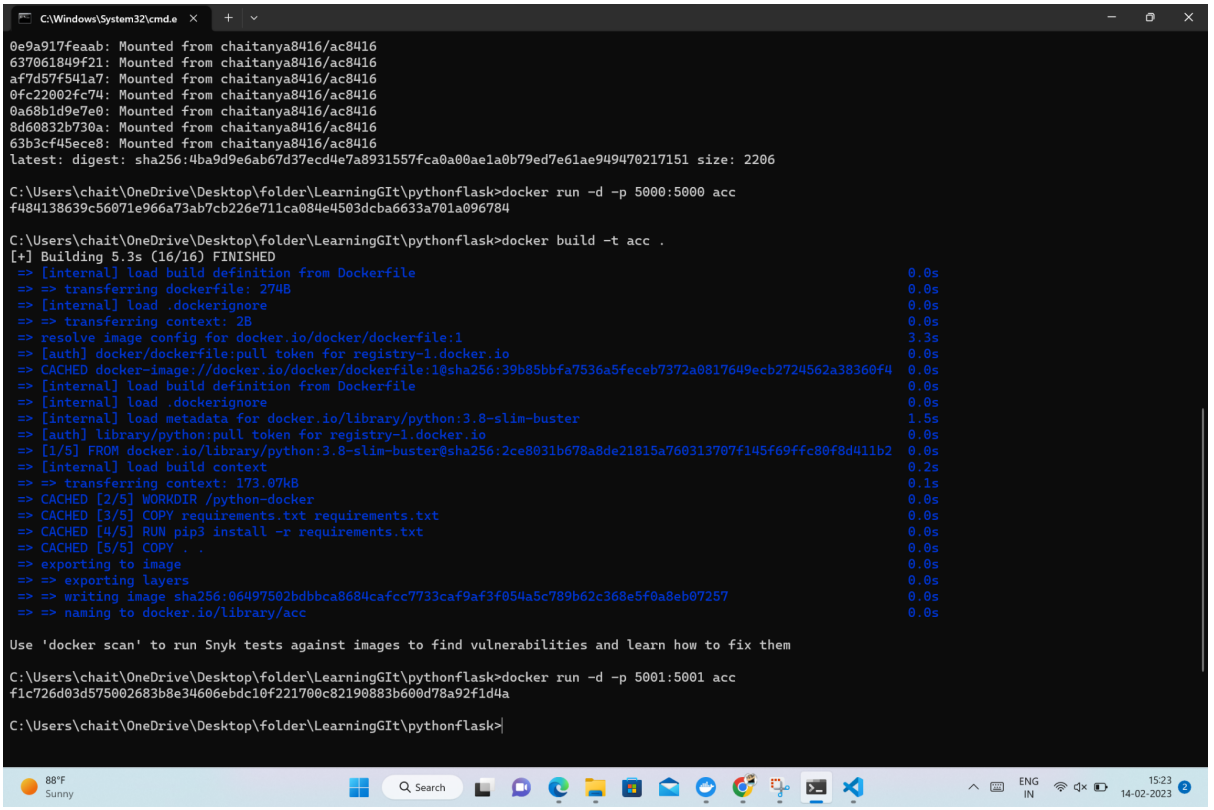
ASSIGNMENT -3

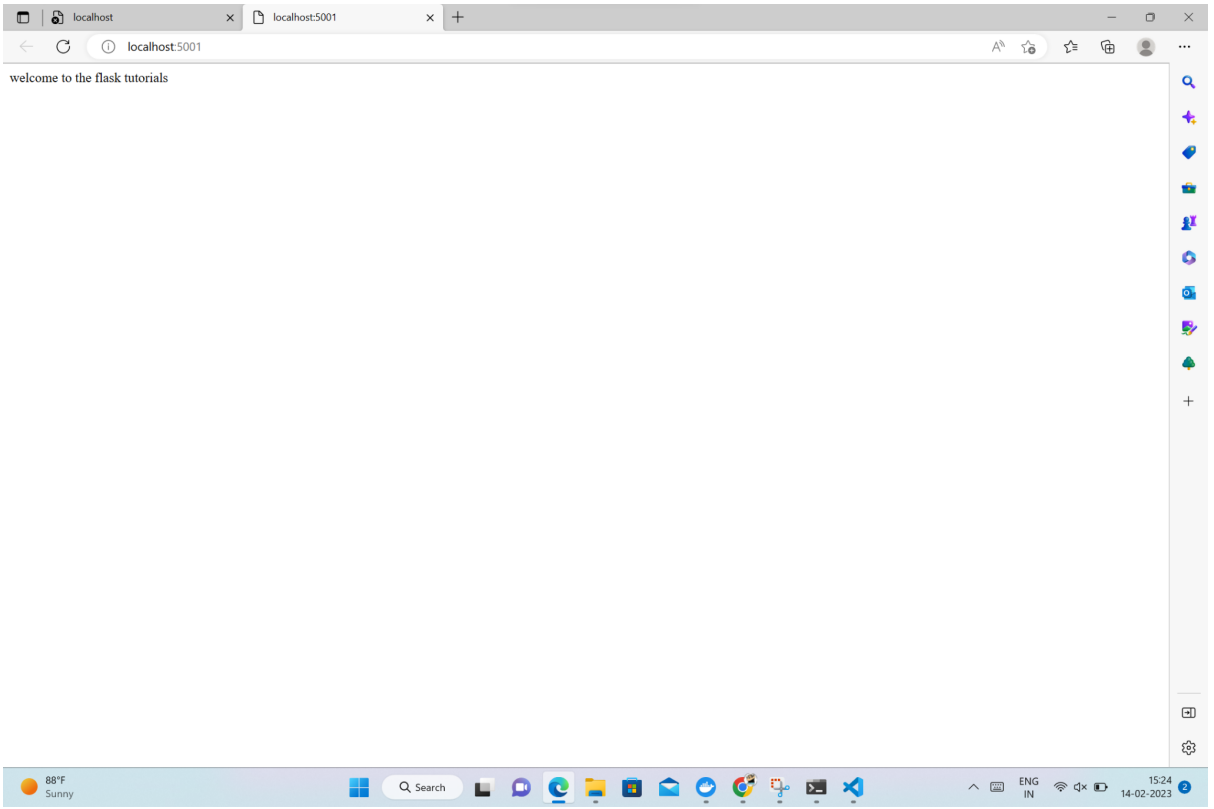
Q1.CREATION OF DOCKER CONTAINER AND IMAGE:

*PYTHON FLASK:

- Open the folder of python flask and go to the command prompt.
- Login to the docker desktop and dockerhub.com in the background.
- In command prompt enter the command docker login, then it will authenticate with our credentials.
- Now, enter “docker images” it will list all our images like python,ubuntu etc. if we have any images otherwise, there will be no images.
- Inorder to build our own image,enter“docker build -timage_name .”
- If we check docker images again then it will show the image which we have created before.
- Create a repository in dockerhub.com.
- Now enter “docker tag image_name:latest username/repository_name” and push the image into our repository using “docker push username/image_name.
- Now run the container and like docker run -d -p 5000:4000 imagename.
- Here 4000 indicates EXPOSE ->It tells about docker that our application will run on the port 4000.
- If we open the docker desktop, we can see that a container is in running state.
- Then it will show that ‘the site is not working’ then change the expose and again build the image and run the container.
- Now, it will show the output as ‘welcome to python flask tutorials’.

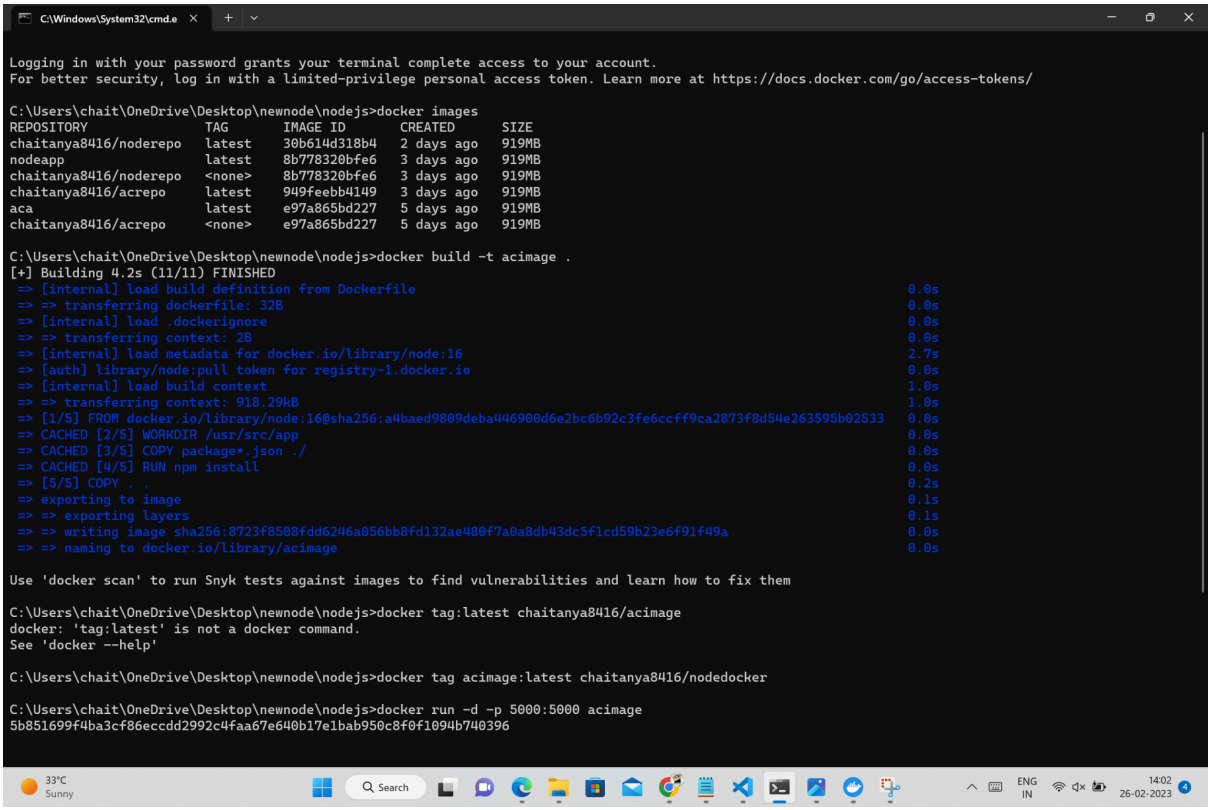

C:\Windows\System32\cmd.e
0e9a917feaab: Mounted from chaitanya8416/ac8416
637061849f21: Mounted from chaitanya8416/ac8416
af7d57f541a7: Mounted from chaitanya8416/ac8416
0fc22002fc74: Mounted from chaitanya8416/ac8416
0a68b1d9e7e0: Mounted from chaitanya8416/ac8416
8d60832b730a: Mounted from chaitanya8416/ac8416
63b3cf45ece8: Mounted from chaitanya8416/ac8416
latest: digest: sha256:4ba9d9e6ab67d37ecd4e7a8931557fca0a00ae1a0b79ed7e61ae949470217151 size: 2206
C:\Users\chait\OneDrive\Desktop\folder\LearningGIt\pythonflask>docker run -d -p 5000:5000 acc
