Q1.Create an docker container and image out of dockerfile for following application:

- Python Flask
- Nodejs

Kindly, provide the details of steps written in dockerfile and why the steps are required with explanation.

All the steps from creating images to container and running the container should be provided in a screenshot , where your username can be seen in the terminal.

1)STEPS TO FOLLOW DOCKER CONTAINER AND RUN THE DOCKER IMAGE ON THE DOCKER DESKTOP

*Open the git bash then enter cd Desktop as command

*Then go to docker login and the docker images

*Open the Docker hub.com and Docker desktop in the background *It shows our docker images which are in the docker hub repository

*If there is no images on the docker hub repository then open the docker hub.com and select our required images like UBUNTU,NODEJS..

*Then enter the command as docker pull ubuntu in git bash and then docker images.so it will show our ubuntu as image in our repository

*Then enter ls command then it will shows all our files in the folder.

*And next create an empty folder and copy the NODEJS folder from the learning git and paste it to our created empty folder and name the folder as NODEJS

//Package.json tells about external libraries,version

//FROM:It defines which service image to be used for your application

//WORKDIR:It defines the folder structure of application inside the container and where everything you download will be there

//COPY: Which contains our application dependency and keep it in a desired folder

//EXPOSE: It tells about docker that our application will run on port 3000 inside the docker

//RUN: It tells about docker to perform installization of whatever is there in your dependency file

//CMD: It tells docker to run the command in command line environment to start the application

*Then clear the path of the NODEJS folder and enter the command as cmd *A command prompt will be displayed on the screen and build a image *Then enter docker build -t image name .

*if we check the docker images it will be displayed the different images of our particular repository

*Then create a repository in our docker hub.com

*Then docker tag image name:latest username/repo name

*Atlast we have to push the image to the docker desktop so that enter the command as docker push username/reponame

*If we check the repositories in dockerhub our created repository will be there

*Finally run the container in the docker so docker run -d -p 5000:3000 imagename *Then it will shows the error as THIS SITE CAN'T BE REACHED

So again go to the NODEJS folder which we created previously.so change the expose

*So again go to the container and run it so it will shows the output as HELLO DOCKER

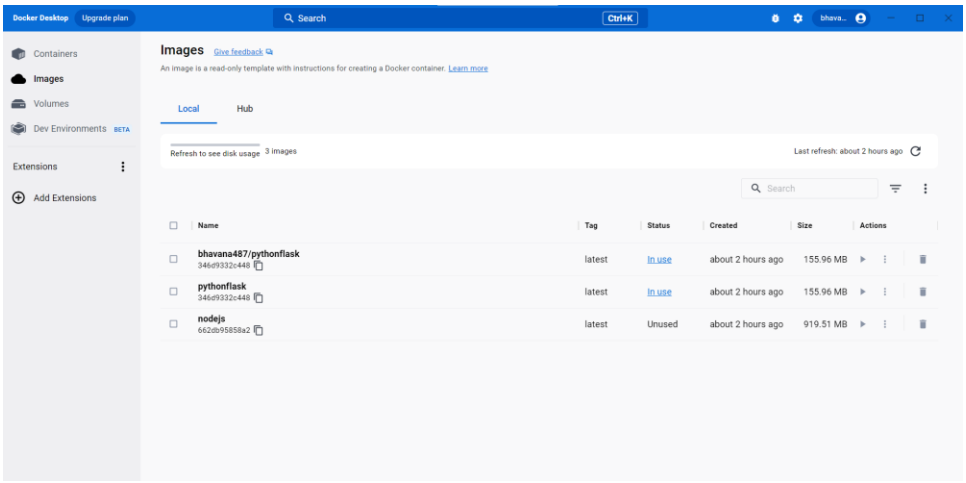
*It will be same for PYTHONFLASK instead of NODEJS

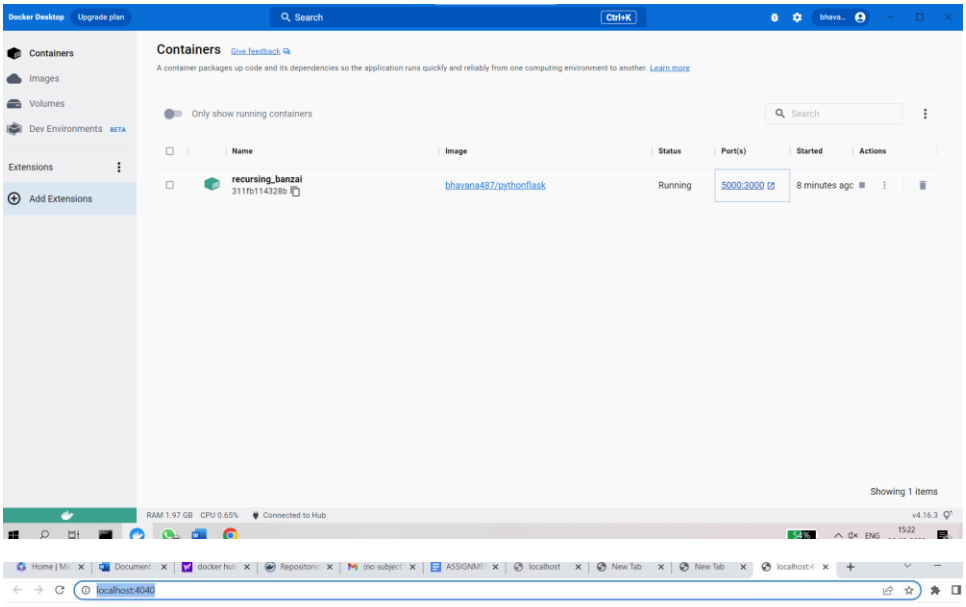
*It shows as output as WELCOME TO FLASK TUTORIALS

