

ASSIGNMENT – 4

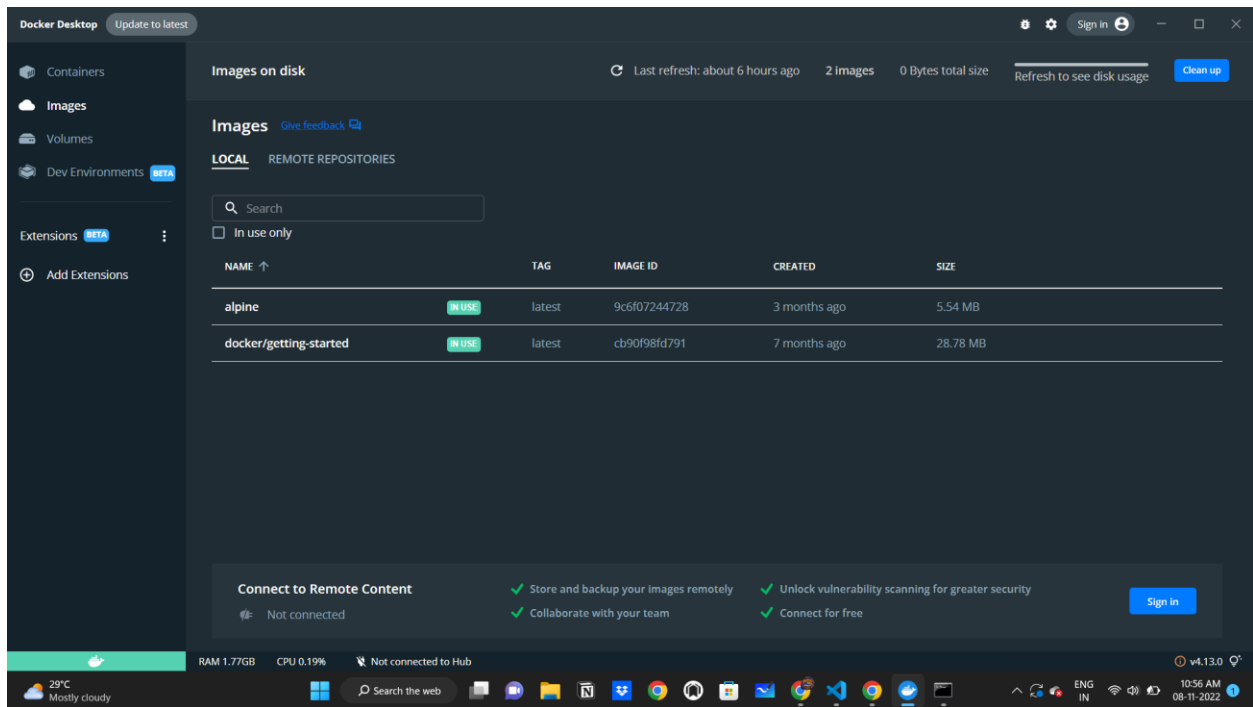
PLASMA DONOR APPLICATION

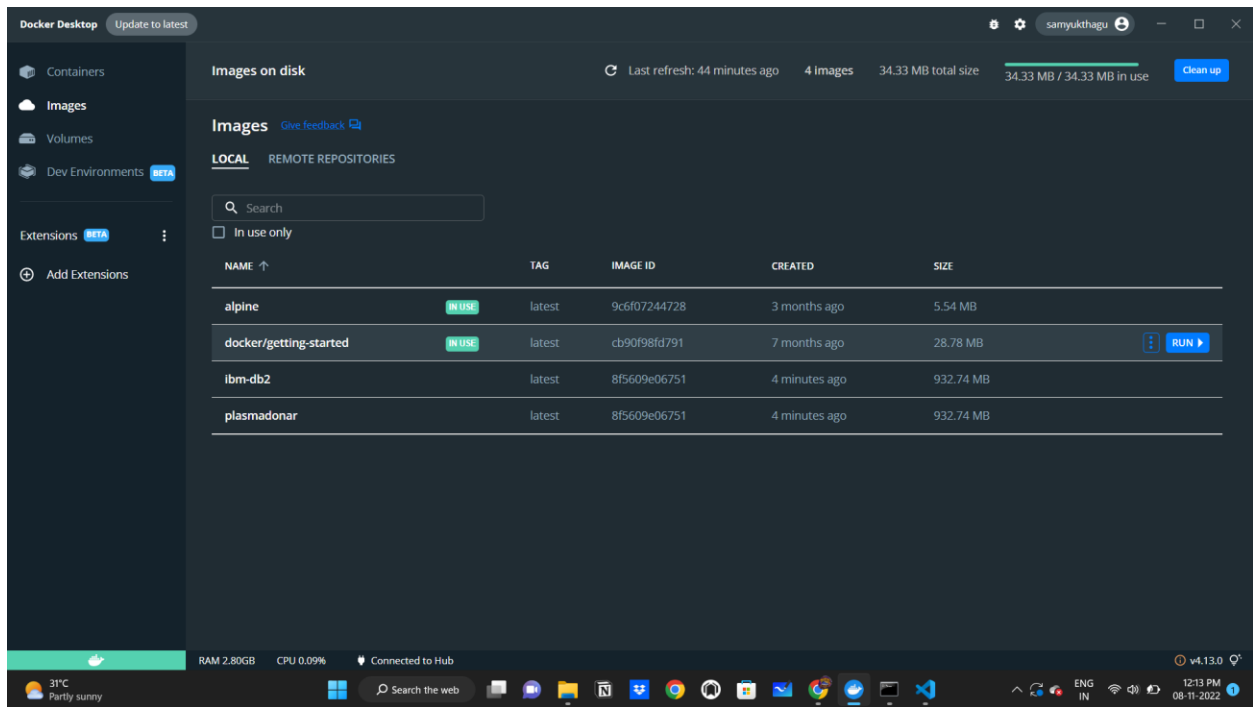
| | |
|-----------------|---------------|
| Date | 08/11/2022 |
| Student Name | Samyuktha G U |
| Student Roll No | 820319104034 |
| Max Mark | |

Questions :

- 1.Pull an Image from docker hub and run it in docker playground.
- 2.Create a docker file for the job portal application and deploy it in Docker desktop application.
- 3.Create a IBM container registry and deploy hello world app or job portal app.
- 4.Create a Kubernetes cluster in IBM cloud and deploy hello world image or job portal image and also expose the same app to run in node port.

Pull an image from docker hub :





