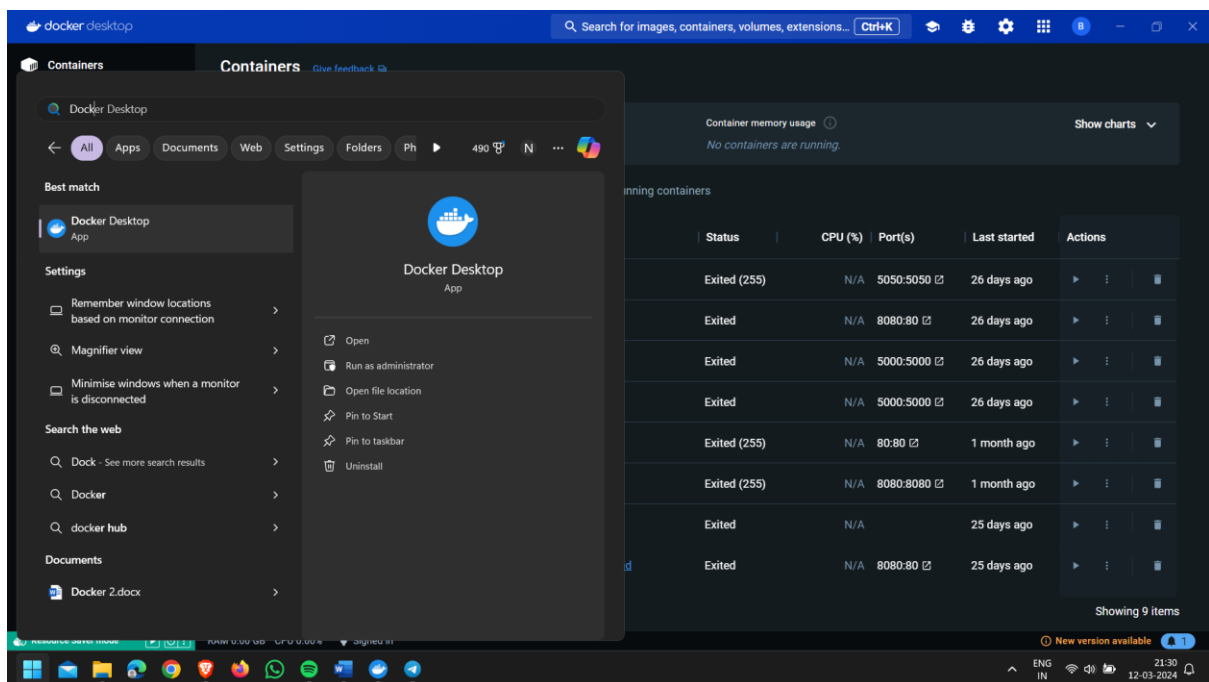


2100032449

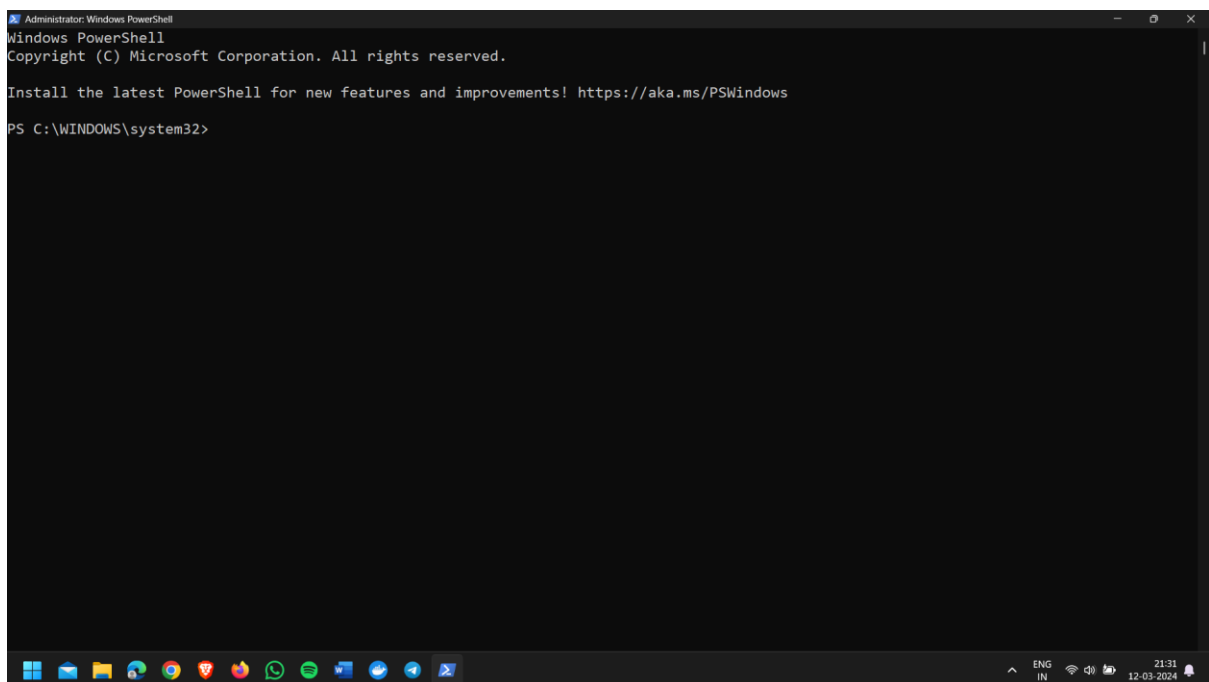
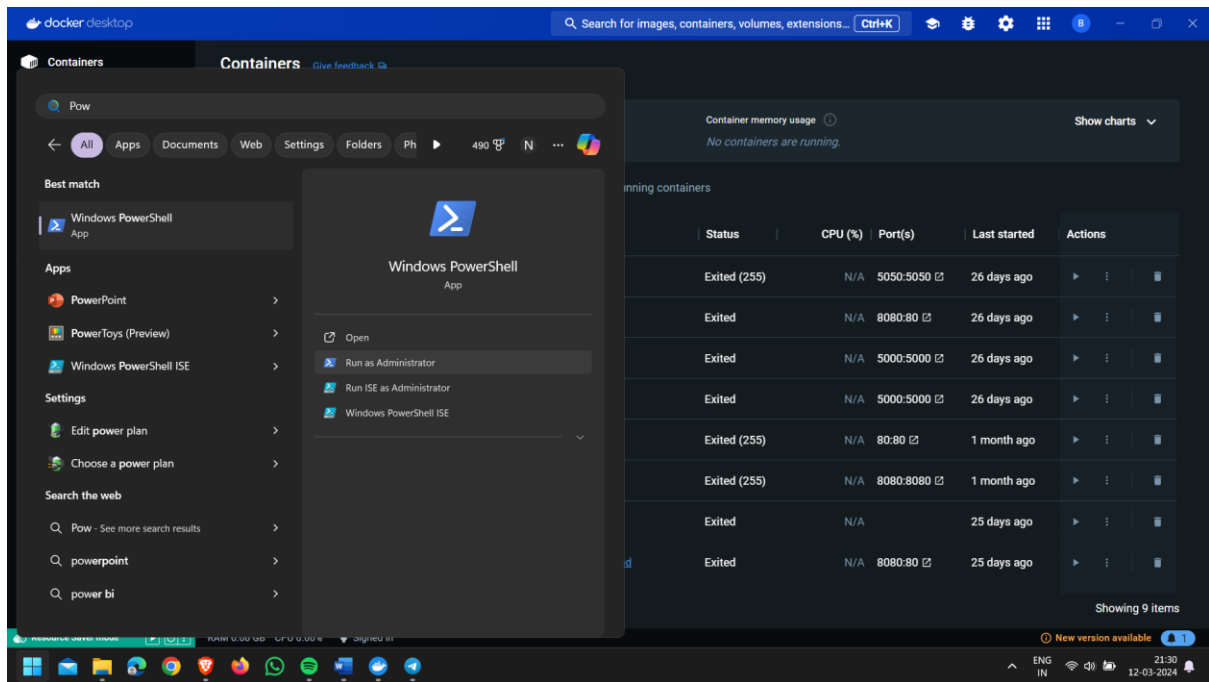
N. Bhanu Satya Venkatachalam

4. Explore Docker Hub for images that will run a website and get them into your development environment and practice

1. Open Docker Desktop



2. Now Open Powershell in Administrator Mode



3. Now to check info use following command

docker info - Displays system wide information about Docker.

