1) 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.

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

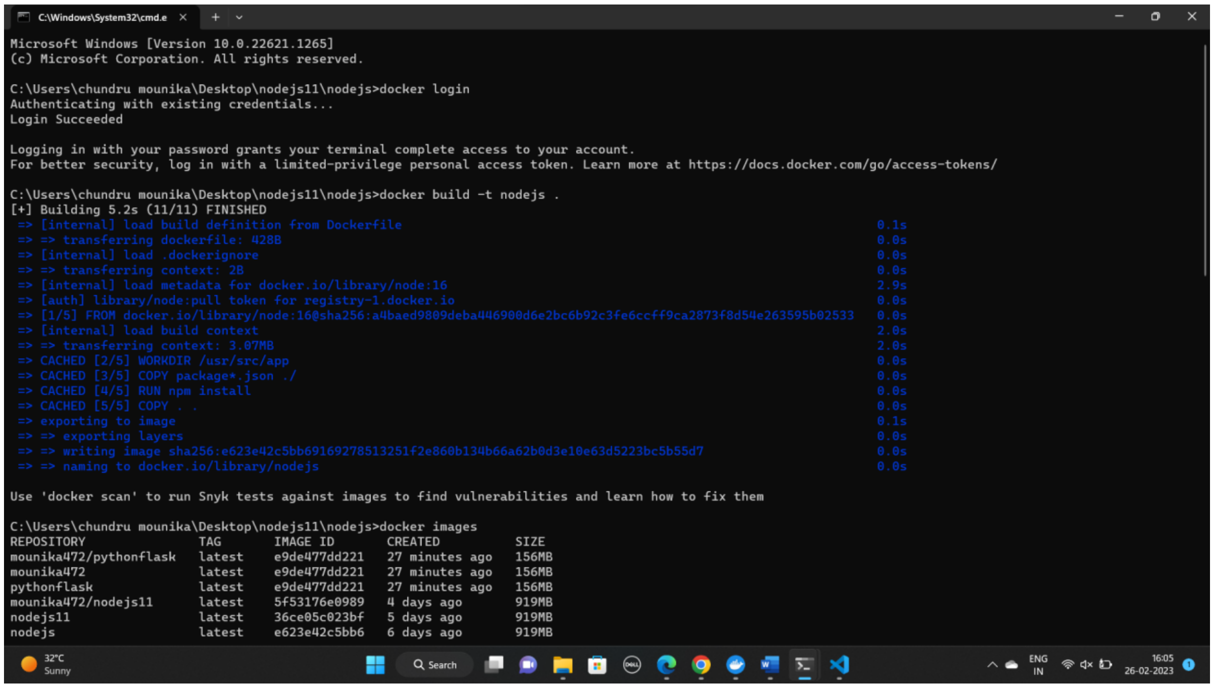
//COPY: Which contains our application dependency and keep it in a desired folder
//EXPOSE: It tells about docker that our application will run on port 3000 inside the docker
//RUN: It tells about docker to perform installization of whatever is there in your dependency file
//CMD: It tells docker to 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 .

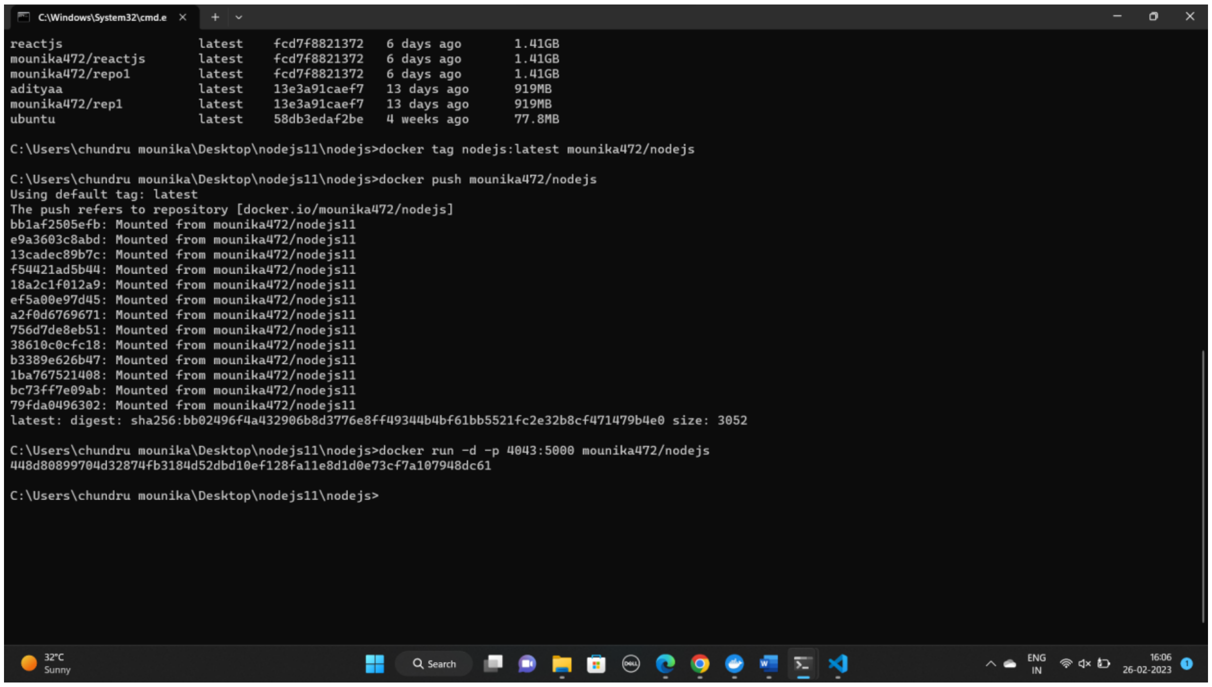
*if we check the docker images it will be displayed the different images of our particular repository
*Then create a repository in our docker hub.com
*Then docker tag image name:latest username/repo name
*Atlast we have to push the image to the docker desktop so that enter the command as docker push 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

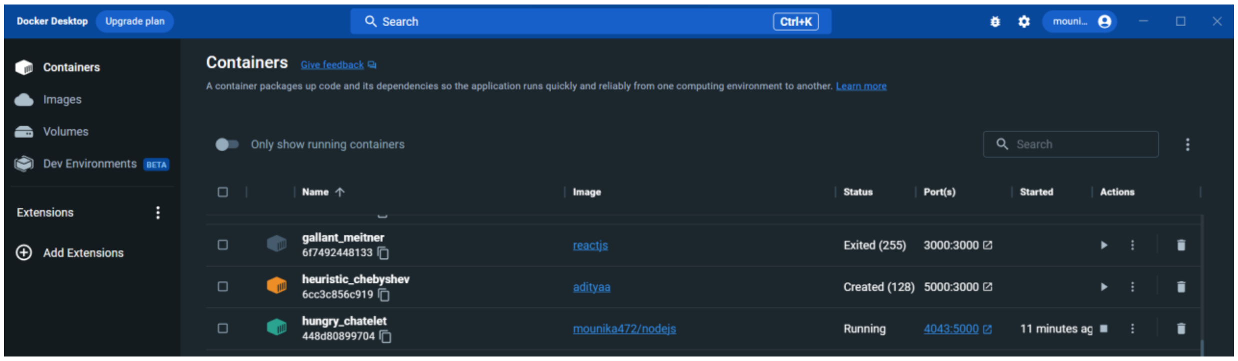
*It shows as output as WELCOME TO FLASK TUTORIALS

