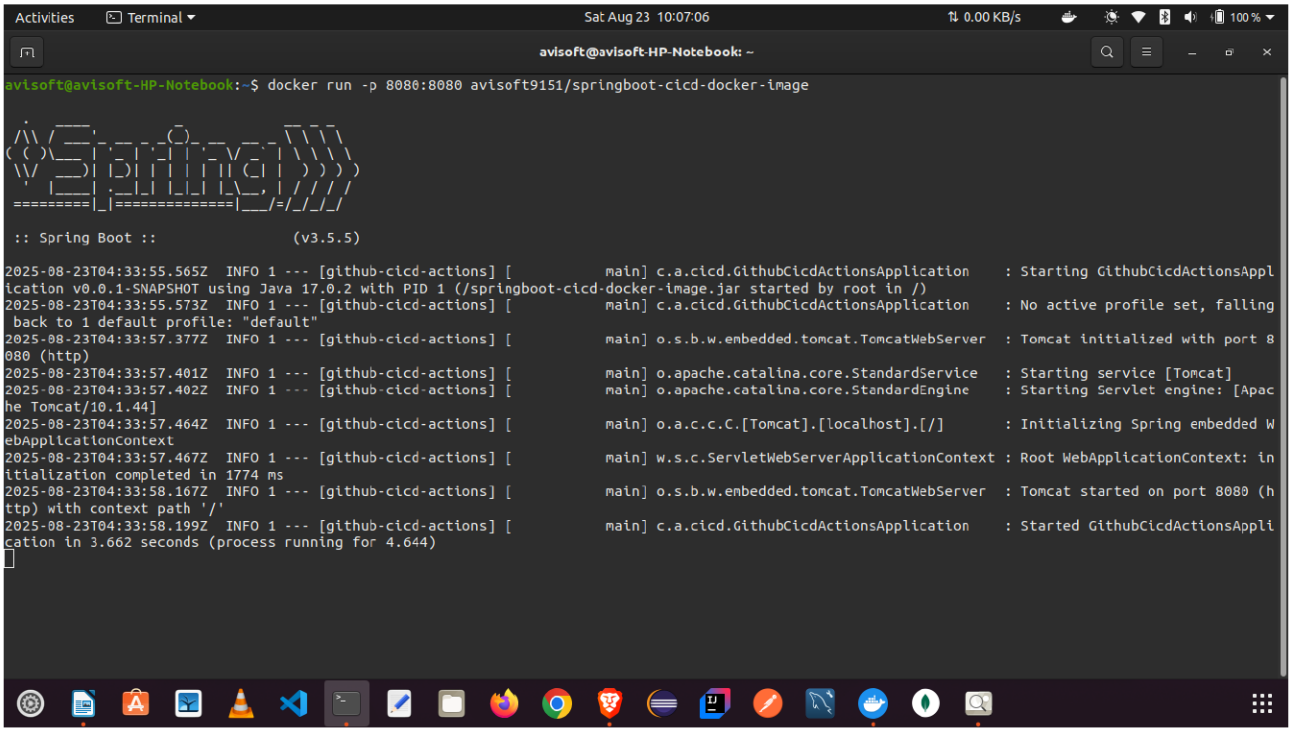


- You can see downloaded the docker image.
- And you observe the docker image add after run - docker images command run on terminal.
- Now open the Docker Desktop and you can see this image added.

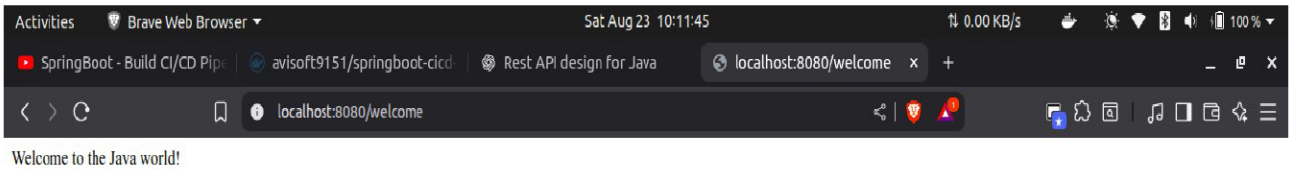


Now we run this image manually through command and run this command on terminal.

Command - [docker run -p 8080:8080 avisoft9151/springboot-cicd-docker-image]



- We can see Application started at port number 8080 and docker container.
- Now go to the browser and hit this endpoints - [http://localhost:8080/welcome]



Final Outcome

- **Fully automated CI/CD pipeline**:
    - Code push → GitHub Actions → Build → Docker Image → Push to Docker Hub.
- No manual build required.
- Can deploy this image anywhere (local, Kubernetes, cloud).