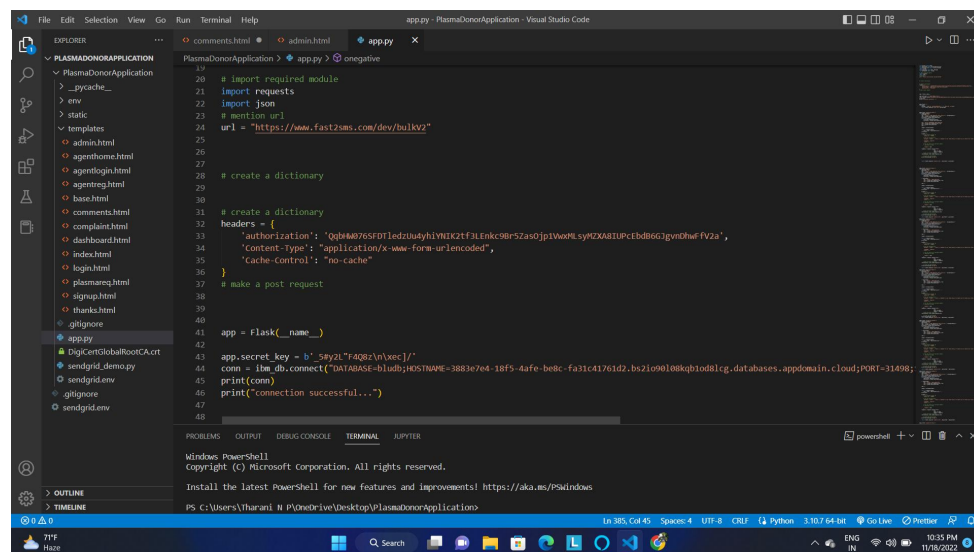


# Containerize Your Flask Application

Student Name	Praveen Kumar S
Student Roll Number	610519104079

## Question:

### 1. Add docker file in flask app

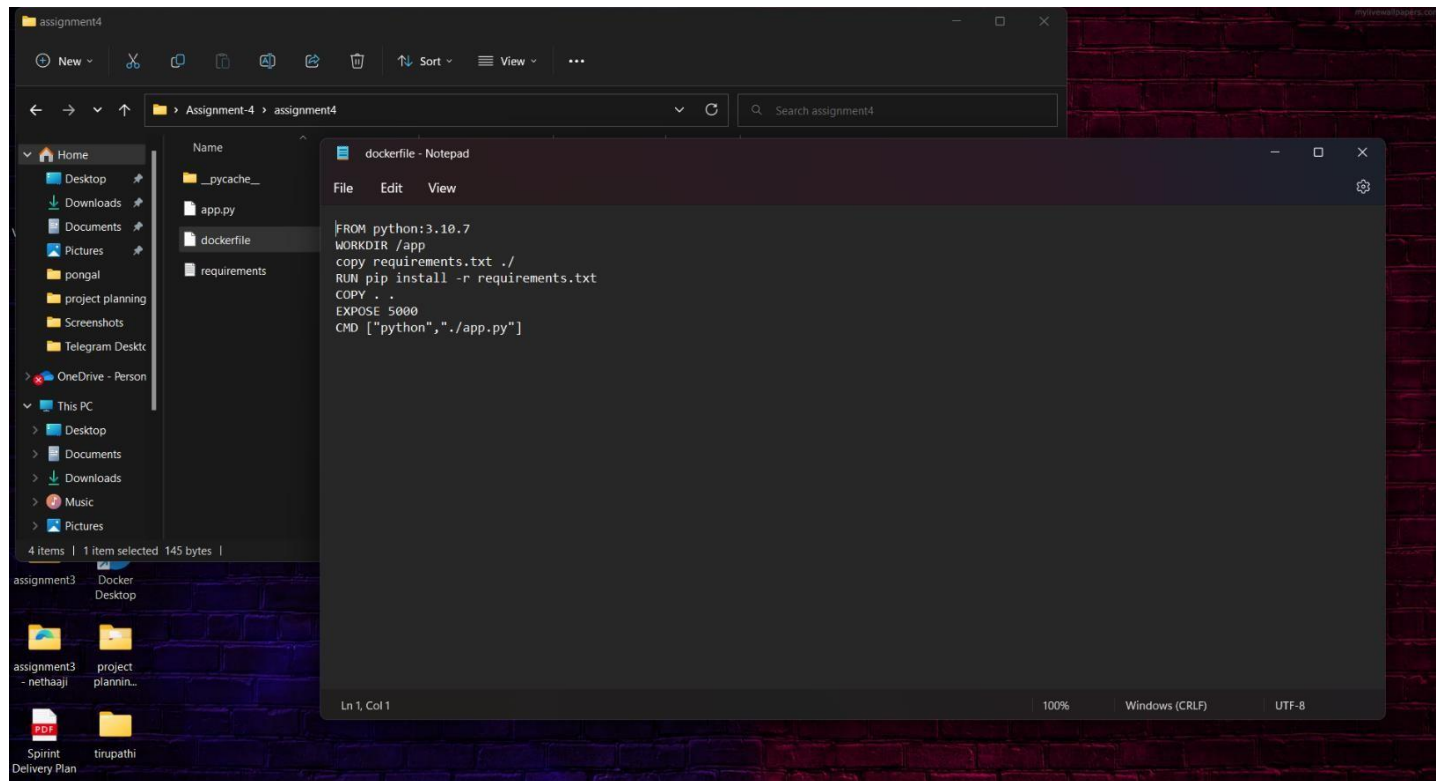


The screenshot shows the Visual Studio Code interface with a project named 'PlasmaDonorApplication'. The Explorer sidebar on the left shows a file tree with various HTML templates and Python files. The main editor window displays a Python file named 'app.py' with the following code:

```
19 # import required module
20 import requests
21 import json
22 # mention url
23 url = "https://www.fast2sum.com/dev/bulky2"
24
25 # create a dictionary
26
27 # create a dictionary
28 headers = {
29     'authorization': 'Qgb4M076SF0rledz0u4yhiVNIK2tF3LEncc9Br5ZasoJp1vwoM5syR2XAB1UPcBd86G1pvdHwF VV2a',
30     'Content-Type': 'application/x-www-form-urlencoded',
31     'Cache-Control': 'no-cache'
32 }
33 # make a post request
34
35 app = Flask(__name__)
36
37 app.secret_key = b'Sey2L7f4Q8eInVscJ/'
38 conn = ibm_db.connect("DA7ABASE-blusdb;HOSTNAME=3883e764-18f5-4afe-bede-fa31c41761d2.b21090108qb10d81cg.databases.appdomain.cloud;PORT=31498;")
39 print(conn)
40 print("connection successful...")
41
42
```

The terminal window at the bottom shows the Windows PowerShell prompt with the command 'PS C:\Users\tharani.N\P\OneDrive\Desktop\PlasmaDonorApplication>'.

## 2. Docker File.



## 3. Build docker image using docker build command.

```
Command Prompt
C:\Users\RISHI\Desktop\Assignment-4\assignment4>docker build -t docker_with_flask_inventory .
[+] Building 115.0s (10/10) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 32B
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/python:3.10.7
=> [1/5] FROM docker.io/library/python:3.10.7@sha256:53e577204d362233ee92aeb5119449271f5eb24f99c61464efe9167ddbc8640f
=> => resolve docker.io/library/python:3.10.7@sha256:53e577204d362233ee92aeb5119449271f5eb24f99c61464efe9167ddbc8640f
=> => sha256:d9d6e6c37c6e95c0bae489270ff19b0195e22b07cae5b410a8884a13b107 8.33kB / 8.33kB
=> => sha256:f686d8928ed378229f246b94b584cca239fb906efc57achdf93d0bd298d5dd6 55.05MB / 55.05MB
=> => sha256:47db815c6a4547dc224b7522193cb1851cf529d2cbdf26f85ab9bbf97099b98 5.16MB / 5.16MB
=> => sha256:53e577204d362233ee92aeb5119449271f5eb24f99c61464efe9167ddbc8640f 2.36kB / 2.36kB