Coding for pull an image in docker hub :

```
Microsoft Windows [Version 10.0.22000.1098]
(c) Microsoft Corporation. All rights reserved.

C:\Users\parva>docker run -d -p 80:80 docker/getting-started
Unable to find image 'docker/getting-started:latest' locally
latest: Pulling from docker/getting-started
d99b388f04e: Pull complete
5867c8a5fcd: Pull complete
4b639e5c3b3: Pull complete
061ed9e2b976: Pull complete
bc19f3e8eeb1: Pull complete
4071be97c256: Pull complete
79f586f1a5db: Pull complete
0c9732f525d6: Pull complete
Digest: sha256:b558be874169471bd4e65bd6eac8c303b271a7ee8553ba47481b73b2bf597aae
Status: Downloaded newer image for docker/getting-started:latest
ac9ce08124cf2d8e433bb14de75ccd79902b99ad5c4d680bc90513741b65f26b

C:\Users\parva>docker container run alphino echo "Hello world"
Unable to find image 'alphino:latest' locally
docker: Error response from daemon: pull access denied for alphino, repository does not exist or may require 'docker login': denied: requested access to the resource is denied.
See 'docker run --help'.

C:\Users\parva>docker container run alpine echo "Hello world"
Unable to find image 'alpine:latest' locally
latest: Pulling from library/alpine
213ec9aee27d: Pull complete
Digest: sha256:bc41182d7ef5fffc53a40b8d4e725193bc10142a1243f395ee852a8d9738fc2ad
Status: Downloaded newer image for alpine:latest
Hello world

C:\Users\parva>cd..

C:\Users>docker run -d -p 80:80 docker/getting-started
ad4e6cfad09566c857edc768b731cf8fc64dc3fc7cbc64bc0124e5fab383b4ec
docker: Error response from daemon: driver failed programming external connectivity on endpoint sweet_vaughan (e2fee54ddc66e45a890e8f1189f79c70ac1ed4e5f1da6a0c42d373487ae2dba1): Bind for 0.0.0.0:80 failed: port
is already allocated.

C:\Users>docker container run alphino echo "Hello world"
Unable to find image 'alphino:latest' locally
docker: Error response from daemon: pull access denied for alphino, repository does not exist or may require 'docker login': denied: requested access to the resource is denied.
See 'docker run --help'.

C:\Users>
```

```

C:\Users\parva\Downloads\plasmadonor>plasmadonor\plasmadonor\static>docker run -p 5000:5000 plasmadonor
Unable to find image 'plasmadonor:latest' locally
docker: Error response from daemon: pull access denied for plasmadonor, repository does not exist or may require 'docker login': denied: requested access to the resource is denied.
See 'docker run --help'.

C:\Users\parva\Downloads\plasmadonor\plasmadonor\plasmadonor\static>cd..

C:\Users\parva\Downloads\plasmadonor\plasmadonor>docker run -p 5000:5000 plasmadonor
Unable to find image 'plasmadonor:latest' locally
docker: Error response from daemon: pull access denied for plasmadonor, repository does not exist or may require 'docker login': denied: requested access to the resource is denied.
See 'docker run --help'.

C:\Users\parva\Downloads\plasmadonor\plasmadonor\plasmadonor>docker run -p 5000:5000
'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\parva\Downloads\plasmadonor\plasmadonor>docker build -t plasmadonor .
[*] Building 382.5s (4/9)
-> [internal] load build definition from Dockerfile
=> transferring dockerfile: 179B
-> [internal] load .dockerignore
=> transferring context: 0B
-> [internal] load metadata for docker.io/library/python:3.10.6
[1/5] FROM docker.io/library/python:3.10.6sha256:745efdfb7e4aac9a8422db8c62d8b35a693e8979a240d29677cb3e6aap1052
-> resolve docker.io/library/python:3.10.6sha256:745efdfb7e4aac9a8422db8c62d8b35a693e8979a240d29677cb3e6aap1052
=> sha256:841ff1fca311ce5d5c092ae92a51b0d00a09d73ac11fca 2.23kB / 2.23kB
=> sha256:3e9d413e5e7ade11f7421376f51b95c7e17b0931f870aa9260968d81f6e4 5.16kB / 5.16kB
=> sha256:69b27697f01242279027b1dca15083365700140999 48.0kB / 48.0kB
=> sha256:745efdfb7e4aac9a8422db8c62d8b35a693e8979a240d29677cb3e6aap1052 2.35kB / 2.35kB
=> sha256:d25a6630881028103f1f606d777b0a5cd3b19126fb0b7c118b0557404bcbf84 8.53kB / 8.53kB
=> sha256:217131e310101116d735081108a33c37958b8289f906d05c92916 / 55.01kB
=> sha256:53ad072f9cd1d8cf8e3b1822b20e758611accbdf60babe70cf1043c08de1901a 54.509B / 54.509B
=> sha256:d6b931175338718374f1701ef593dd2af6613c7908c655b6e2a150e644ba 106.959B / 106.959B
=> sha256:d6b931175338718374f1701ef593dd2af6613c7908c655b6e2a150e644ba 106.959B / 106.959B
=> extracting sha256:1671565c8bffc365c96ac1d3fbc16da73001f108436c187958042d4f99vadu2a 3.6kB / 3.6kB
=> extracting sha256:3e9d413e5e7ade11f7421376f51b95c7e17b0931f870aa9260968d81f6e4 0.4kB / 0.4kB
=> extracting sha256:69b27697f01242279027b1dca15083365700140999 258.25kB / 258.25kB
=> sha256:c71afcc83705adcc4d543c140540d4f82635b0b204f0057ea22c6ac8a1d2b5a5 20.029B / 20.029B
=> extracting sha256:53ad072f9cd1d8cf8e3b1822b20e758611accbdf60babe70cf1043c08de1901a 2.77kB / 2.77kB
=> sha256:86da405c30453304061fcd11b1e1a107204416365f01a3e6d0289431a0d6b 2.34kB / 2.34kB
=> sha256:433ab2fe829d419ddc1c3550993aae08f21601a7c8e31c6d6d04c48fbced3c 3.04kB / 3.04kB
-> [internal] load build context
=> transferring context: 331.95kB