4. **docker ps**: Lists the runnin' Docker containers.

5. To pull an Image we can use following Command

docker pull <image_name>

for example for tomcat image

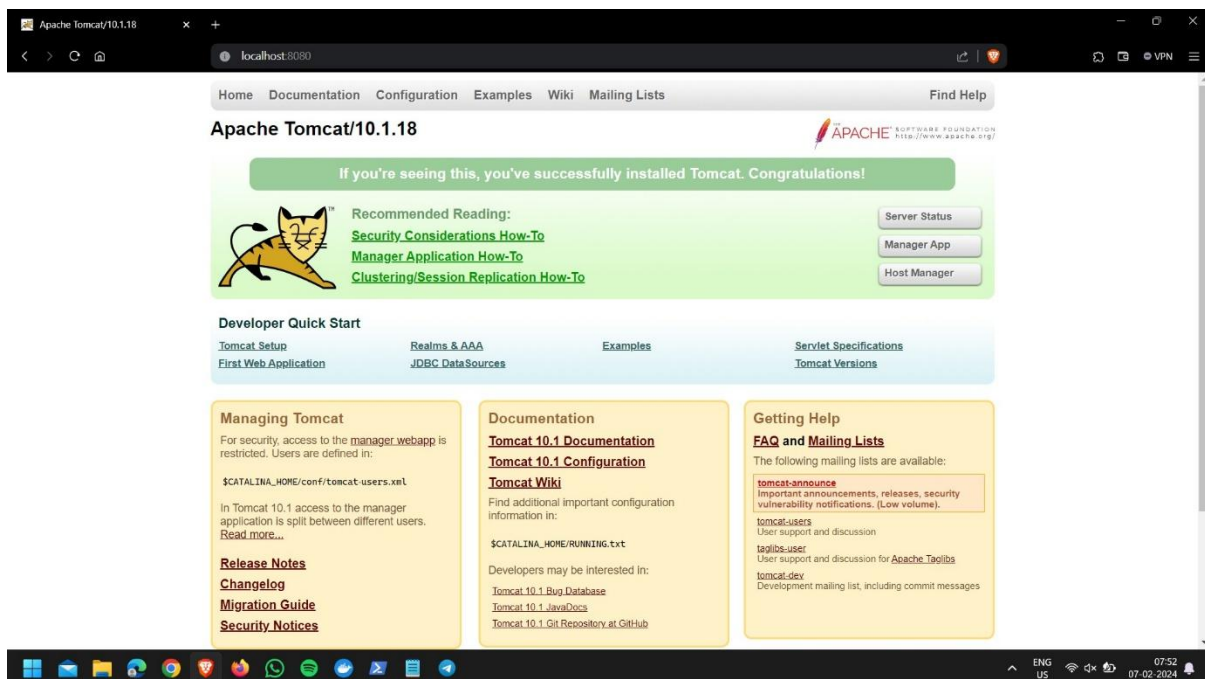
6. To run a pulled image

Use command **docker run -d -p 8080:8080 --name <image_name>**

```
root@5c1e769cbf9d:/usr/local/tomcat
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\WINDOWS\system32> docker run -d -p 8080:8080 --name my-tomcat-container tomcat:latest
Unable to find image 'tomcat:latest' locally
latest: Pulling from library/tomcat
31bd5f451a84: Pull complete
26611c45681a: Pull complete
88a51578b7e9: Pull complete
da92af0c0eb31: Pull complete
9868ee624b7: Pull complete
8c196050ae39: Pull complete
a2c1e994a1a0: Pull complete
3738cbd2ee8d: Pull complete
Digest: sha256:5e95c2e2b75387025bfc0da1e421079ac2b84378efeb70d0b2d54e4fae5832b4
Status: Downloaded newer image for tomcat:latest
5c1e769cbf9d8eae8f735ba2de79cd3b9fd57b493d1726b4f9b34d9dd909238
PS C:\WINDOWS\system32> docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS                               NAMES
5c1e769cbf9d   tomcat:latest   "catalina.sh run"       18 seconds ago Up 7 seconds   0.0.0.0:8080->8080/tcp             my-tomcat-container
PS C:\WINDOWS\system32> docker exec -it 5c1e769cbf9d /bin/bash
root@5c1e769cbf9d:/usr/local/tomcat# cp -r /usr/local/tomcat/webapps.dist/* /usr/local/tomcat/webapps/
root@5c1e769cbf9d:/usr/local/tomcat#
```



7. To build a container using existing image use following command

docker build t <image_name> . : Builds a Docker image from a Dockerfile in the current directory.