NODEJS


```
Microsoft Windows [Version 10.0.22621.1265]
(c) Microsoft Corporation. All rights reserved.

C:\Users\chundru mounika\Desktop\nodejs11\nodejs>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\chundru mounika\Desktop\nodejs11\nodejs>docker build -t nodejs .
[+] Building 5.2s (11/11) FINISHED
 => [internal] load build definition from Dockerfile                                                      0.1s
 => => transferring dockerfile: 428B                                                                      0.0s
 => [internal] load .dockerignore                                                                         0.0s
 => => transferring context: 2B                                                                           0.0s
 => [internal] load metadata for docker.io/library/node:16                                                2.9s
 => [auth] library/node:pull token for registry-1.docker.io                                               0.0s
 => [1/5] FROM docker.io/library/node:16@sha256:a4baed9809deba446900d6e2bc6b92c3fe6ccff9ca2873f8d54e263595b02533   0.0s
 => [internal] load build context                                                                         2.0s
 => => transferring context: 3.07MB                                                                       2.0s
 => CACHED [2/5] WORKDIR /usr/src/app                                                                     0.0s
 => CACHED [3/5] COPY package*.json ./                                                                    0.0s
 => CACHED [4/5] RUN npm install                                                                          0.0s
 => CACHED [5/5] COPY . .                                                                                 0.0s
 => exporting to image                                                                                    0.1s
 => => exporting layers                                                                                   0.0s
 => => writing image sha256:e623e42c5bb69169278513251f2e860b134b66a62b0d3e10e63d5223bc5b55d7              0.0s
 => => naming to docker.io/library/nodejs                                                                 0.0s

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

C:\Users\chundru mounika\Desktop\nodejs11\nodejs>docker images
REPOSITORY                 TAG       IMAGE ID       CREATED          SIZE
mounika472/pythonflask     latest    e9de477dd221   27 minutes ago   156MB
mounika472                 latest    e9de477dd221   27 minutes ago   156MB
pythonflask                latest    e9de477dd221   27 minutes ago   156MB
mounika472/nodejs11        latest    5f53176e0989   4 days ago       919MB
nodejs11                   latest    36ce05c023bf   5 days ago       919MB
nodejs                     latest    e623e42c5bb6   6 days ago       919MB
```



```
reactjs                 latest    fcd7f8821372   6 days ago       1.41GB
mounika472/reactjs      latest    fcd7f8821372   6 days ago       1.41GB
mounika472/repo1        latest    fcd7f8821372   6 days ago       1.41GB
adityaa                 latest    13e3a91caef7   13 days ago      919MB
mounika472/rep1         latest    13e3a91caef7   13 days ago      919MB
ubuntu                  latest    58db3edaf2be   4 weeks ago      77.8MB

C:\Users\chundru mounika\Desktop\nodejs11\nodejs>docker tag nodejs:latest mounika472/nodejs

C:\Users\chundru mounika\Desktop\nodejs11\nodejs>docker push mounika472/nodejs
Using default tag: latest
The push refers to repository [docker.io/mounika472/nodejs]
bb1af2505efb: Mounted from mounika472/nodejs11
e9a3603c8abd: Mounted from mounika472/nodejs11
13cadec89b7c: Mounted from mounika472/nodejs11
f54421ad5b44: Mounted from mounika472/nodejs11
18a2c1f012a9: Mounted from mounika472/nodejs11
ef5a00e97d45: Mounted from mounika472/nodejs11
a2f0d6769671: Mounted from mounika472/nodejs11
756d7de8eb51: Mounted from mounika472/nodejs11
38610c0cfc18: Mounted from mounika472/nodejs11
b3389e626b47: Mounted from mounika472/nodejs11
1ba767521408: Mounted from mounika472/nodejs11
bc73ff7e09ab: Mounted from mounika472/nodejs11
79fda0496302: Mounted from mounika472/nodejs11
latest: digest: sha256:bb02496f4a432906b8d3776e8ff49344b4bf61bb5521fc2e32b8cf471479b4e0 size: 3052

C:\Users\chundru mounika\Desktop\nodejs11\nodejs>docker run -d -p 4043:5000 mounika472/nodejs
448d80899704d32874fb3184d52dbd10ef128fa11e8d1d0e73cf7a107948dc61

C:\Users\chundru mounika\Desktop\nodejs11\nodejs>
```

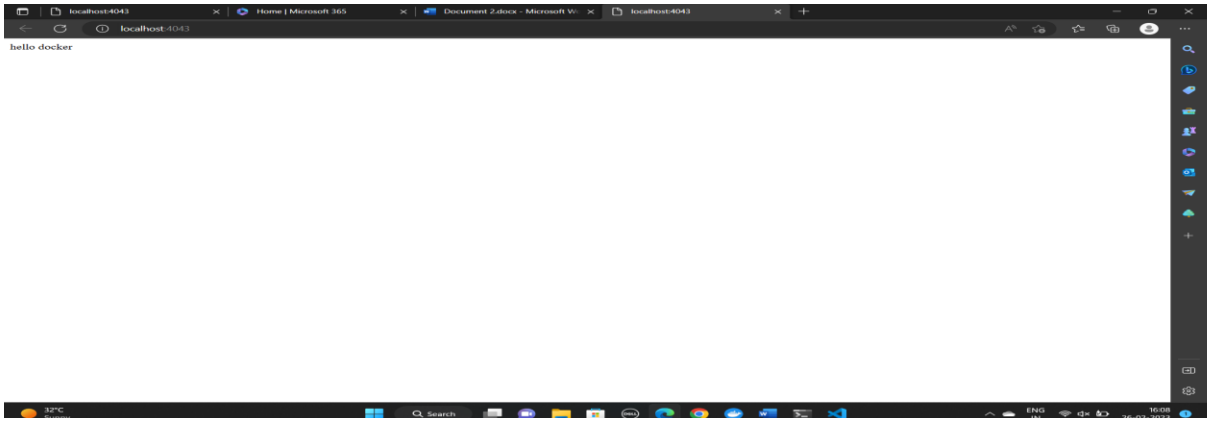


Containers

A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. Learn more

Only show running containers

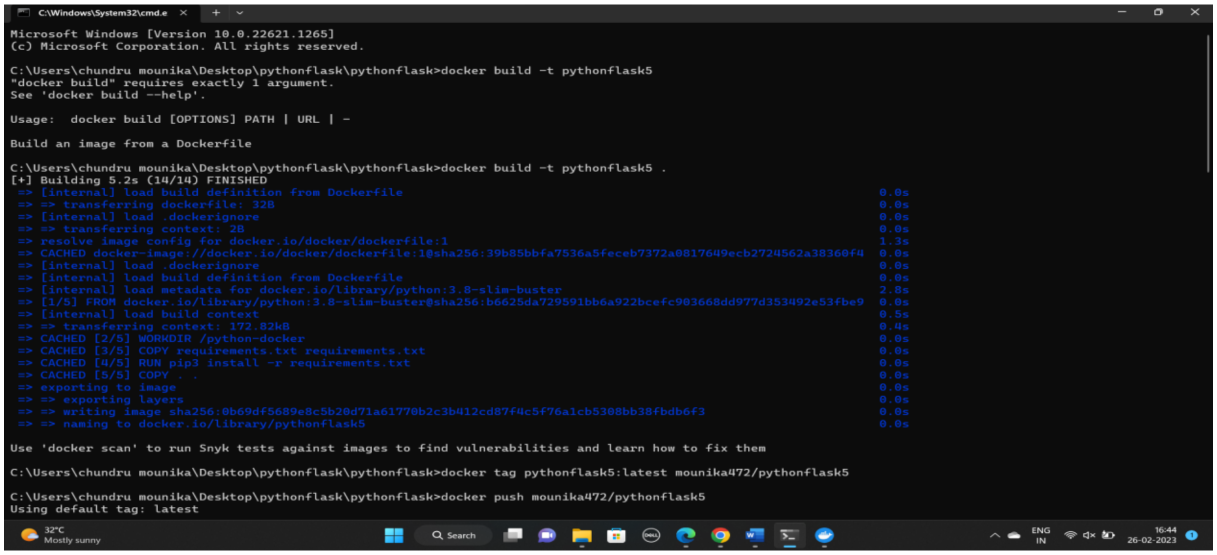
Name	Image	Status	Port(s)	Started
gallant_meitner 6f7492448133	reactjs	Exited (255)	3000:3000	
heuristic_chebyshev 6cc3c856c919	adityaa	Created (128)	5000:3000	
hungry_chatelet 448d80899704	mounika472/nodejs	Running	4043:5000	11 minutes ag