```

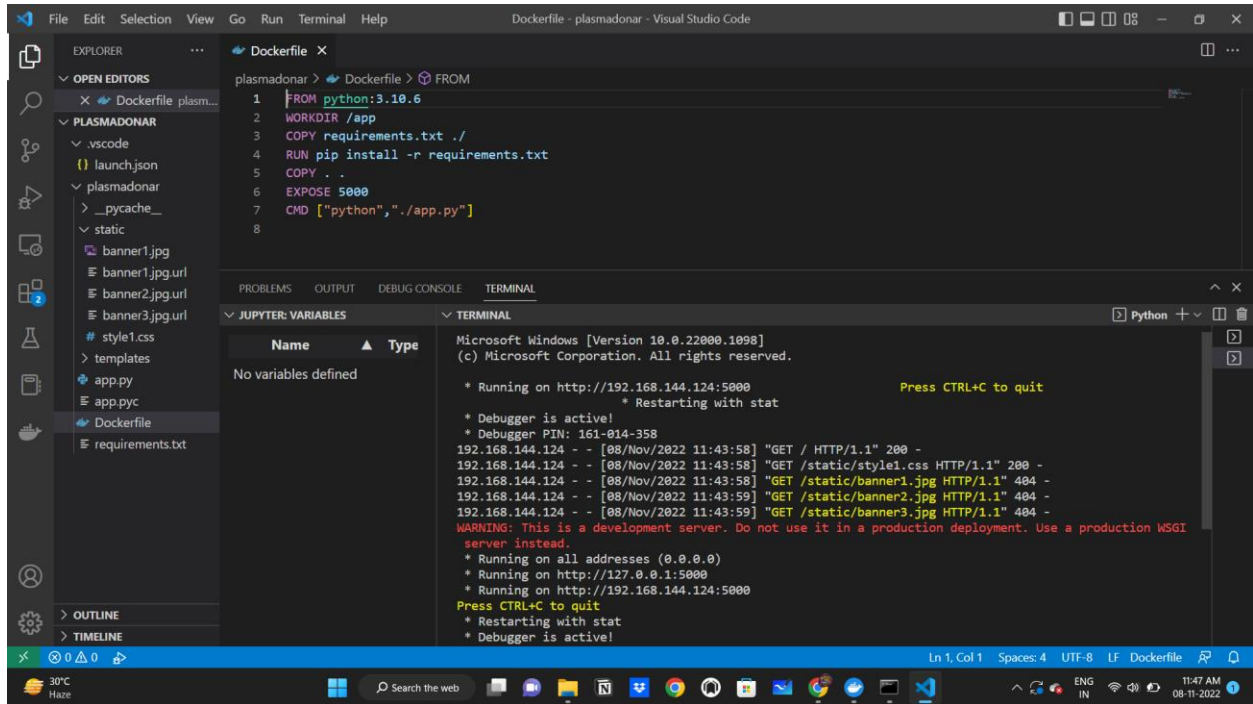
```
Select Command Prompt - docker build -t plasmadonar .
Usage: docker run [OPTIONS] IMAGE [COMMAND] [ARG...]

Run a command in a new container

C:\Users\parva\Downloads\plasmadonar>docker build -t plasmadonar .
[*] Building 573.2s (8/9)

[+] [internal] load build definition from Dockerfile
0.1s
=> Transferring dockerfile: 179B
0.0s
=> Transferring context: 2B
0.1s
[+] [internal] load metadata for docker.io/library/python:3.10.6
0.3s
[1/5] FROM docker.io/library/python:3.10.6@sha256:745ef6fb7e4daac9a8422bd8c6208b35a603e8079a248d7967774b36aa01052
553.7s
=> resolve docker.io/library/python:3.10.6@sha256:745ef6fb7e4daac9a8422bd8c6208b35a603e8079a248d7967774b36aa01052
0.0s
=> sha256:8d1f843ccaa4f8c5cd5df3ce0926e79588308da0870017db9c56d873ac13fca / 2.22kB
0.0s
=> sha256:4e94d1c55e7ade117f41376f971095c7e1700311f4744aa092f080801f6a4 / 5.10kB / 5.10kB
17.3s
=> sha256:1a67723b-082e-402d-b812-792e761da30c4ab5546415000360746 / 10.30kB / 10.30kB
48.3s
=> sha256:745ef6fb7e4daac9a8422bd8c6208b35a603e8079a248d7967774b36aa01052 / 2.25kB
0.0s
=> sha256:425a6630801d83003f760dd777bb5c5d10128f0b671d10570464bcf84 / 8.55kB / 8.55kB
0.0s
=> sha256:1671565ccdd8fc365c36661d3fbc10d73a01f1bda58b6c179588a28f99a4da2e / 55.01kB / 55.01kB
152.0s
=> sha256:53aa07719c4d16f1eb97b1121b30e758c1acde608abef0bf1043c80de1001a / 54.50kB / 54.50kB
247.2s
=> sha256:d08081117931073107f4f93d02f6613c7908c6552eb2ca155d4da1947900 / 136.79kB / 136.79kB