Creating docker image and container for python flask application

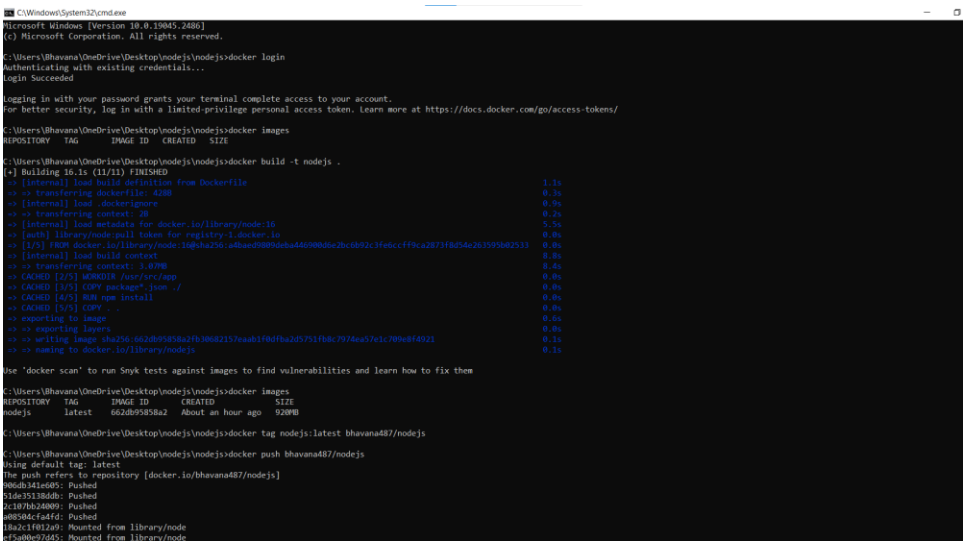
```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.2486]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Bhavana\OneDrive\Desktop\python flask\pythonflask>docker login
Authenticating with existing credentials...
Login Succeeded

Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/access-tokens/

C:\Users\Bhavana\OneDrive\Desktop\python flask\pythonflask>docker images
REPOSITORY   TAG       IMAGE ID       CREATED             SIZE
nodejs       latest    662db95858a2   About an hour ago   920MB

C:\Users\Bhavana\OneDrive\Desktop\python flask\pythonflask>docker build -t pythonflask .
[+] Building 30.1s (16/16) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 274B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => resolve image config for docker.io/docker/dockerfile:1
 => [auth] docker/dockerfile:pull token for registry-1.docker.io
 => CACHED docker-image://docker.io/docker/dockerfile:1@sha256:39b85bbfa7536a5feceb7372a0817649ecb2724562a38360f4d6a7782a409b14
 => [internal] load build definition from Dockerfile
 => [internal] load .dockerignore
 => [internal] load metadata for docker.io/library/python:3.8-slim-buster
 => [auth] library/python:pull token for registry-1.docker.io
 => [1/5] FROM docker.io/library/python:3.8-slim-buster@sha256:b6625da729591bb6a922bcefc903668dd977d353492e53fbe9d399b9f41be32b
 => [internal] load build context
 => => transferring context: 28.74MB
 => CACHED [2/5] WORKDIR /python-docker
 => CACHED [3/5] COPY requirements.txt requirements.txt
 => CACHED [4/5] RUN pip3 install -r requirements.txt
 => CACHED [5/5] COPY . .
 => exporting to image
 => => exporting layers
 => => writing image sha256:346d9332c44818f3439ac5217b9058dd5d769b5650a36e7b4337f05f57463f57
 => => naming to docker.io/library/pythonflask

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them

C:\Users\Bhavana\OneDrive\Desktop\python flask\pythonflask>docker images
REPOSITORY    TAG       IMAGE ID       CREATED             SIZE
pythonflask   latest    346d9332c448   About an hour ago   156MB
nodejs        latest    662db95858a2   About an hour ago   920MB

C:\Users\Bhavana\OneDrive\Desktop\python flask\pythonflask>
```