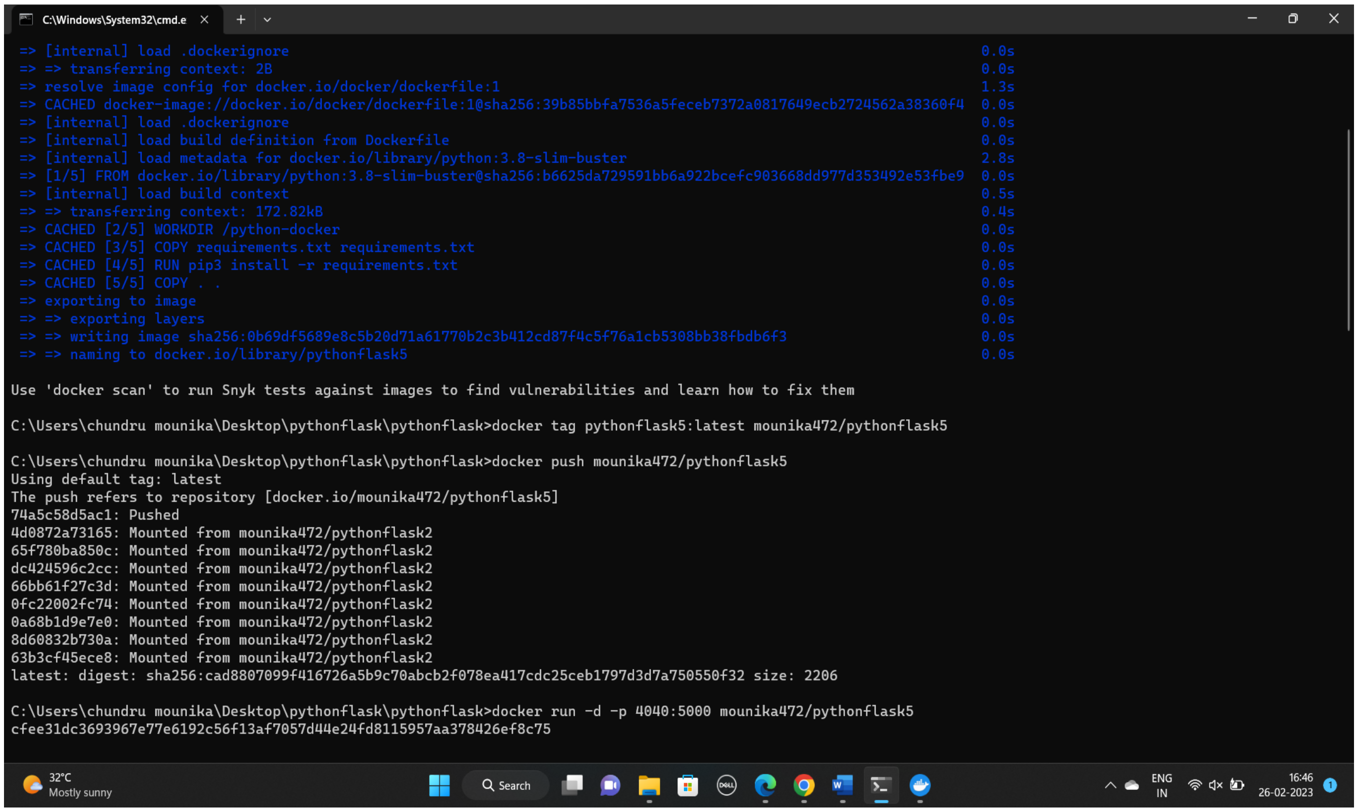
localhost:4043

hello docker

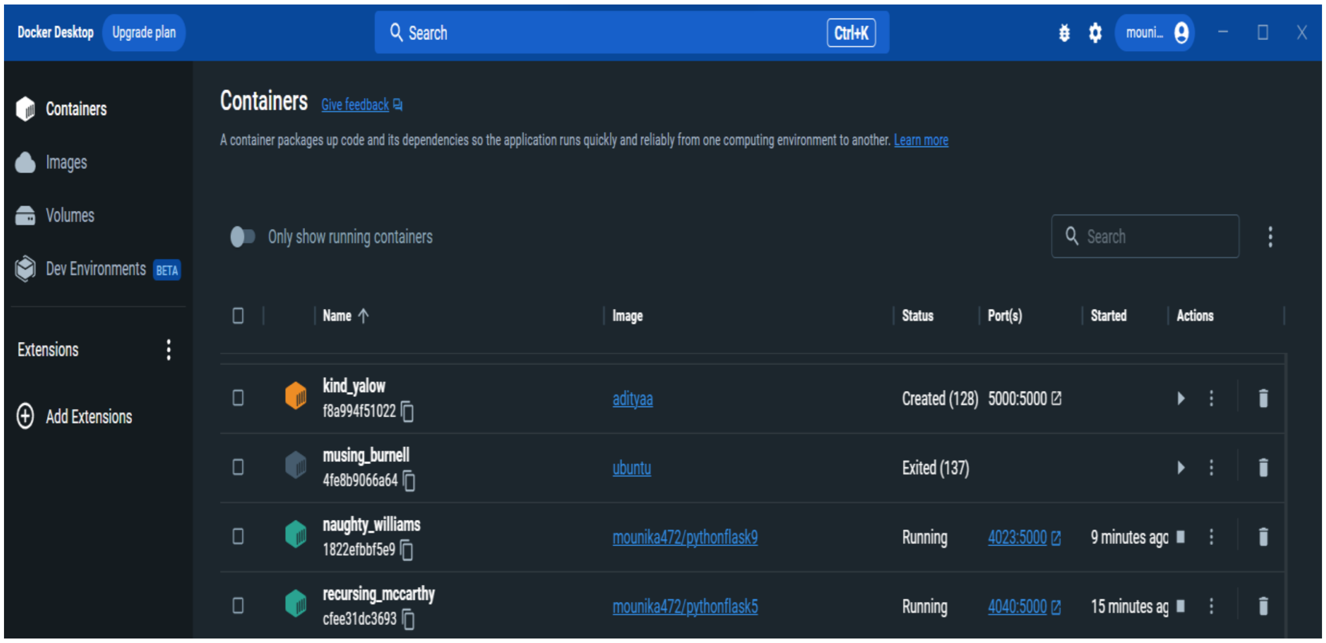
python flask



```
Microsoft Windows [Version 10.0.22621.1265]
(c) Microsoft Corporation. All rights reserved.

C:\Users\chundru mounika\Desktop\pythonflask\pythonflask>docker build -t pythonflask5
"docker build" requires exactly 1 argument.
See 'docker build --help'.

Usage:  docker build [OPTIONS] PATH | URL | -

Build an image from a Dockerfile

C:\Users\chundru mounika\Desktop\pythonflask\pythonflask>docker build -t pythonflask5 .
[+] Building 5.2s (14/14) FINISHED
 => [internal] load build definition from Dockerfile                                                          0.0s
 => => transferring dockerfile: 32B                                                                           0.0s
 => [internal] load .dockerignore                                                                             0.0s
 => => transferring context: 2B                                                                               0.0s
 => resolve image config for docker.io/docker/dockerfile:1                                                    1.3s
 => CACHED docker-image://docker.io/docker/dockerfile:1@sha256:39b85bbfa7536a5feceb7372a0817649ecb2724562a38360f4  0.0s
 => [internal] load .dockerignore                                                                             0.0s
 => [internal] load build definition from Dockerfile                                                          0.0s
 => [internal] load metadata for docker.io/library/python:3.8-slim-buster                                     2.8s
 => [1/5] FROM docker.io/library/python:3.8-slim-buster@sha256:b6625da729591bb6a922bcefc903668dd977d353492e53fbe9  0.0s
 => [internal] load build context                                                                             0.5s
 => => transferring context: 172.82kB                                                                         0.4s
 => 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 sha256:0b69df5689e8c5b20d71a61770b2c3b412cd87f4c5f76a1cb5308bb38fbdb6f3                  0.0s
 => => naming to docker.io/library/pythonflask5                                                               0.0s

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

C:\Users\chundru mounika\Desktop\pythonflask\pythonflask>docker tag pythonflask5:latest mounika472/pythonflask5

C:\Users\chundru mounika\Desktop\pythonflask\pythonflask>docker push mounika472/pythonflask5
Using default tag: latest
```