=> => sha256:d480cf985966814b180d9c8b522917b13a1ce49fc190f2a047220307fedea5c 2.22kB / 2.22kB
=> => sha256:bfa48494000001a037b72870d2a6a2536f9da8bc5d1ceddd72d79f4a51fe7a0be 10.88MB / 10.88MB
=> => sha256:a572f7a256d36a93ab0777049771b120c5d7dce75ea2a2d3d9444793b26b2ef1 54.58MB / 54.58MB
=> => sha256:8f7d052589528f8db73153451e112bbd8e1854549bd1e0e6f4ac0b4a2ee98172 196.85MB / 196.85MB
=> => sha256:7110f04115ae2d7232ed9e59b97db7cf7337c91f95edc25428baa3e522064187 6.29MB / 6.29MB
=> => sha256:c4b413c6a4894499e2ee7df958d411ccfcc899dcccffe61de71cbb433e4e76143 20.05MB / 20.05MB
=> => sha256:22311b72a3cb993d70dc2f9440feb89ca571ae931b080d975c81f4270ca909b8 231B / 231B
=> => sha256:8dc8fe38b6fa1182030e006f135471f3726cef064f6fc90edbdff0f450efad79 3.04MB / 3.04MB
=> => extracting sha256:f606d8928ed378229f246b94b504cca239fb906efc57acbd9340bd298d5dd6
=> => extracting sha256:47db815c6a4547dc224b7522193cb1851cf529d2cbdf26f85ab9bbf97099b98
=> => extracting sha256:bfa48494000001a037b72870d2a6a2536f9da8bc5d1ceddd72d79f4a51fe7a0be
=> => extracting sha256:a572f7a256d36a93ab0777949771b120c5d7dce75ea2a2d3d9444793b26b2ef1
=> => extracting sha256:8f7d052589528f8db73153451e112bbd8e1854549bd1e0e6f4ac0b4a2ee98172
=> => extracting sha256:7110f04115ae2d7232ed9e59b97db7cf7337c91f95edc25428baa3e522064187
=> => extracting sha256:c4b413c6a4894499e2ee7df958d411ccfcc899dcccffe61de71cbb433e4e76143
=> => extracting sha256:22311b72a3cb993d70dc2f9440feb89ca571ae931b080d975c81f4270ca909b8
=> => extracting sha256:8dc8fe38b6fa1182030e006f135471f3726cef064f6fc90edbdff0f450efad79
=> [internal] load build context
=> => transferring context: 190B
=> [2/5] WORKDIR /app
=> [3/5] COPY requirements.txt ./
=> [4/5] RUN pip install -r requirements.txt
=> [5/5] COPY . .
=> exporting to image
=> => exporting layers
=> => writing image sha256:5d844c12d2eff9ba170fc15034f0c9c00efbed528ea42cb6a351f21377d4c6bd
=> => naming to docker.io/library/docker_with_flask_inventory
0.0s
0.0s
0.0s
0.0s
1.2s
109.0s
0.0s
0.0s
73.0s
3.8s
0.0s
0.0s
0.0s
10.3s
34.3s
100.3s
41.3s
59.3s
60.6s
64.3s
2.3s
0.3s
0.4s
3.0s
6.8s
0.4s
0.8s
0.0s
0.2s
0.0s
0.0s
0.3s
0.0s
4.1s
0.0s
0.7s
0.1s
0.0s
0.0s

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

C:\Users\RISHI\Desktop\Assignment-4\assignment4>kubect1 get nodes
NAME STATUS ROLES AGE VERSION
docker-desktop Ready control-plane 6m19s v1.25.2

C:\Users\RISHI\Desktop\Assignment-4\assignment4>
```