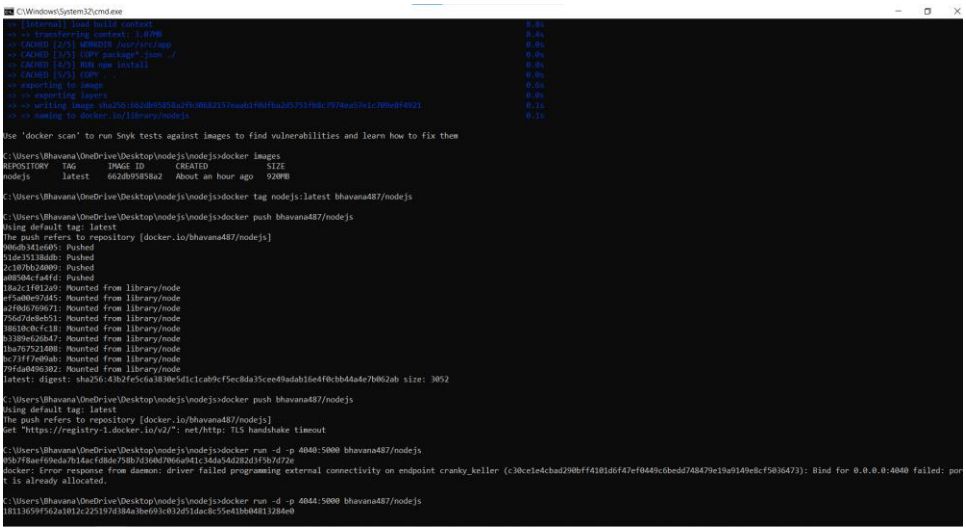
```
C:\Windows\System32\cmd.exe
 => => writing image sha256:346d9332c44818f3439ac5217b9058dd5d769b5650a36e7b4337f05f57463f57                                0.1s
 => => naming to docker.io/library/pythonflask                                                                              0.1s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them

C:\Users\Bhavana\OneDrive\Desktop\python flask\pythonflask>docker images
REPOSITORY    TAG       IMAGE ID       CREATED             SIZE
pythonflask   latest    346d9332c448   About an hour ago   156MB
nodejs        latest    662db95858a2   About an hour ago   920MB

C:\Users\Bhavana\OneDrive\Desktop\python flask\pythonflask>docker tag pythonflask:latest bhavana487/pythonflask

C:\Users\Bhavana\OneDrive\Desktop\python flask\pythonflask>docker push bhavana487/pythonflask
Using default tag: latest
The push refers to repository [docker.io/bhavana487/pythonflask]
7c8dd92baa15: Pushed
6d4142b6a4b: Pushed
fdfad6f74d6b: Pushed
0d2552ca0441: Pushed
66bb61f27c3d: Mounted from library/python
0fc22002fc7a: Mounted from library/python
0a68b1d9e7e0: Mounted from library/python
8d60832b730a: Mounted from library/python
63b3cf45ece8: Mounted from library/python
latest: digest: sha256:21f5eb39604873b873e1b7d1b398dda52ba5ce2c0f1228710e329210113ae5a5 size: 2206

C:\Users\Bhavana\OneDrive\Desktop\python flask\pythonflask>docker run -d -p 5000:3000
"docker run" requires at least 1 argument.
See 'docker run --help'.

Usage:  docker run [OPTIONS] IMAGE [COMMAND] [ARG...]

Run a command in a new container

C:\Users\Bhavana\OneDrive\Desktop\python flask\pythonflask>docker run -d -p 5000:3000 bhavana487
Unable to find image 'bhavana487:latest' locally
docker: Error response from daemon: pull access denied for bhavana487, repository does not exist or may require 'docker login': denied: requested access to the resource is denied.
See 'docker run --help'.

C:\Users\Bhavana\OneDrive\Desktop\python flask\pythonflask>docker run -d -p 5000:3000 bhavana487/pythonflask
211fb114328ba0640d75447c8ec61cd199f7706f442e569e597318f05ee9f1a5

C:\Users\Bhavana\OneDrive\Desktop\python flask\pythonflask>
```





Creating docker image and container for nodejs application





Steps for creating dockerfile: 1)FROM : It defines which service image

2)create app directory : WORKDIR : it defines the folder structure of the application inside the container.