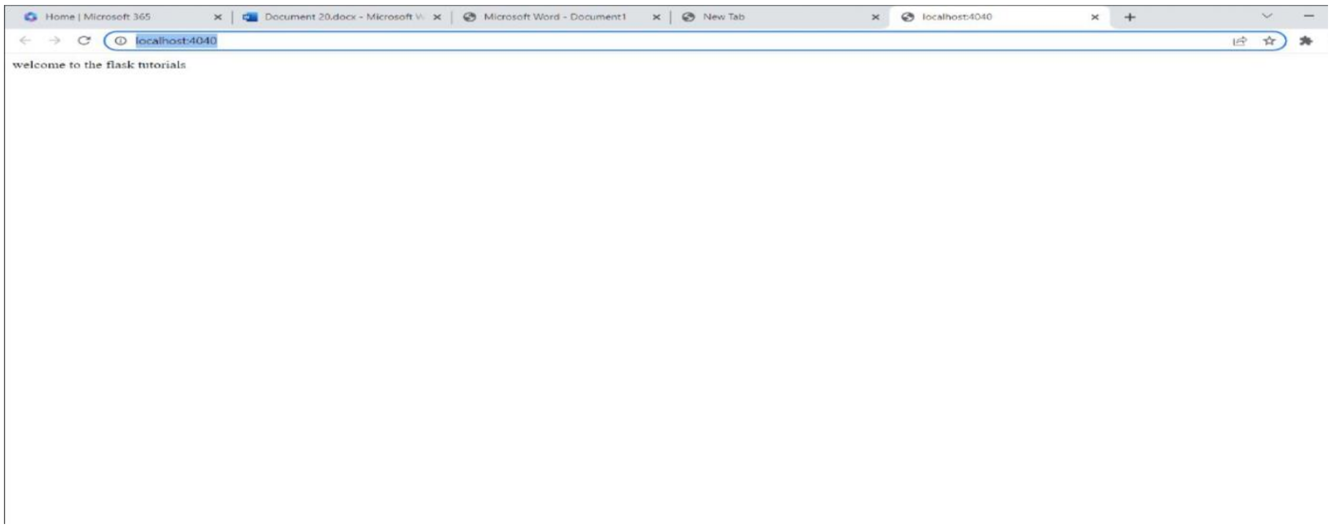
```
 => [internal] load .dockerignore                                                                             0.0s
 => => transferring context: 2B                                                                               0.0s
 => resolve image config for docker.io/docker/dockerfile:1                                                    1.3s
 => CACHED docker-image://docker.io/docker/dockerfile:1@sha256:39b85bbfa7536a5feceb7372a0817649ecb2724562a38360f4  0.0s
 => [internal] load .dockerignore                                                                             0.0s
 => [internal] load build definition from Dockerfile                                                          0.0s
 => [internal] load metadata for docker.io/library/python:3.8-slim-buster                                     2.8s
 => [1/5] FROM docker.io/library/python:3.8-slim-buster@sha256:b6625da729591bb6a922bcefc903668dd977d353492e53fbe9  0.0s
 => [internal] load build context                                                                             0.5s
 => => transferring context: 172.82kB                                                                         0.4s
 => 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 sha256:0b69df5689e8c5b20d71a61770b2c3b412cd87f4c5f76a1cb5308bb38fbdb6f3                  0.0s
 => => naming to docker.io/library/pythonflask5                                                               0.0s

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

C:\Users\chundru mounika\Desktop\pythonflask\pythonflask>docker tag pythonflask5:latest mounika472/pythonflask5

C:\Users\chundru mounika\Desktop\pythonflask\pythonflask>docker push mounika472/pythonflask5
Using default tag: latest
The push refers to repository [docker.io/mounika472/pythonflask5]
74a5c58d5ac1: Pushed
4d0872a73165: Mounted from mounika472/pythonflask2
65f780ba850c: Mounted from mounika472/pythonflask2
dc424596c2cc: Mounted from mounika472/pythonflask2
66bb61f27c3d: Mounted from mounika472/pythonflask2
0fc22002fc74: Mounted from mounika472/pythonflask2
0a68b1d9e7e0: Mounted from mounika472/pythonflask2
8d60832b730a: Mounted from mounika472/pythonflask2
63b3cf45ece8: Mounted from mounika472/pythonflask2
latest: digest: sha256:cad8807099f416726a5b9c70abcb2f078ea417cdc25ceb1797d3d7a750550f32 size: 2206

C:\Users\chundru mounika\Desktop\pythonflask\pythonflask>docker run -d -p 4040:5000 mounika472/pythonflask5
cfee31dc3693967e77e6192c56f13af7057d44e24fd8115957aa378426ef8c75
```



Containers

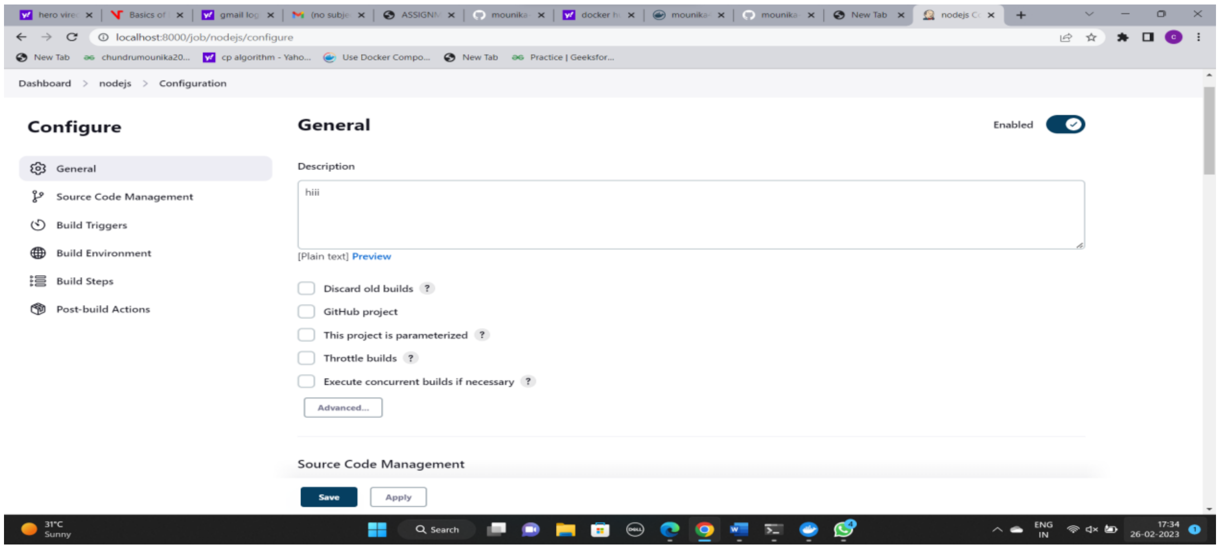
A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. Learn more