## Running in docker desktop

The screenshot shows the Visual Studio Code interface with a file explorer on the left containing 'app.py', 'dockerfile', and 'requirements.txt'. The main editor displays the content of 'app.py', which is a Flask application. The terminal at the bottom shows the command to run the application and its output.

```
app.py
1 from flask import Flask
2 from markupsafe import escape
3
4 app = Flask(__name__)
5
6 @app.route("/")
7 def index():
8     return "<p> Inventory Management System <br> Hello Guys <p>"
9
10 @app.route("/<name>")
11 def hello(name):
12     return f"hello,{escape(name)}!"
13
14 if __name__ == "__main__":
15     app.run(host="0.0.0.0",port=5000)
```

```
PS C:\Users\RISHI\Desktop\Assignment-4\assignment4> & C:/Users/RISHI/AppData/Local/Programs/Python/Python310/python.exe c:/Users/RISHI/Desktop/Assignment-4/assignment4/app.py
* Serving Flask app 'app'
* Debug mode: off
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://192.168.101.5:5000
Press CTRL+C to quit
```

## 4. Run and test your container.

The screenshot shows the Docker Desktop interface. The left sidebar contains navigation options: Containers, Images, Volumes, and Dev Environments (marked as BETA). The main area is titled 'Images on disk' and shows a list of Docker images. The status bar at the bottom indicates RAM usage (3.53GB), CPU usage (3.60%), and connection to Docker Hub.

Image Name	Tag	Digest	Updated	Size
hubproxy.docker.internal:500...	kubernetes-v1.25.2-cni-v...	09d7e1dbc2c4	about 2 months ago	363.32 MB
jp.icr.io/do-fs/docker_with_flas...	latest	5d844c12d2ef	about 14 hours ago	932.79 MB
jp.icr.io/do-fs/docker_with_flas...	latest	5d844c12d2ef	about 14 hours ago	932.79 MB
k8s.gcr.io/coredns	v1.9.3	5185b96f0bec	6 months ago	48.8 MB
k8s.gcr.io/etcd	3.5.4-0	a8a176a5d5d6	5 months ago	299.52 MB
k8s.gcr.io/kube-apiserver	v1.25.2	97801f839490	about 2 months ago	127.73 MB
k8s.gcr.io/kube-controller-ma...	v1.25.2	dbfceb93c69b	about 2 months ago	117.1 MB
k8s.gcr.io/kube-proxy	v1.25.2	1c7d8c51823b	about 2 months ago	61.69 MB
k8s.gcr.io/kube-scheduler	v1.25.2	ca0ea1ee3cfd	about 2 months ago	50.58 MB
k8s.gcr.io/pause	3.8	4873874c08ef	5 months ago	711.18 KB
kubernetesui/dashboard	v2.6.1	783e2b6d87ed	3 months ago	245.72 MB
kubernetesui/metrics-scraper	v1.0.8	115053965e86	5 months ago	43.82 MB
rishiragulr/flask-docker	latest	5d844c12d2ef	about 14 hours ago	932.79 MB