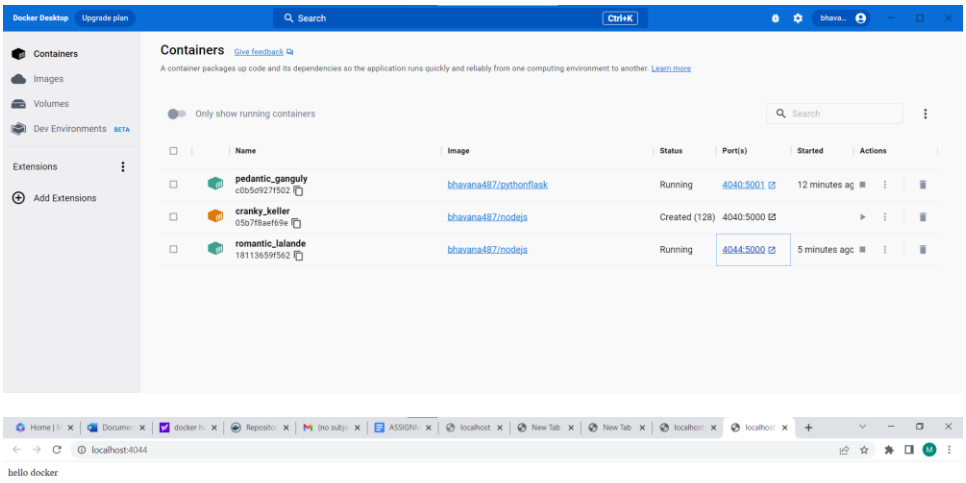
3)Install app dependencies: COPY : It tells the file which contains our application dependency

4)RUN : It tells docker to perform installation

5) COPY . . : Copy all the stuff just got download and keep it on desired file

6) Expose : It tells our docker that our application will run on port 3000 inside the container

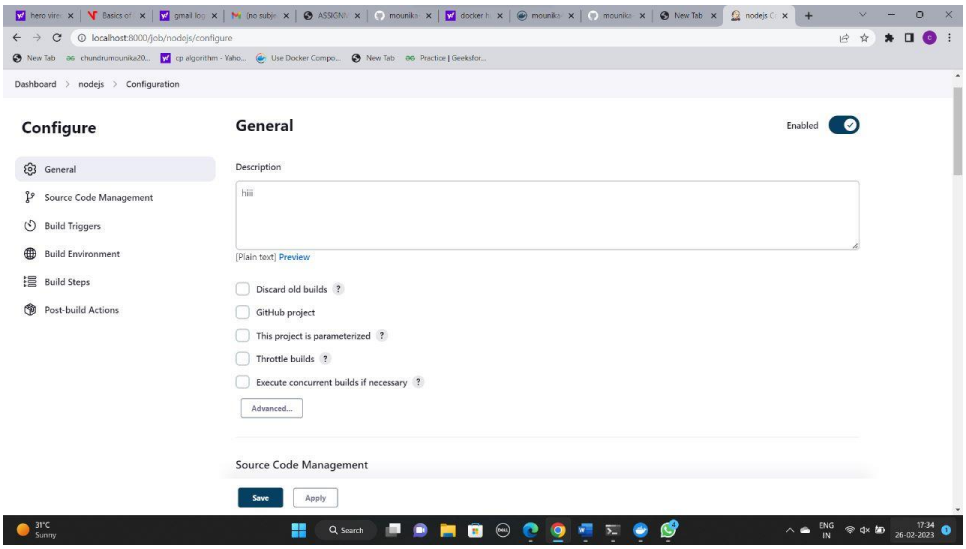
7) CMD [“node” , “filename”] : It tells the docker to run this command in command line environment to start the application, Here node is nothing but message