Name	Image	Status	Port(s)	Started	Actions
kind_yalow f8a994f51022	adityaa	Created (128)	5000:5000		
musing_burnell 4fe8b9066a64	ubuntu	Exited (137)			
naughty_williams 1822efbbf5e9	mounika472/pythonflask9	Running	4023:5000	9 minutes ago	
recursing_mccarthy cfee31dc3693	mounika472/pythonflask5	Running	4040:5000	15 minutes ag	



```
welcome to the flask tutorials
```

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.


localhost:8000/job/nodejs/configure

Dashboard > nodejs > Configuration

Configure

General

Enabled

Description

hiii

[Plain text] Preview

Discard old builds

GitHub project

This project is parameterized

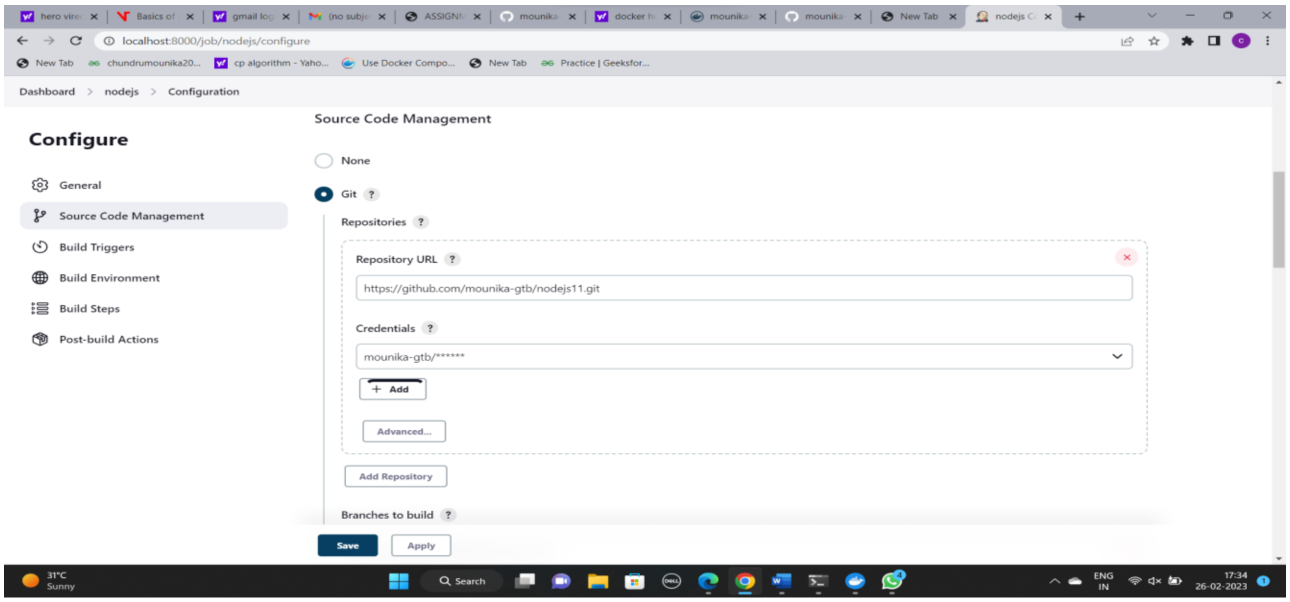
Throttle builds

Execute concurrent builds if necessary

Advanced...

Source Code Management

Save Apply


localhost:8000/job/nodejs/configure

Dashboard > nodejs > Configuration

Configure

Source Code Management

None

[✓ Git]

Repositories

Repository URL

https://github.com/mounika-gtb/nodejs11.git

Credentials

mounika-gtb/******

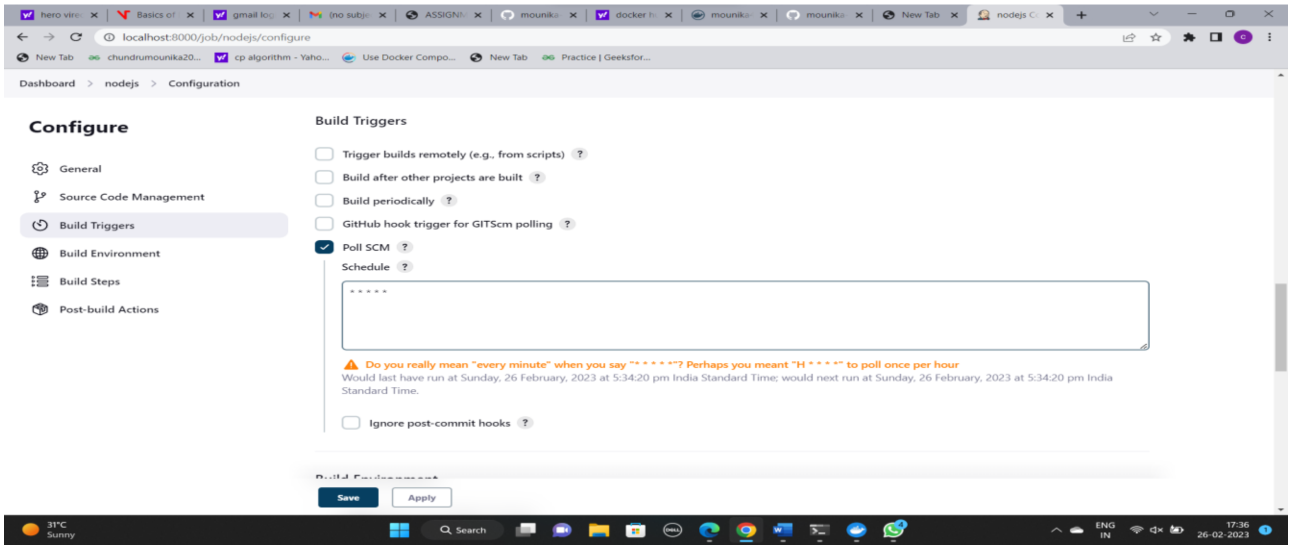
Add

Advanced...