532.2s
=> sha256:4005c36da054267efc30223f6b4d6cf409a6b7f644d28a18dbcb781c24dc / 6.29kB / 6.29kB
204.7s
=> extracting sha256:1671565ccdd8fc365c36661d3fbc10d73a01f1bda58b6c179588a28f99a4da2e
3.6s
=> extracting sha256:3e04d13e50c7ade117f41376f971095c7e1700311f4744aa092f080801f6a4
0.0s
=> extracting sha256:fabc7528c685216129e8e5707362a7702e7b1daa58ab8554aa41508810b570d
0.3s
=> sha256:c71afcc376908c44c51d3c141004d782b1900b30410057ea22fda6a1620345 / 20.02kB
301.4s
=> extracting sha256:53aa07719c4d16f1eb97b1121b30e758c1acde608abef0bf1043c80de1001a
2.7s
=> sha256:86da1083c70455308c65fcd1212baee1c07040f6305f9c350da205413040b / 234B / 234B
250.2s
=> sha256:433ab2fe0291d19ddc13559091aae87f1601a7c85a11c6d6b04c48f6ed3c / 3.04kB / 3.04kB
275.0s
=> extracting sha256:000091317513b7183741701ef938d2afaa613c7900c0553be8e2a15de648a
12.9s
=> extracting sha256:0009256ed5d97e7c3d2256ebdc6ff493aeb7dd4d28a18dbcb781c24dc
0.0s
=> extracting sha256:c71afcc376908c44c51d3c141004d782b1900b30410057ea22fda6a1620345 / 20.02kB
301.4s
=> extracting sha256:86da1083c70455308c65fcd1212baee1c07040f6305f9c350da205413040b
0.0s
=> extracting sha256:433ab2fe0291d19ddc13559091aae87f1601a7c85a11c6d6b04c48f6ed3c
0.0s
[+] [internal] load build context
0.1s
=> transferring context: 331.05kB
0.1s
[2/5] WORKDIR /app
0.3s
[3/5] COPY requirements.txt ./
0.1s
[4/5] RUN pip install --requirements.txt
12.3s
[5/5] COPY . .
0.2s
```