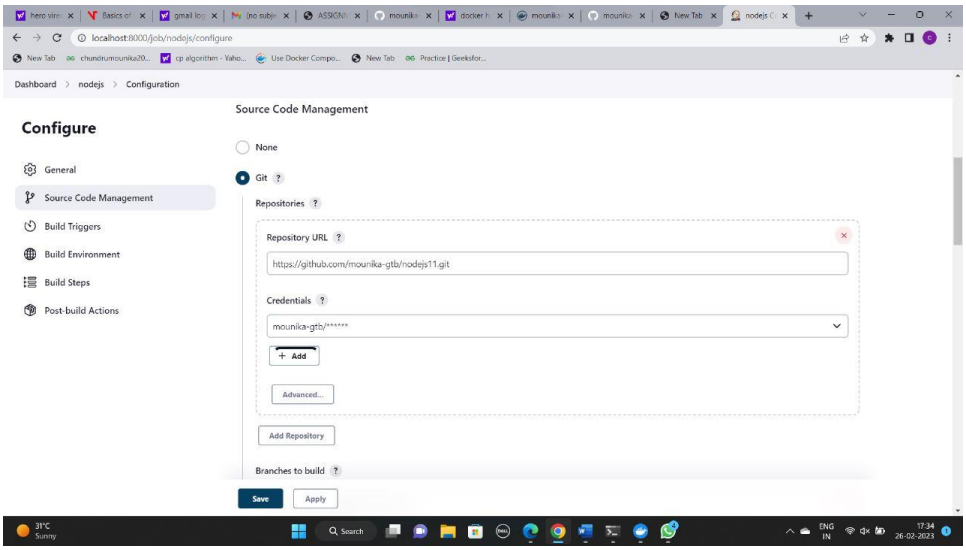


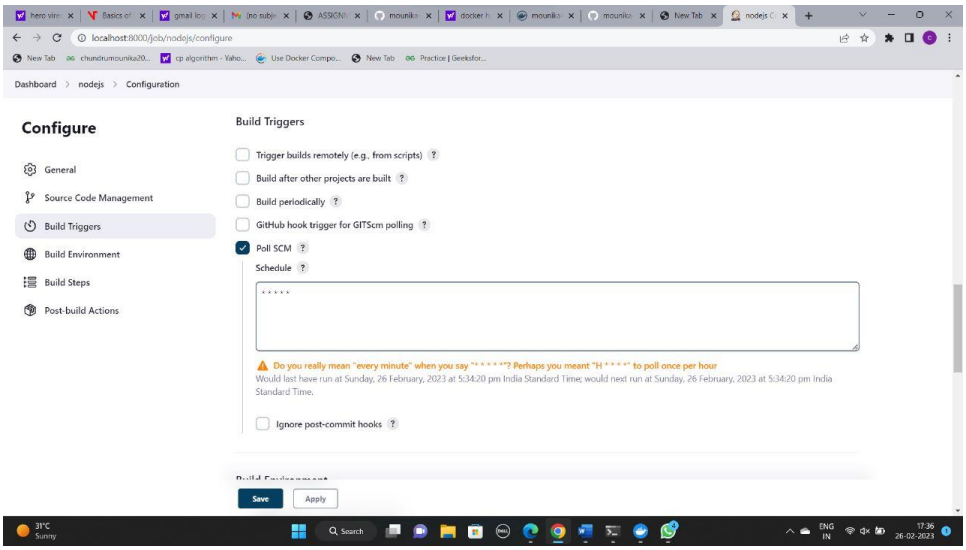
==========

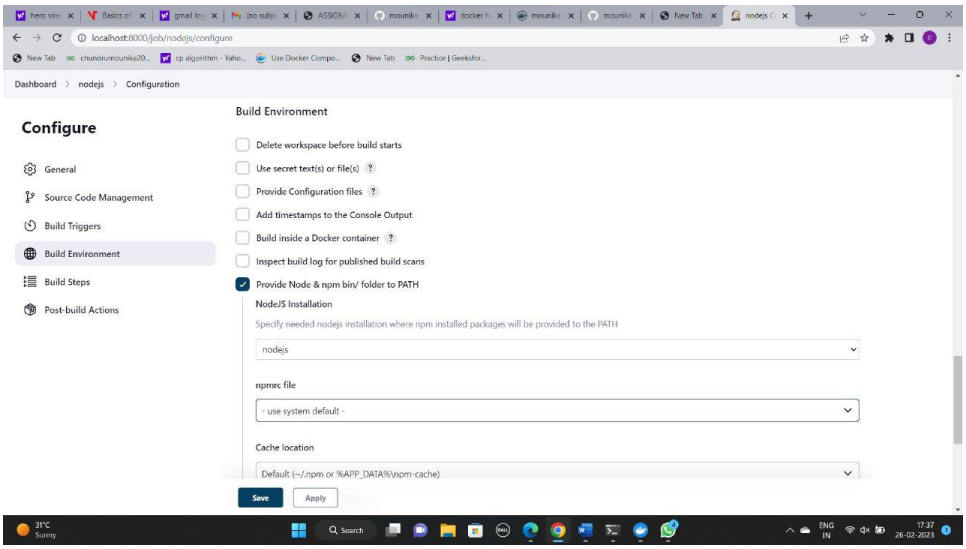
==========

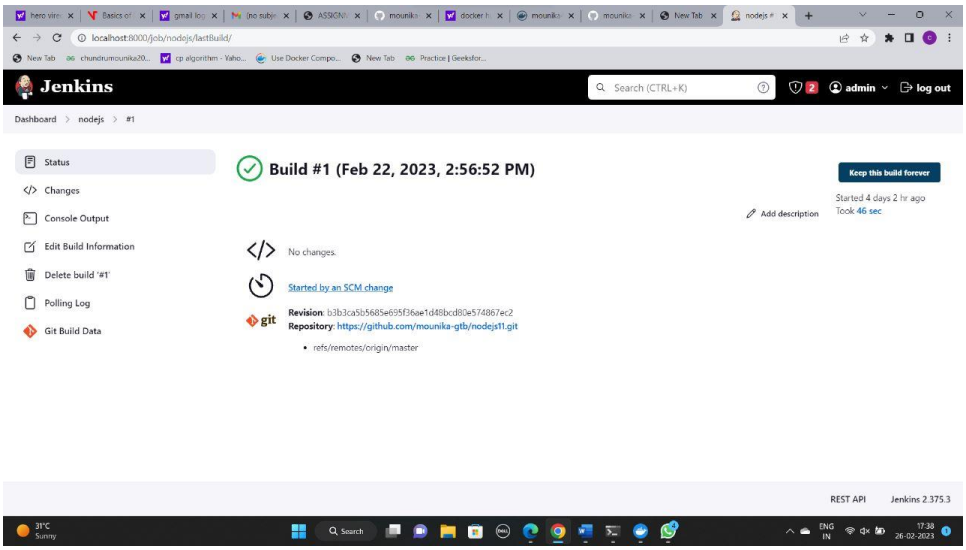
Q2.Create a CI-CD pipeline for a Nodejs Application in jenkins and all the steps involved in it should be given in a screenshot and your jenkins username must be visible in the screenshot.


Dashboard > nodejs > Configuration
Configure
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
General
Enabled
Description
hiii
[Plain text] Preview
Discard old builds
GitHub project
This project is parameterized
Throttle builds
Execute concurrent builds if necessary
Advanced...
Source Code Management
Save
Apply


Dashboard > nodejs > Configuration
Configure
Source Code Management
None
Git
Repositories
Repository URL
https://github.com/mounika-gtb/nodejs11.git
Credentials
mounika-gtb/******
Add
Advanced...
Add Repository
Branches to build
Save
Apply


Dashboard > nodejs > Configuration
Configure
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 5:34:20 pm India Standard Time; would next run at Sunday, 26 February, 2023 at 5:34:20 pm India Standard Time.
Ignore post-commit hooks
Save
Apply





==========

==========

Q3. Create documentation of how you are going to create a CI-CD pipeline for python applications.