Add Repository

Branches to build

Save Apply



Dashboard > nodejs > Configuration

Configure

- 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
- [x] Poll SCM

Schedule

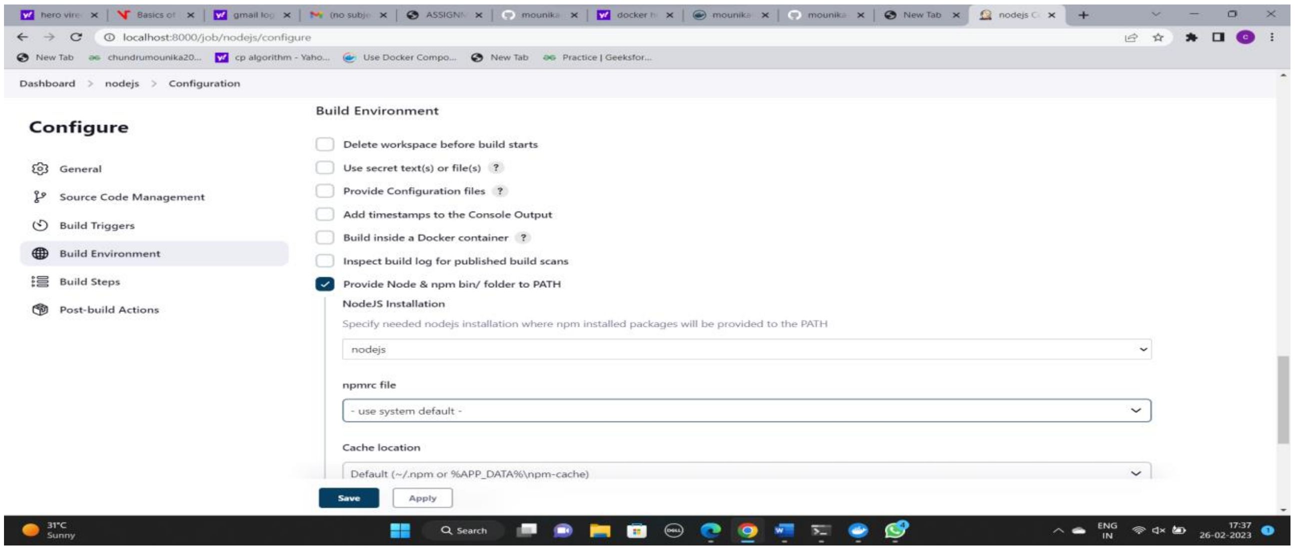
```
* * * * *
```

⚠ 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



Dashboard > nodejs > Configuration

Configure

Build Environment

- [ ] Delete workspace before build starts
- [ ] Use secret text(s) or file(s)
- [ ] Provide Configuration files
- [ ] Add timestamps to the Console Output
- [ ] Build inside a Docker container
- [ ] Inspect build log for published build scans
- [x] Provide Node & npm bin/ folder to PATH

NodeJS Installation

Specify needed nodejs installation where npm installed packages will be provided to the PATH

nodejs

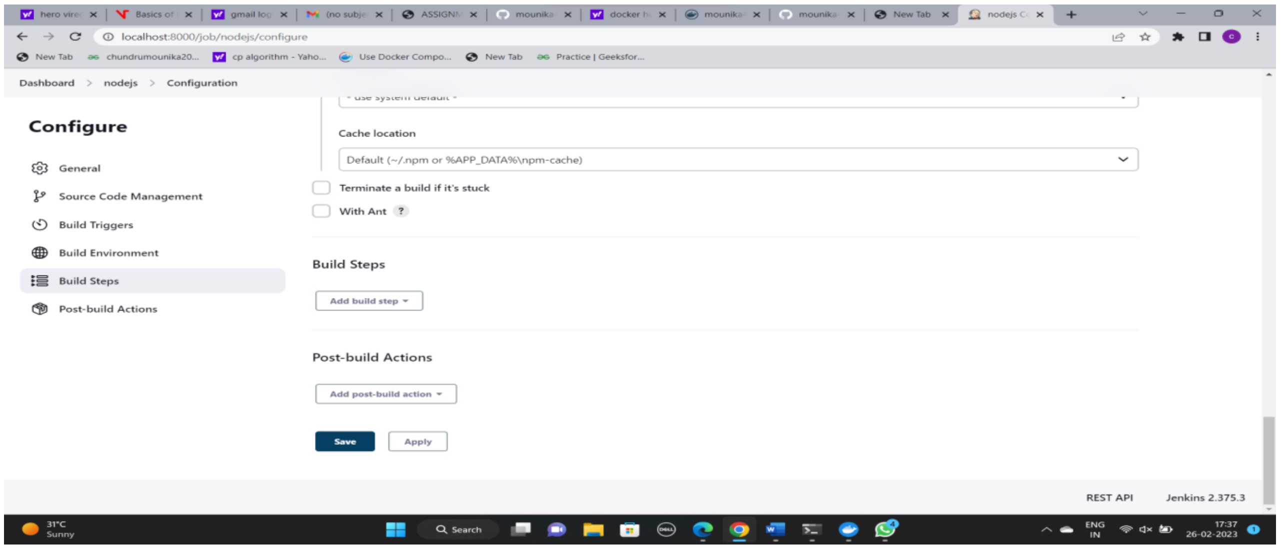
npmrc file

- use system default -

Cache location

Default (~/.npm or %APP_DATA%\npm-cache)

Save Apply



Dashboard > nodejs > Configuration

Configure

Cache location

Default (~/.npm or %APP_DATA%\npm-cache)

- [ ] Terminate a build if it's stuck
- [ ] With Ant

Build Steps

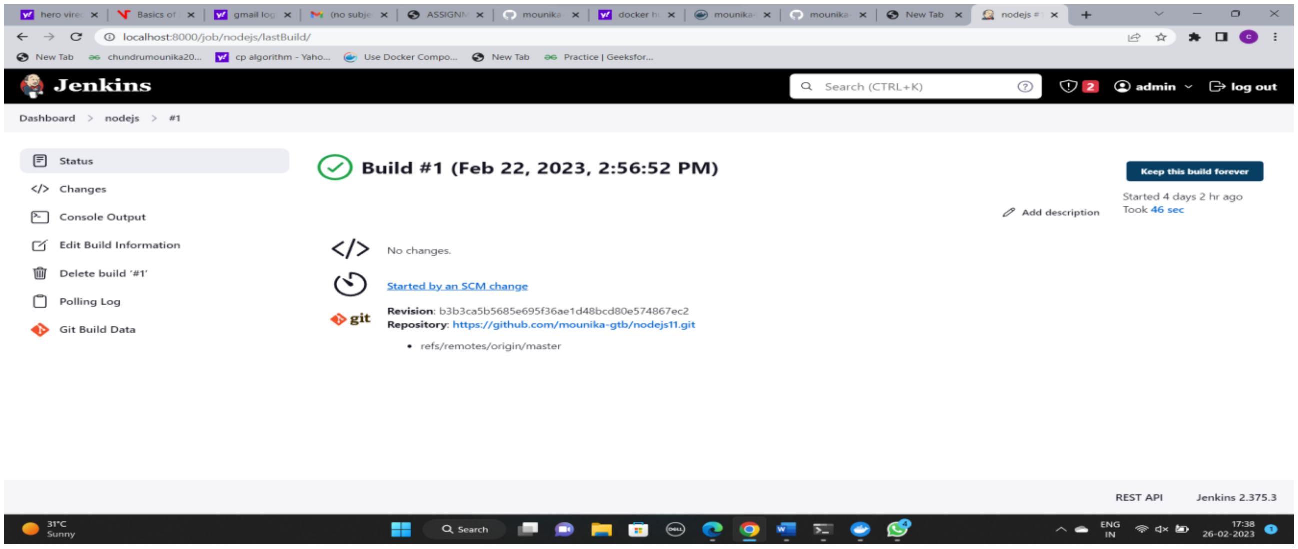
Add build step

Post-build Actions

Add post-build action

Save Apply

REST API Jenkins 2.375.3


Build #1 (Feb 22, 2023, 2:56:52 PM)

No changes.