Docker File :



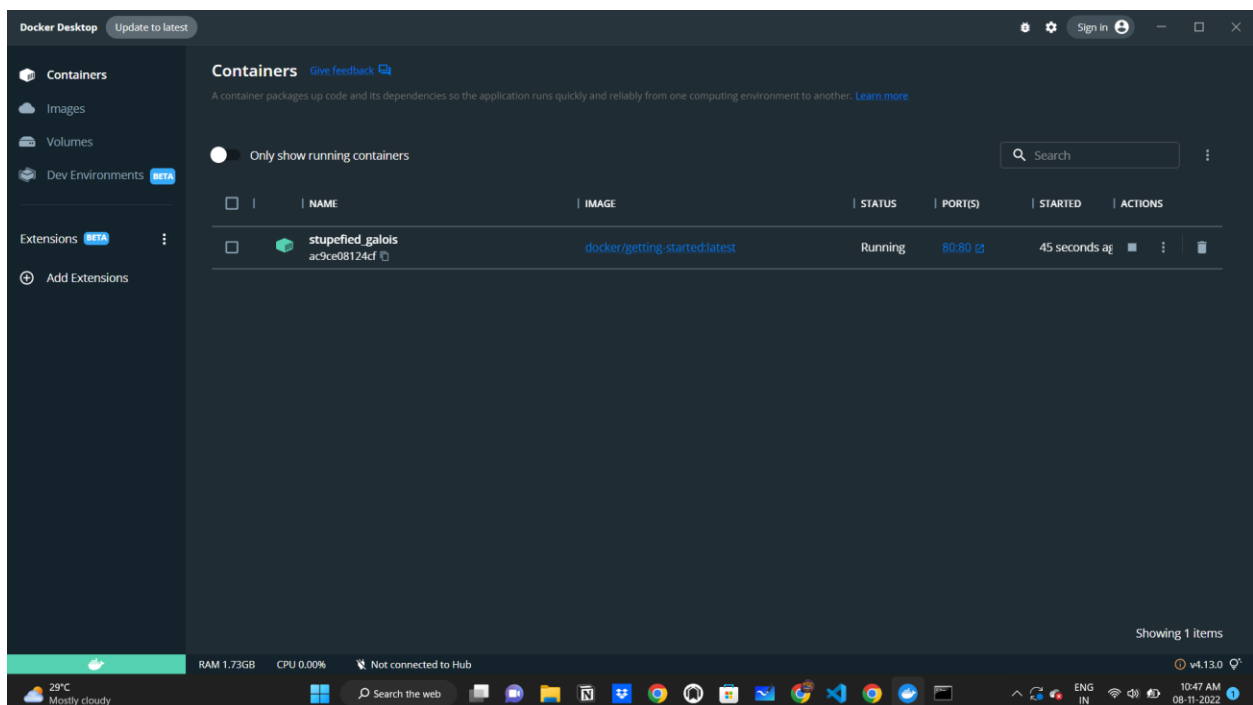
The screenshot shows the Visual Studio Code interface with a Dockerfile open in the editor. The Dockerfile contains the following instructions:

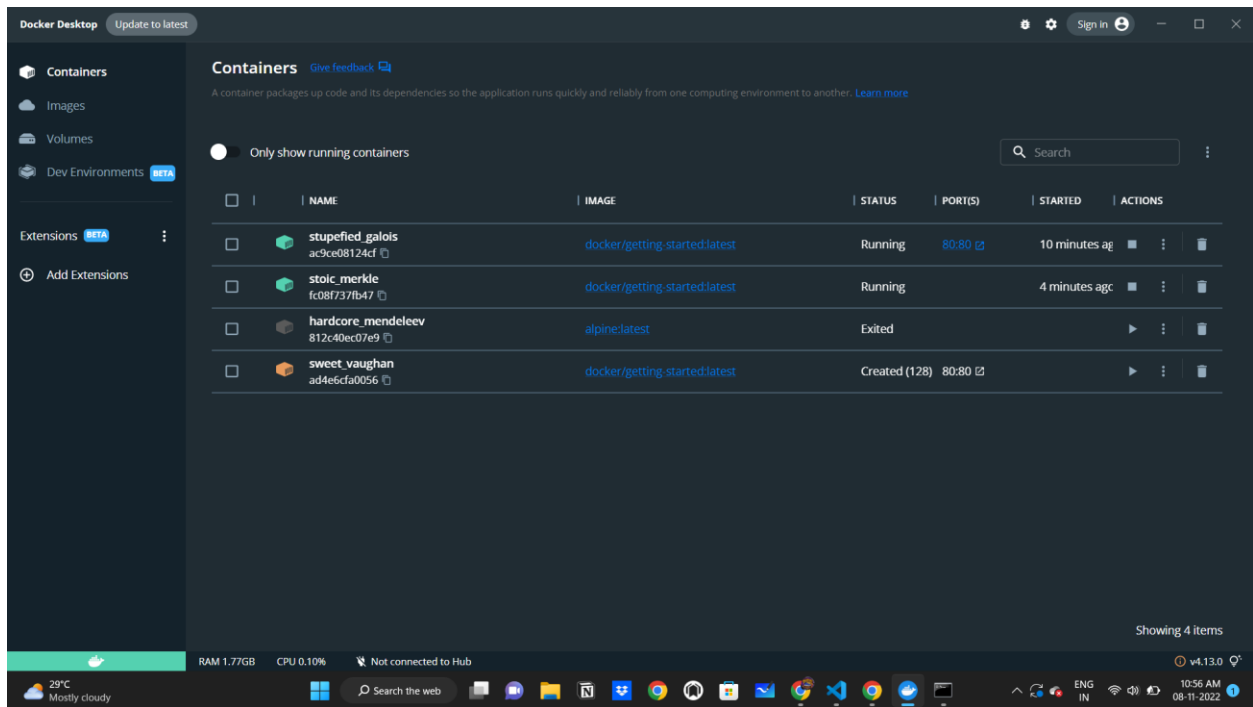
```
1 FROM python:3.10.6
2 WORKDIR /app
3 COPY requirements.txt ./
4 RUN pip install -r requirements.txt
5 COPY . .
6 EXPOSE 5000
7 CMD ["python", "./app.py"]
8
```