In the documentation you have mentioned each step which should be taken to configure the CI-CD pipeline for a python application including the plugins you are using and the global tool configuration.

Steps for creating CI-CD pipeline for python flask in

*Already we have pythonflask folder so open that and clear the path then enter

cmd

*A command prompt will be displayed on the screen so run the basic commands like git init, git status , git add .,git status, git commit -m “learn”

*Then create a repository in the github then copy the link’

*Then go to cmd and enter git remote add origin link

*git push origin master. make sure that our repository is in master or not

* Then open docker desktop and open previous cmd and enter docker login, docker images, docker build -t imagename, docker tag imagename:latest username/repo name, docker push username/repo name

*Open dockerhub.com and check so that our repository will be there

*Open localhost:8082 here 8082 is my expose Then create a new item as NODEJS

*Then click on that go to configure select git in that, enter the repository URL which is in our created repo then select POLL SCM and enter * * * * *

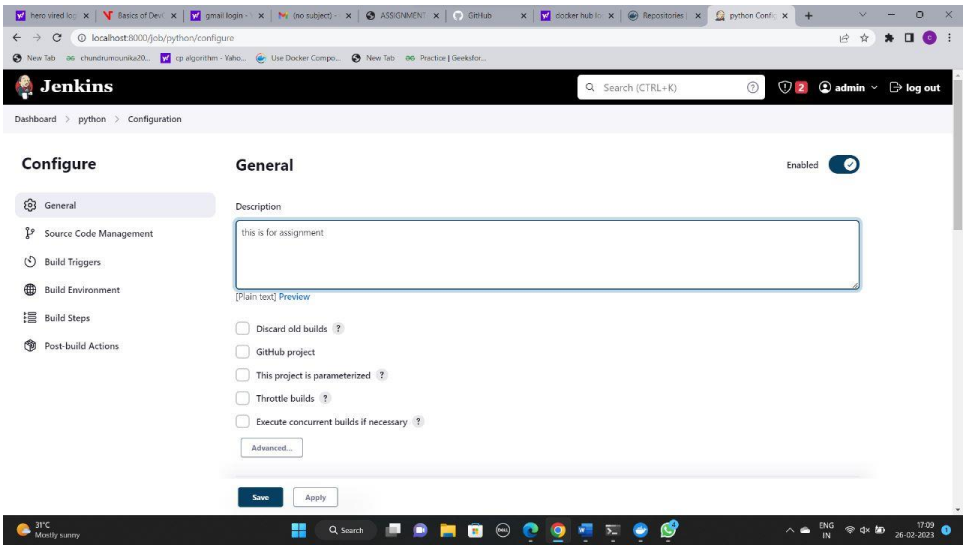
*Next select provide node and NPM bin/folder to path in the build environment