Started by an SCM change

Revision: b3b3ca5b5685e695f36ae1d48bcd80e574867ec2
Repository: https://github.com/mounika-gtb/nodejs11.git

- refs/remotes/origin/master

Started 4 days 2 hr ago
Took 46 sec

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

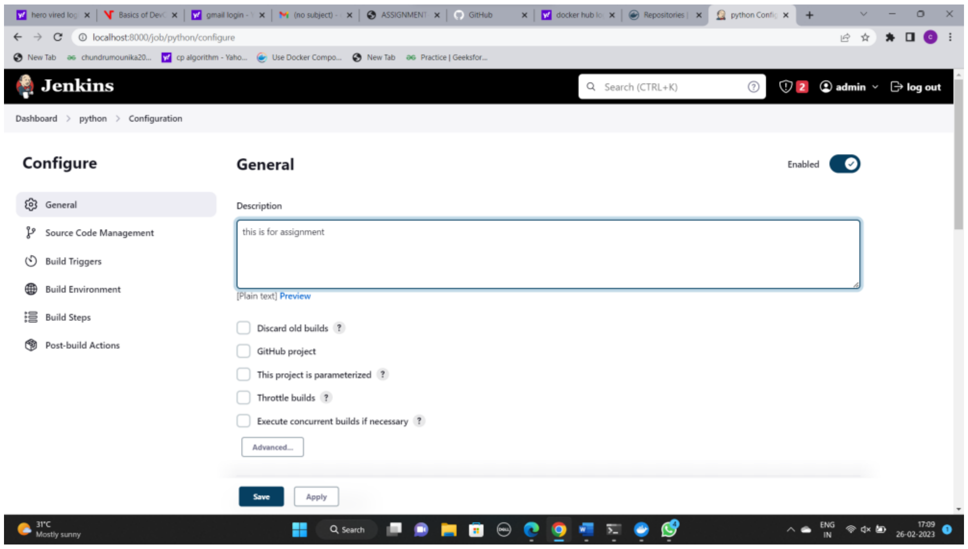
In the documentation you have mentioned each step which should be taken to configure the CI-CD pipeline for a python application including the plugins you are using and the global tool configuration.

Steps for creating CI-CD pipeline for python flask in 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



General

Enabled

Description

this is for assignment

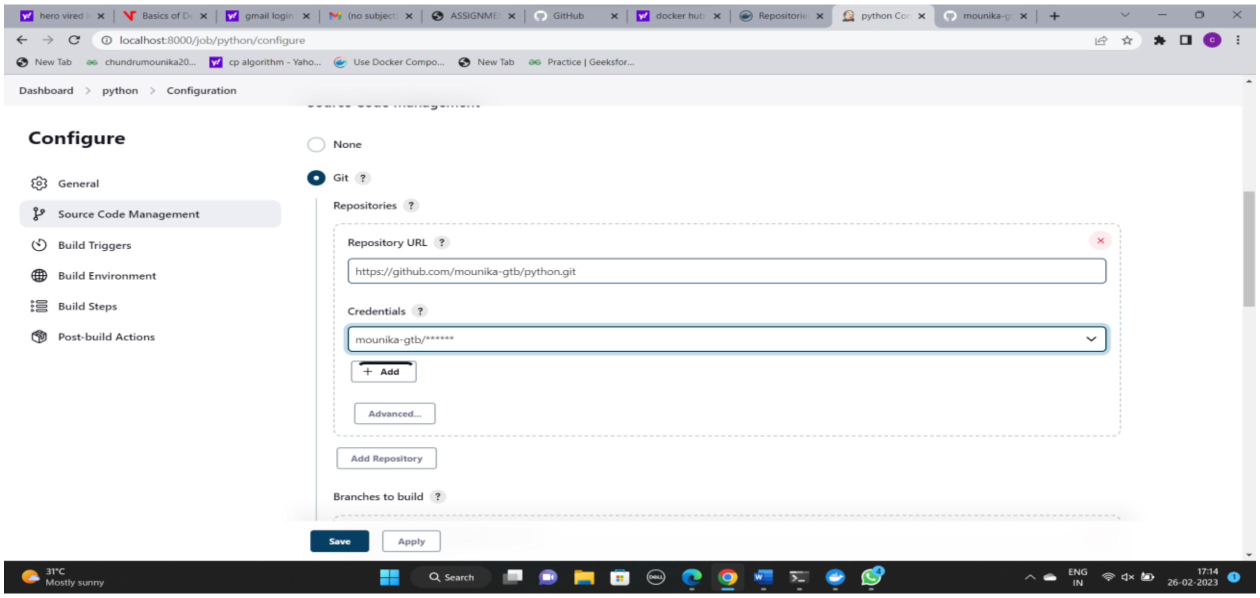
[Plain text] Preview

- [ ] Discard old builds
- [ ] GitHub project
- [ ] This project is parameterized
- [ ] Throttle builds
- [ ] Execute concurrent builds if necessary

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.



None [✓ Git]

Repositories

Repository URL

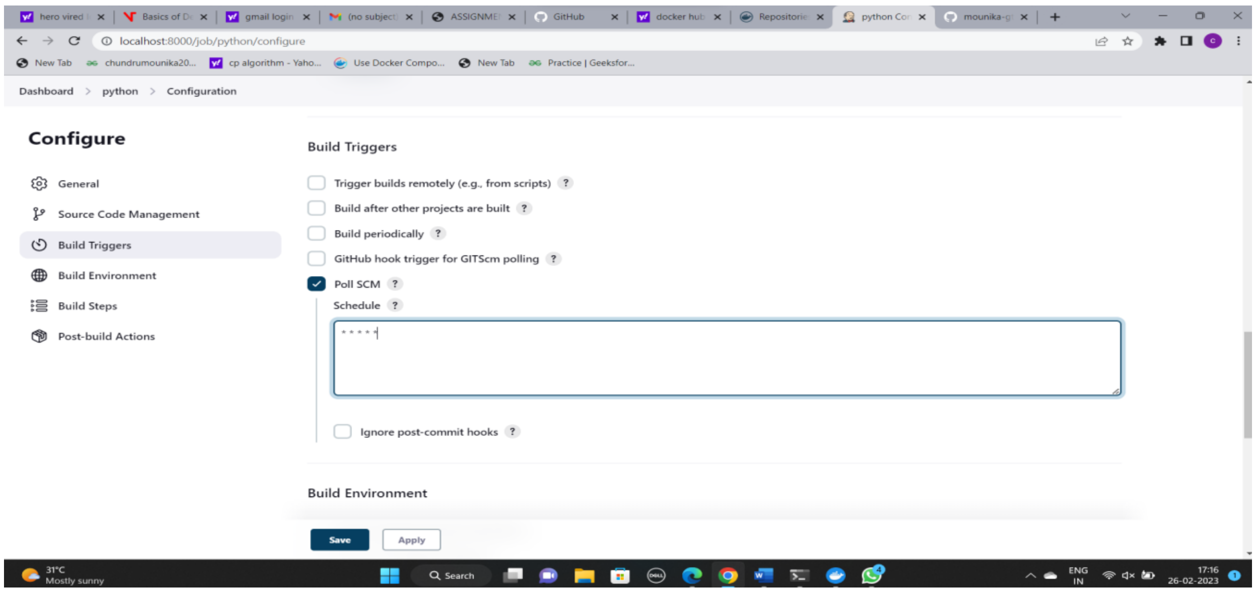
https://github.com/mounika-gtb/python.git