The terminal window shows the output of the Docker build and run process. It indicates that the container is running on http://192.168.144.124:5000 and displays the following logs:

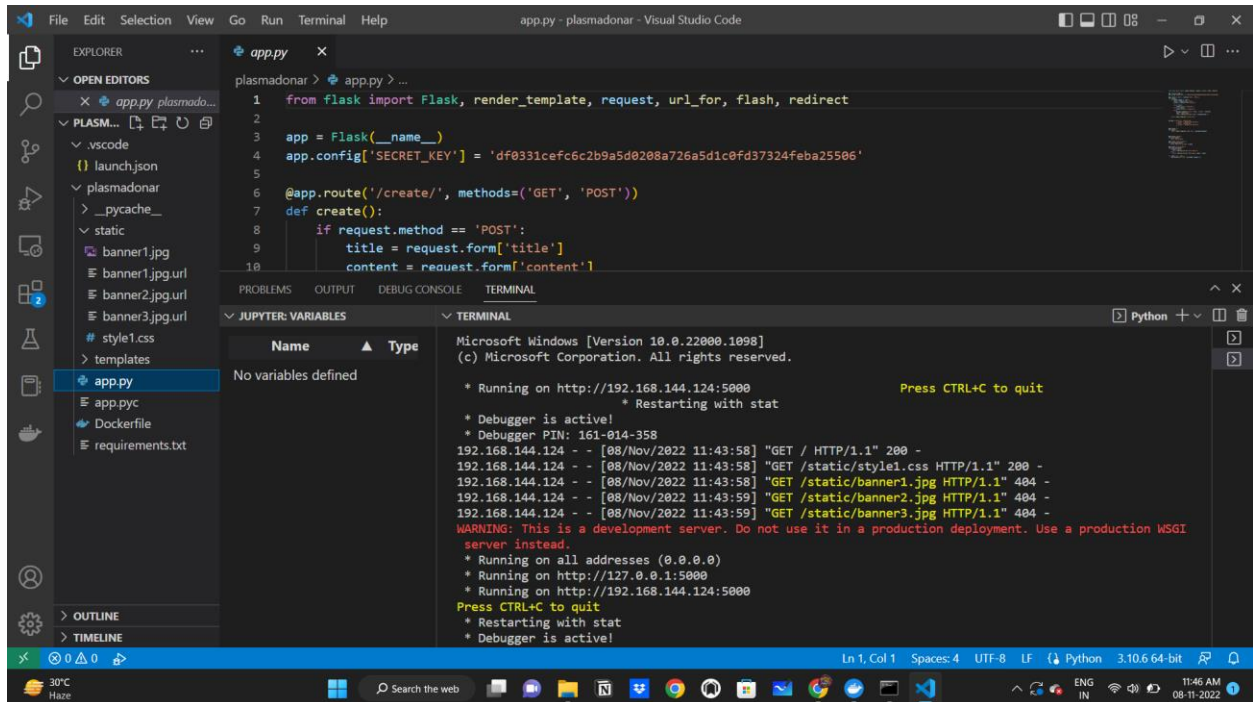
```
* Running on http://192.168.144.124:5000
* Debugger is active!
* Restarting with stat
192.168.144.124 - - [08/Nov/2022 11:43:58] "GET / HTTP/1.1" 200 -
192.168.144.124 - - [08/Nov/2022 11:43:58] "GET /static/style1.css HTTP/1.1" 200 -
192.168.144.124 - - [08/Nov/2022 11:43:58] "GET /static/banner1.jpg HTTP/1.1" 404 -
192.168.144.124 - - [08/Nov/2022 11:43:58] "GET /static/banner2.jpg HTTP/1.1" 404 -
192.168.144.124 - - [08/Nov/2022 11:43:58] "GET /static/banner3.jpg HTTP/1.1" 404 -
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI
server instead
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://192.168.144.124:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
```

IBM Container :





Hello world program :



Home Page :