*Meanwhile select the build an publish, reponame, tag as latest

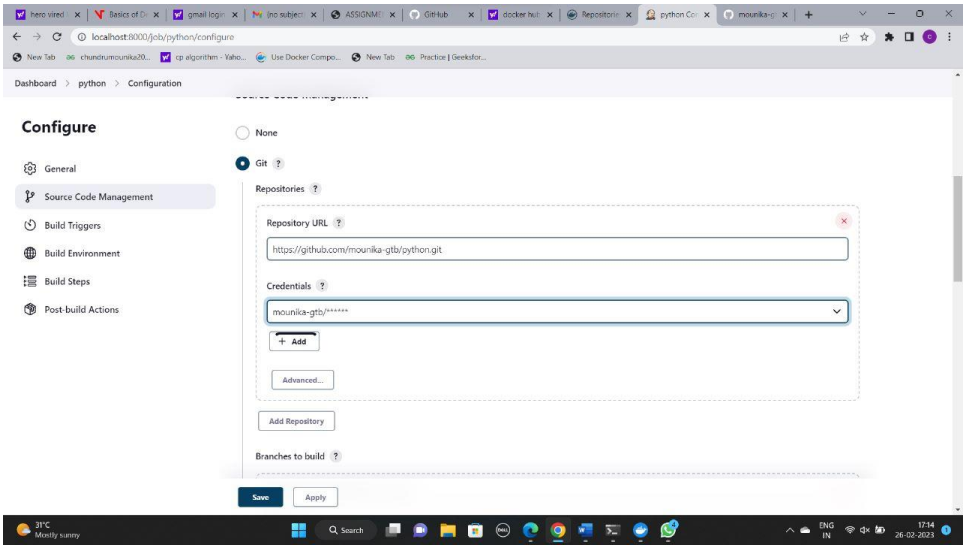
*Then enter registry credentials then apply and save

*Next click on build now then it shows error.so that Go to Dashboard go to manage Jenkins□global tool configuration, enter the path then enter NODEJS application then enter the NODEJS version as 18.10.0 Then it shows the output as BUILD SUCCESS

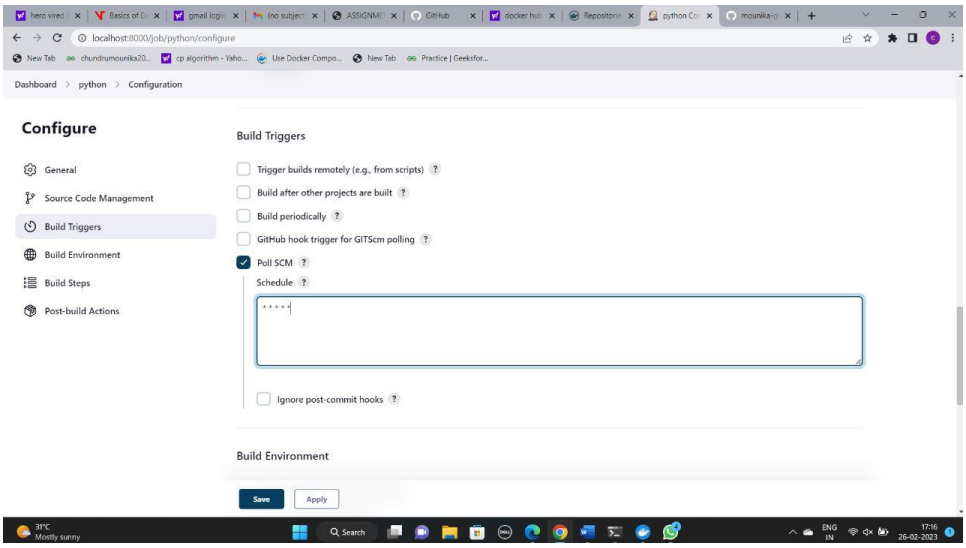
Steps for creating CI-CD pipeline for python flask in zenkins step-1: open zenkins with your credentials and add item give any item name and click free style project and click ok and in general give description



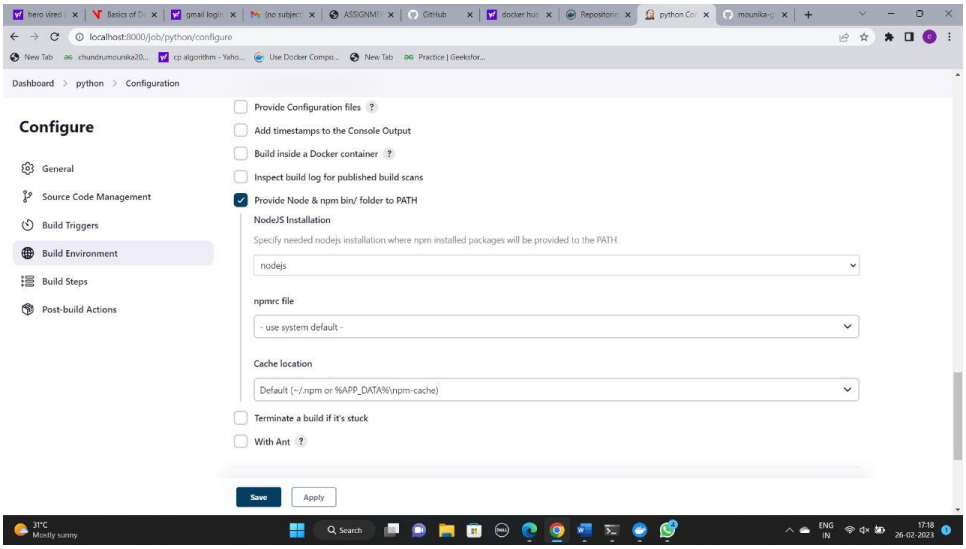
step-2: In sorce code management give the git repository url which contains the pythonflask application and if the repository is public then select crendentials none otherwise click add and zenkins and give your crendentials.