Credentials

mounika-gtb/******

Branches to build

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



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
- [x] Poll SCM

Schedule

```
* * * * *
```

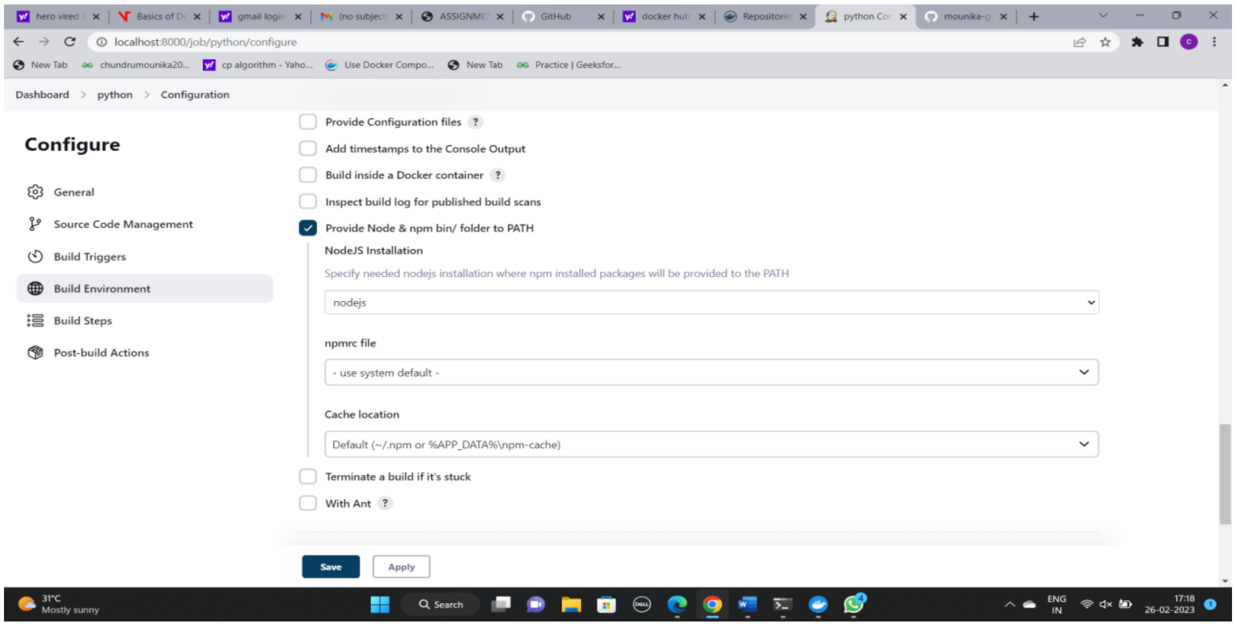
- [ ] Ignore post-commit hooks

Build Environment

Save Apply

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 python flask is present



Dashboard > python > Configuration

Configure

General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions

- [ ] Provide Configuration files
- [ ] Add timestamps to the Console Output
- [ ] Build inside a Docker container
- [ ] Inspect build log for published build scans
- [x] Provide Node & npm bin/ folder to PATH

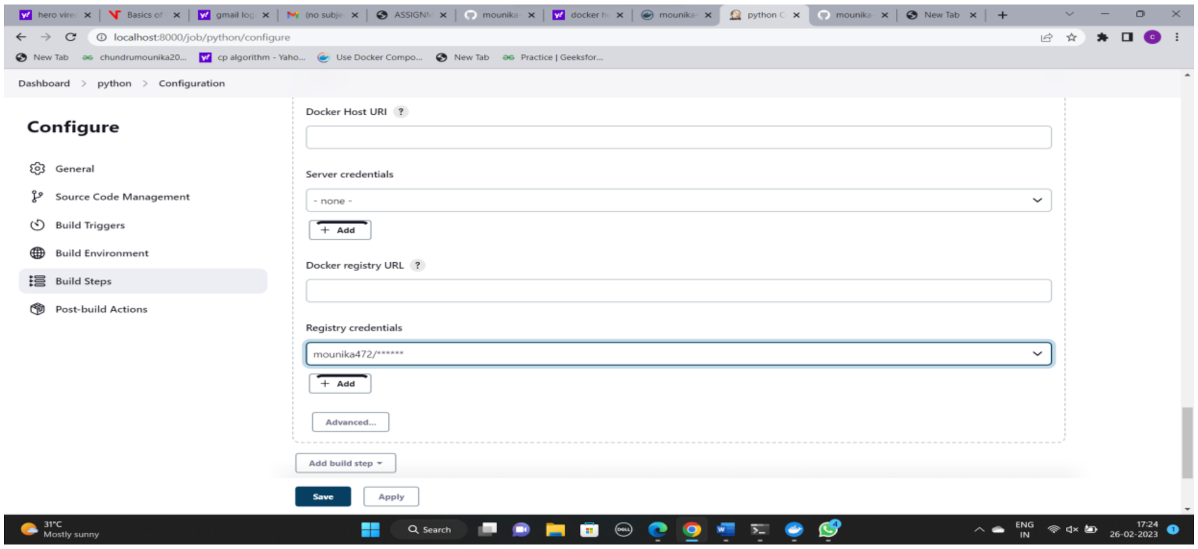
NodeJS Installation
Specify needed nodejs installation where npm installed packages will be provided to the PATH
nodejs

npmrc file
- use system default -

Cache location
Default (~/.npm or %APP_DATA%\npm-cache)

- [ ] Terminate a build if it's stuck
- [ ] With Ant

Save Apply



Docker Host URI

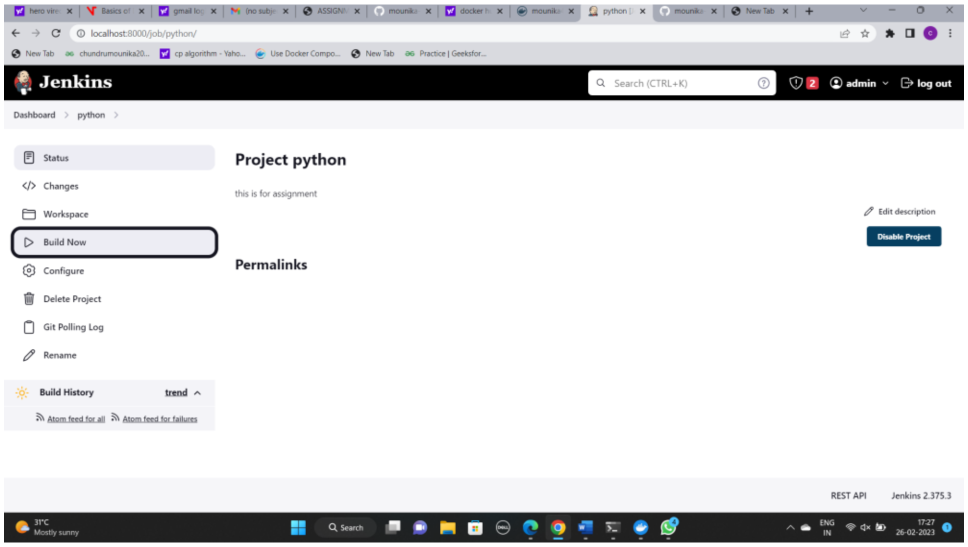
Server credentials: - none -

Docker registry URL

Registry credentials: mounika472/******

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

for python flask in zenkins



Project python

this is for assignment

Permalinks

*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 pythonflask application then enter the pythonflask version as 18.10.0 Then it shows the output

as BUILD SUCCESS