f484138639c56071e966a73ab7cb226e711ca084e4503dcba6633a701a096784
C:\Users\chait\OneDrive\Desktop\folder\LearningGIt\pythonflask>docker build -t acc .
[+] Building 5.3s (16/16) FINISHED
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 274B 0.0s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> resolve image config for docker.io/docker/dockerfile:1 3.3s
=> [auth] docker/dockerfile:pull token for registry-1.docker.io 0.0s
=> CACHED docker-image://docker.io/docker/dockerfile:1@sha256:39b85bbfa7536a5feceb7372a0817649ecb2724562a38360f4 0.0s
=> [internal] load build definition from Dockerfile 0.0s
=> [internal] load .dockerignore 0.0s
=> [internal] load metadata for docker.io/library/python:3.8-slim-buster 1.5s
=> [auth] library/python:pull token for registry-1.docker.io 0.0s
=> [1/5] FROM docker.io/library/python:3.8-slim-buster@sha256:2ce8031b678a8de21815a760313707f145f69ffc80f8d411b2 0.0s
=> [internal] load build context 0.2s
=> => transferring context: 173.07kB 0.1s
=> CACHED [2/5] WORKDIR /python-docker 0.0s
=> CACHED [3/5] COPY requirements.txt requirements.txt 0.0s
=> CACHED [4/5] RUN pip3 install -r requirements.txt 0.0s
=> CACHED [5/5] COPY . . 0.0s
=> exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:06497502bdbbca8684cafcc7733caf9af3f054a5c789b62c368e5f0a8eb07257 0.0s
=> => naming to docker.io/library/acc 0.0s
Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
C:\Users\chait\OneDrive\Desktop\folder\LearningGIt\pythonflask>docker run -d -p 5001:5001 acc
f1c726d03d575002683b8e34606ebdc10f221700c82190883b600d78a92f1d4a
C:\Users\chait\OneDrive\Desktop\folder\LearningGIt\pythonflask>

OUTPUT:

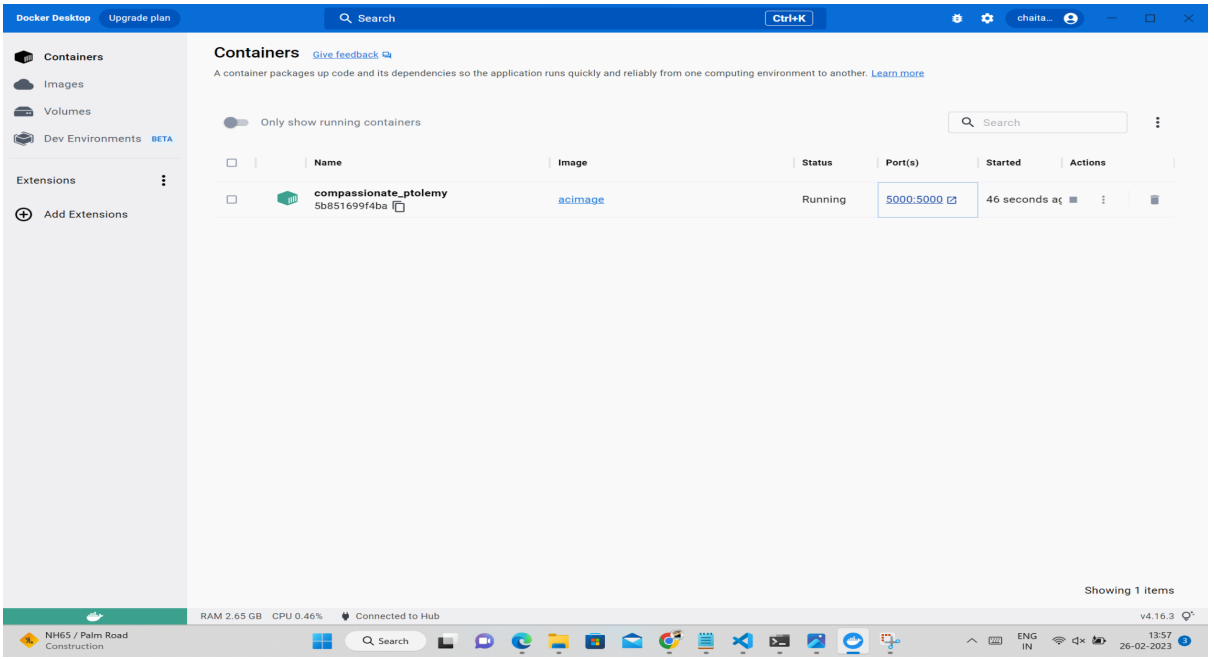

localhost
localhost:5001
localhost:5001
welcome to the flask tutorials

NODEJS:

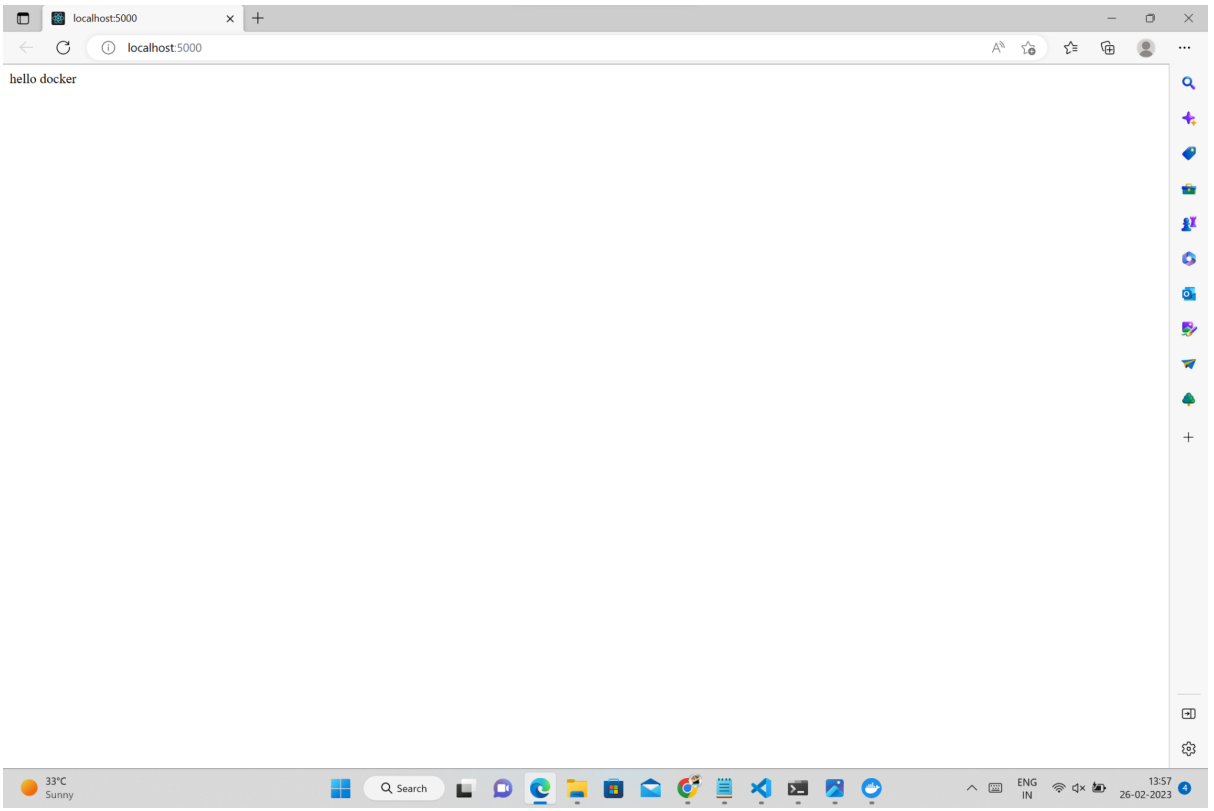
- Repeat the same process which we did for python flask.
- But instead of the python flask folder, open nodejs folder and enter all the commands required to run a container.



Container is running in docker desktop



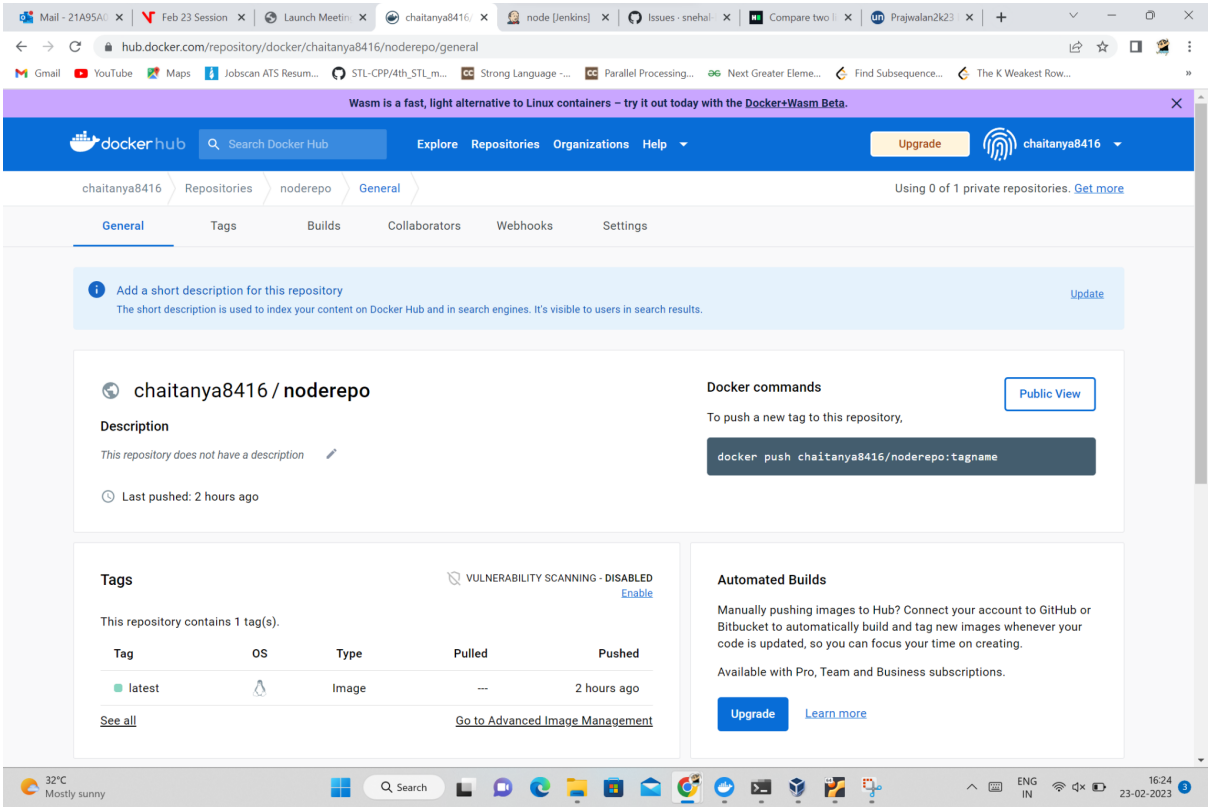
OUTPUT:

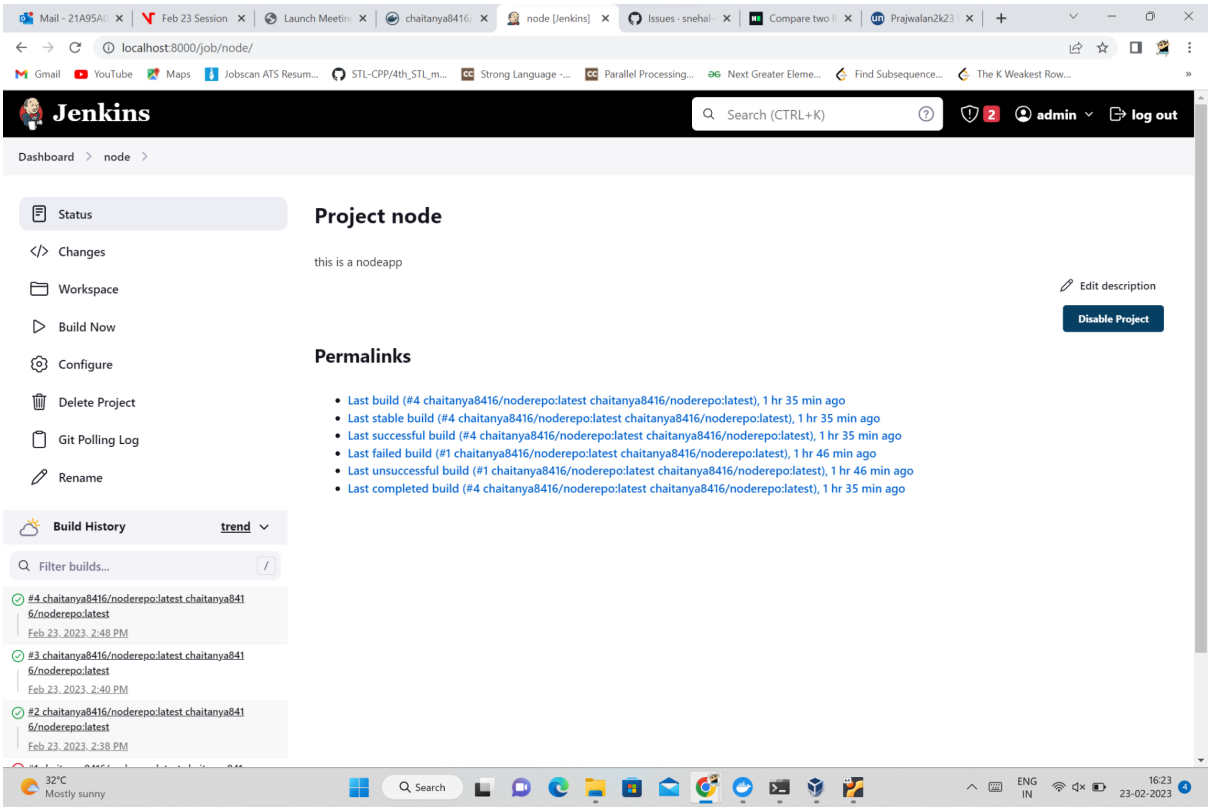


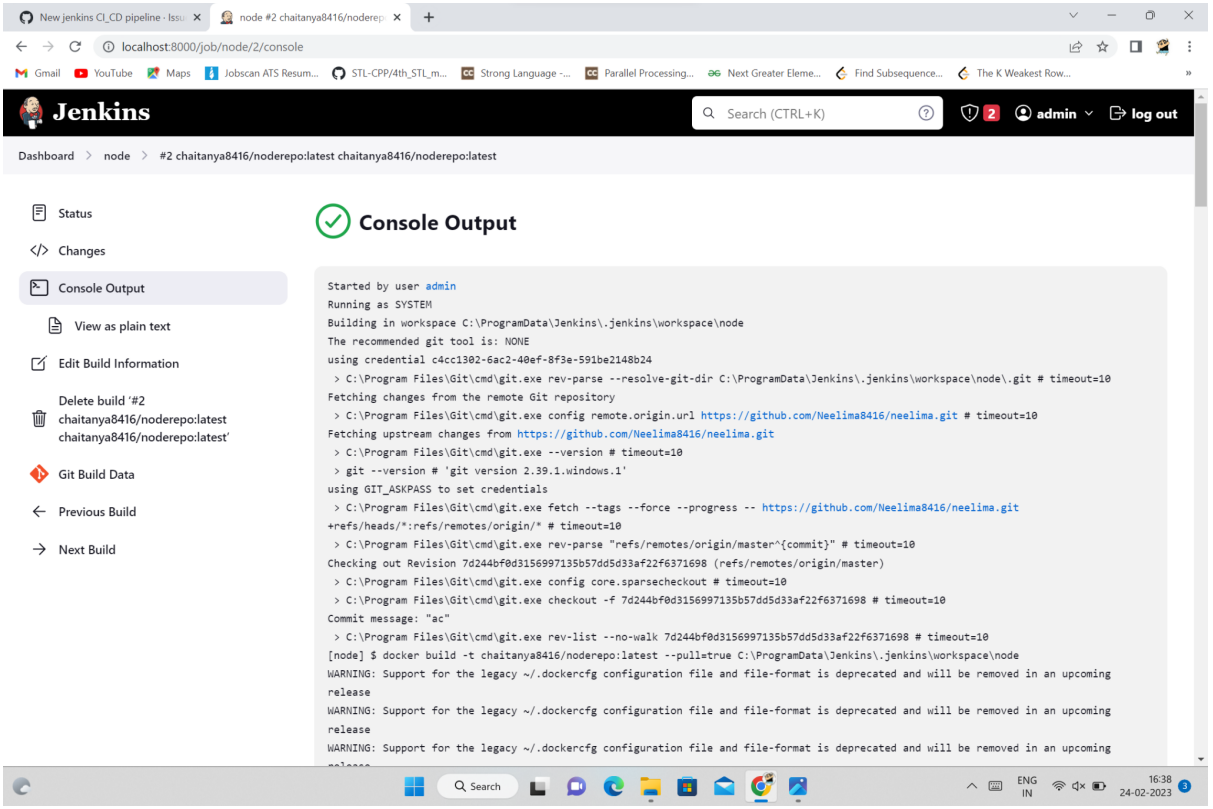
Q2.CI-CD PIPELINE APPLICATION USING JENKINS:

- Open nodejs folder in our local system and clear the path and open command prompt and create a repository in github.
- Now, enter the commands git init, git add . , git commit and git push origin master.
- Then all the files in nodejs folder will be pushed to our remote repository.
- Now, open docker desktop and open command prompt and enter docker login and build an image , and push that in to repository.
- We can find that in the repositories of docker hub.
- Now, open localhost:8000 here 8000 is our expose and create a new job (free style project)

Now, configure as follows:



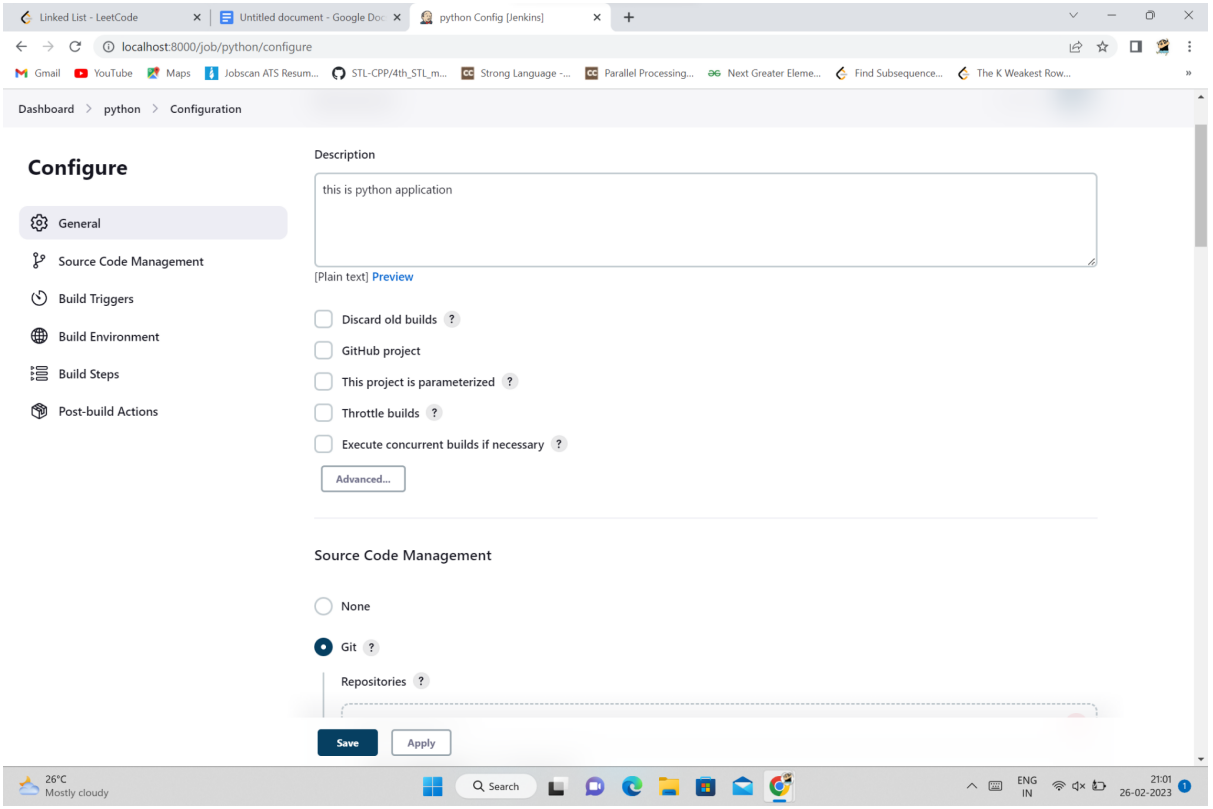


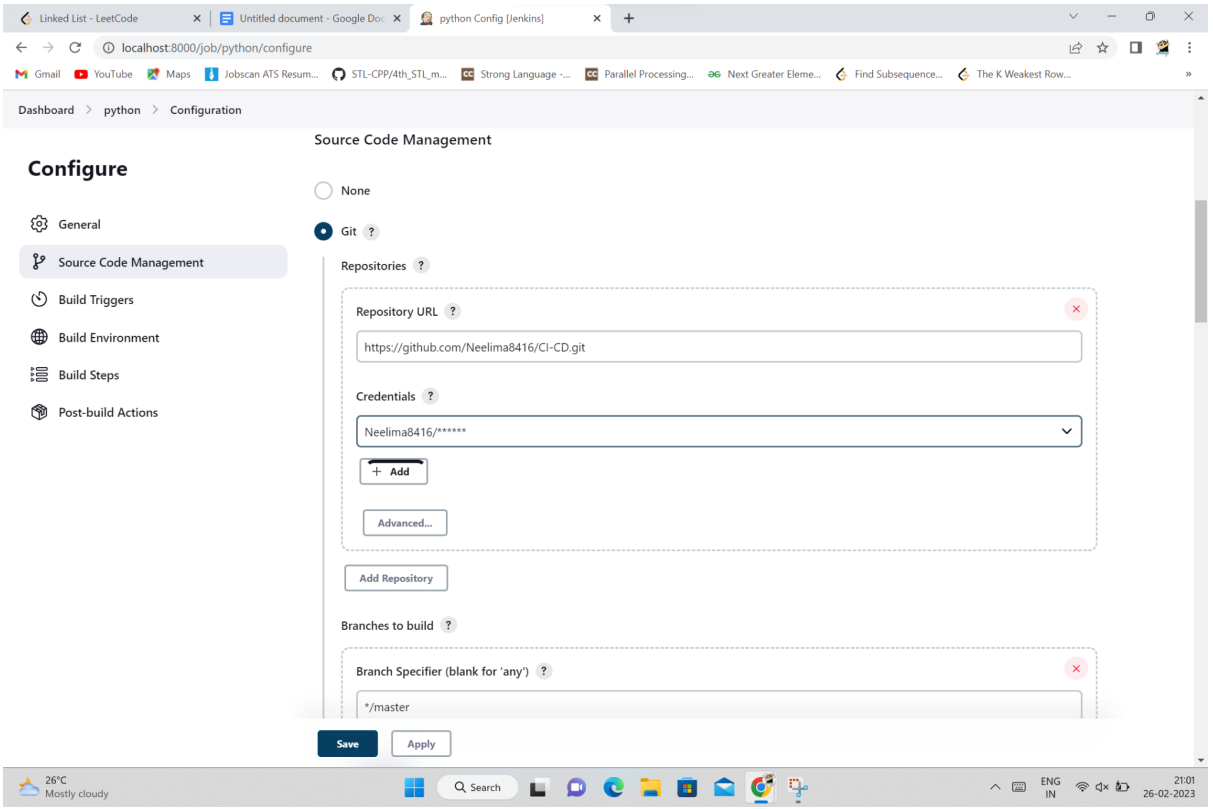


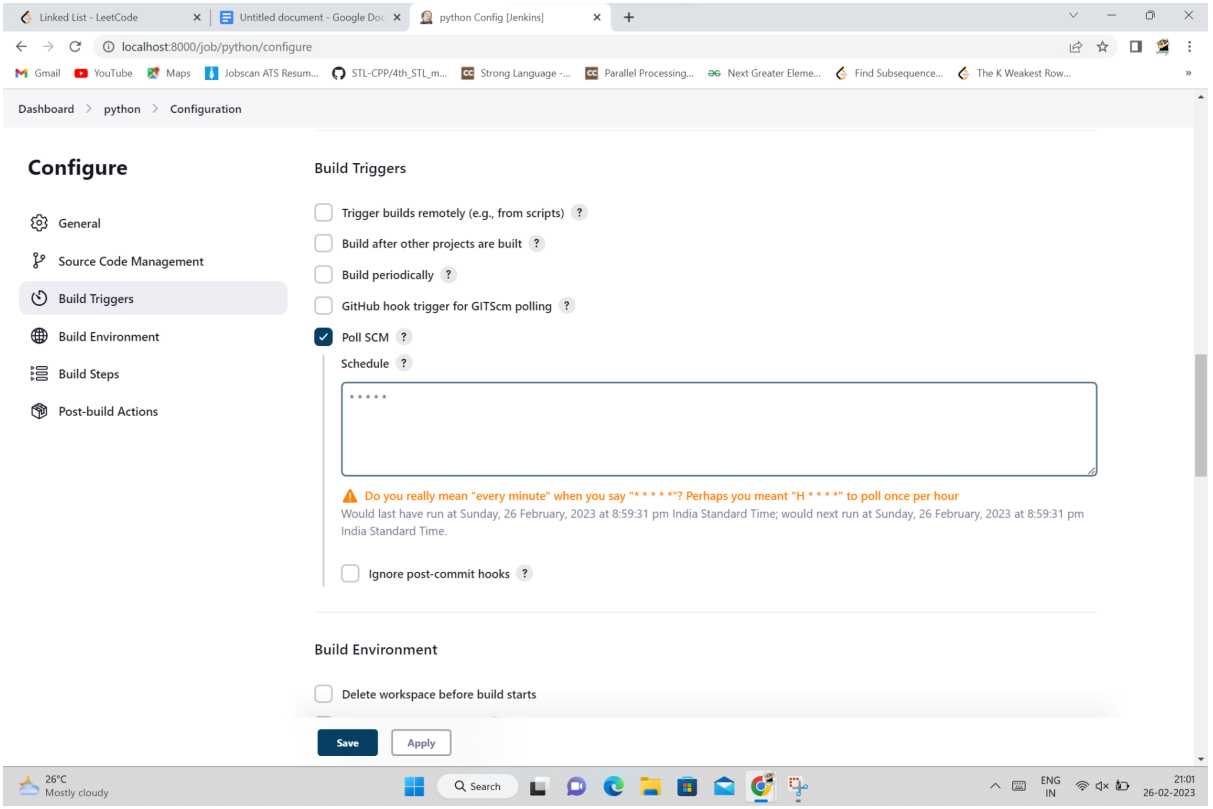
Q3.STEPS TO CREATE CI-CD PIPELINE FOR PYTHON APPLICATIONS:

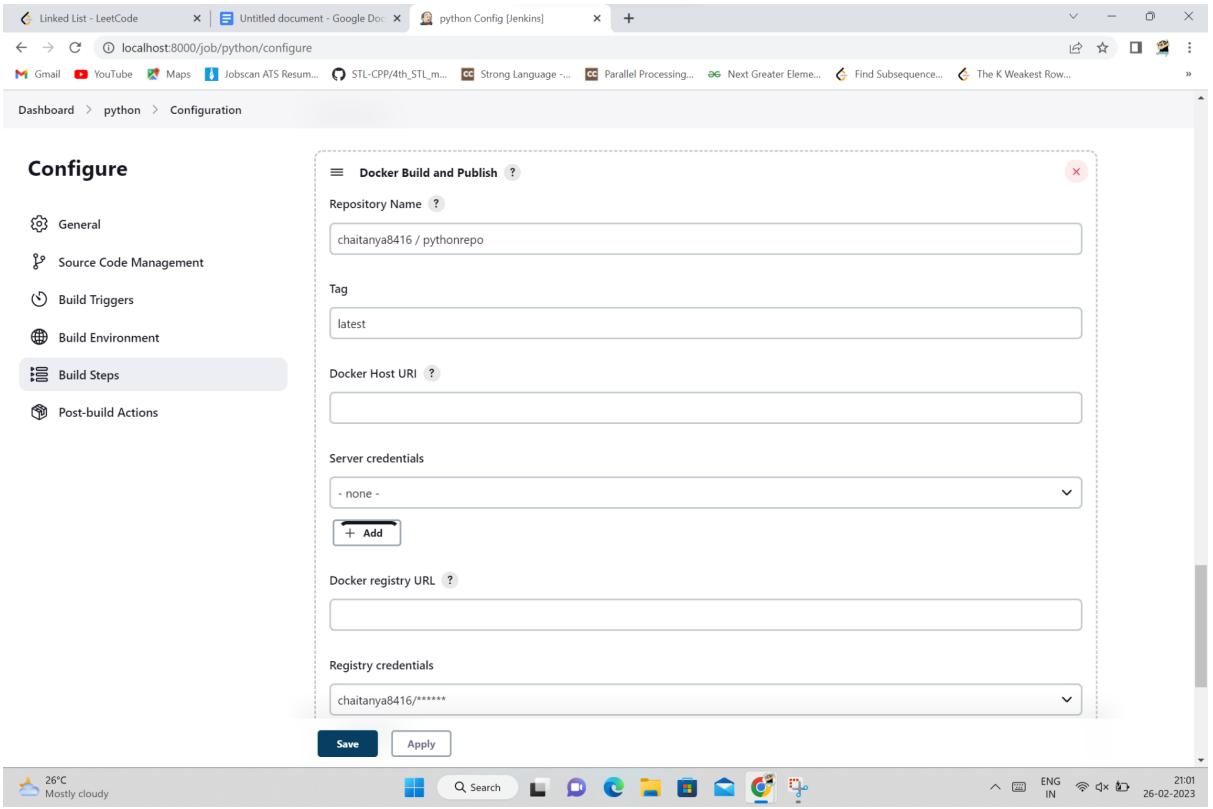
- Open jenkins like localhost:8000 and login with your credentials and add an item.
- Click on the freestyle project and click on OK.
- If we want to add a description we can add it in the description box.
- In source code management, enter the git repository URL which is containing the python flask application .
- If the repository is private, then add the credentials and select them otherwise select none.
- Now, in build triggers select poll SCM enter * * * * * it means that the task will be done every minute.
- In build steps, select Docker build and publish and add the docker hub repository credentials where the docker image and container for python flask is present.
- Now, save and apply and select the option build now.

- If the build is successful, then it indicates that a CI-CD pipeline is created successfully.






Linked List - LeetCode
Untitled document - Google Doc
python Config [Jenkins]
localhost:8000/job/python/configure
Gmail
YouTube
Maps
Jobscan ATS Resum...
STL-CPP/4th_STL_m...
Strong Language -...
Parallel Processing...
Next Greater Eleme...
Find Subsequence...
The K Weakest Row...
Dashboard
python
Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Build Triggers
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Build periodically
GitHub hook trigger for GITScm polling
Poll SCM
Schedule
* * * * *
Do you really mean "every minute" when you say "* * * * *"? Perhaps you meant "H * * * *" to poll once per hour
Would last have run at Sunday, 26 February, 2023 at 8:59:31 pm India Standard Time; would next run at Sunday, 26 February, 2023 at 8:59:31 pm India Standard Time.
Ignore post-commit hooks
Build Environment
Delete workspace before build starts
Save
Apply
26°C
Mostly cloudy
Search
ENG
IN
21:01
26-02-2023


Linked List - LeetCode
Untitled document - Google Doc
python Config [Jenkins]
localhost:8000/job/python/configure
Gmail
YouTube
Maps
Jobscan ATS Resum...
STL-CPP/4th_STL_m...
Strong Language -...
Parallel Processing...
Next Greater Eleme...
Find Subsequence...
The K Weakest Row...
Dashboard
python
Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
Docker Build and Publish
Repository Name
chaitanya8416 / pythonrepo
Tag
latest
Docker Host URI
Server credentials
- none -
Add
Docker registry URL
Registry credentials
chaitanya8416/******
Save
Apply
26°C
Mostly cloudy
Search
ENG
IN
21:01
26-02-2023