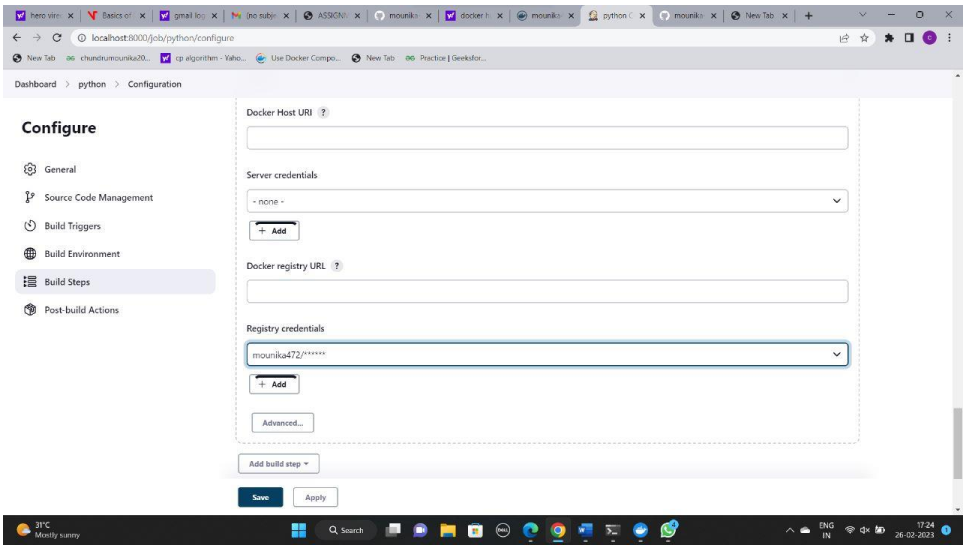


Step-3:In build triggers select poll SCM give * * * * * it means it will observe for every minute

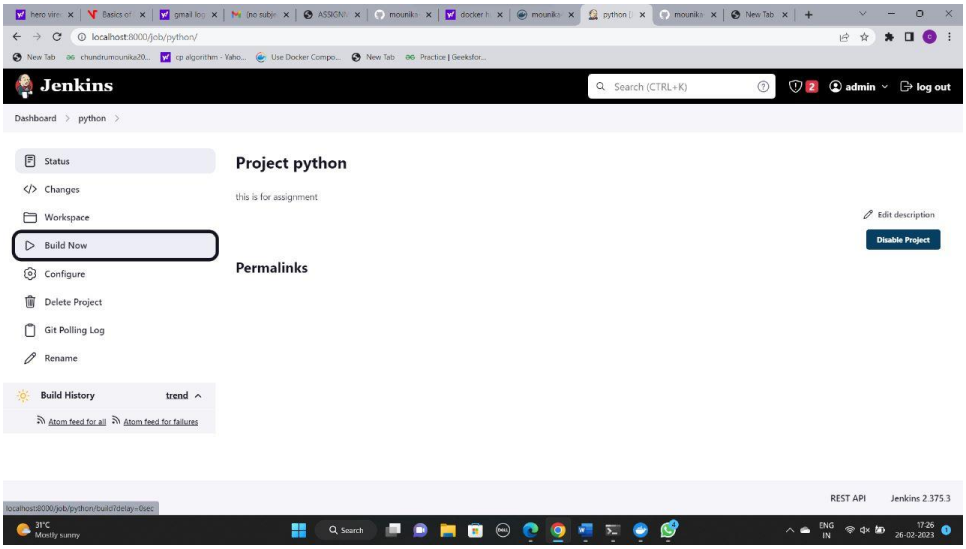


Step-4:In Build Steps, we need to select Docker Build and publish and we need give our dockerhub repository credentials in which docker image and container for pythonflask is present





step-5 : We need to save and apply then select build now and we successfully created CI-CD pipeline for python flask in zenkins



==========
==========