8. To create a nginx container

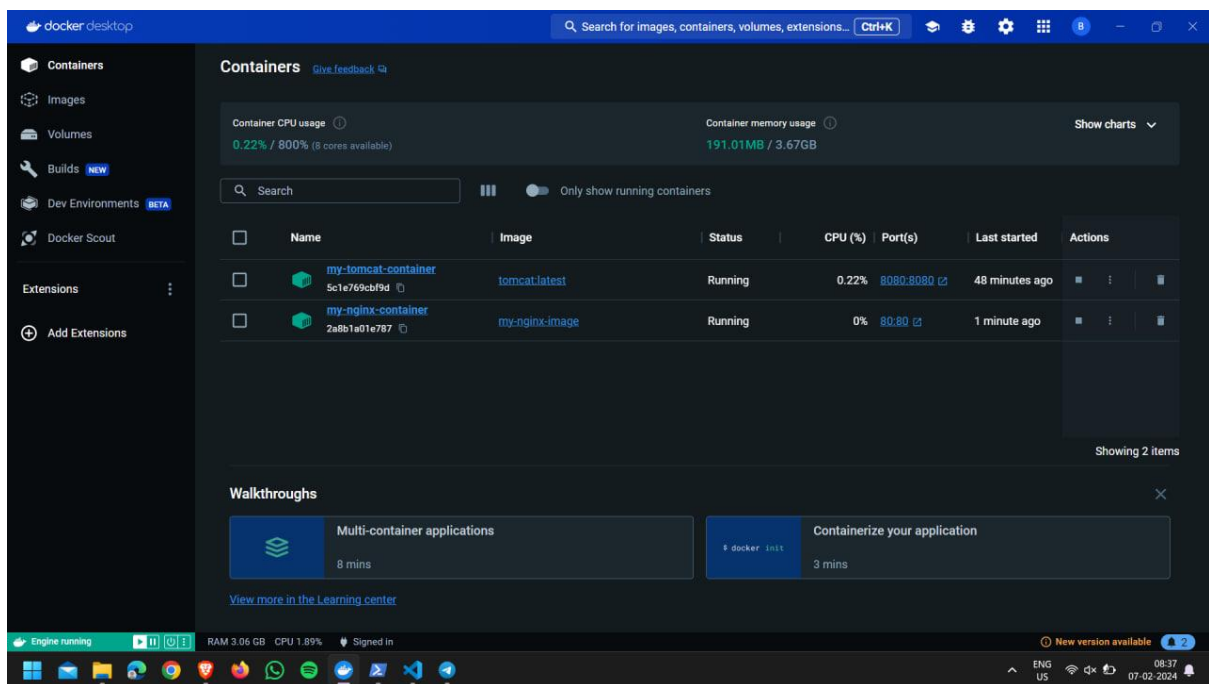
```
Administrator: Windows PowerShell

Mode                LastWriteTime         Length Name
-----
d-----         07-02-2024    08:32            0 Dockerfile

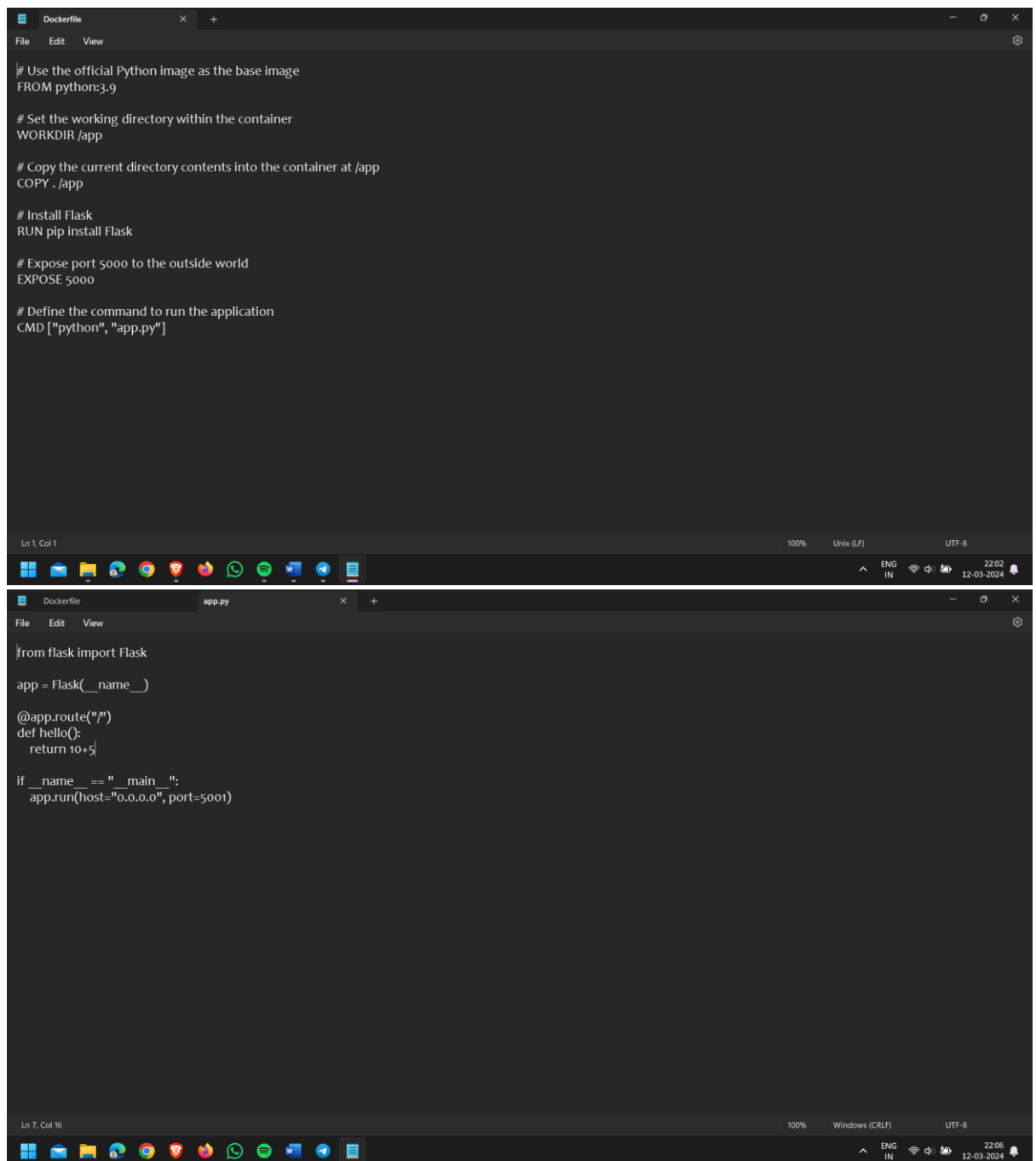
PS C:\Dockerfiles> cat Dockerfile
PS C:\Dockerfiles> cd Dockerfile
PS C:\Dockerfiles> Get-Content Dockerfile
Get-Content: Find path 'C:\Dockerfiles\Dockerfile' because it does not exist.
PS C:\Dockerfiles> cd Dockerfile
PS C:\Dockerfiles> cat Dockerfile
cat: Dockerfile: ObjectNotFoundException: ObjectNotFoundException: (C:\Dockerfiles\Dockerfile:String) [Set-Location], ItemNotFoundException
+ CategoryInfo          : ObjectNotFoundException: (C:\Dockerfiles\Dockerfile:String) [Set-Location], ItemNotFoundException
+ FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.SetLocationCommand

PS C:\Dockerfiles> docker build -t my-nginx-image .
[+] Building 41.9s (6/6) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile              0.1s
=> [internal] load .dockerignore                                  0.1s
=> [internal] load metadata for docker.io/library/nginx:latest  5.0s
=> [internal] resolve image config for docker.io/library/nginx:latestsha256:84c52df455c467e12ef85cad6a25c0990564f03c4850799b41d673873 36.5s
=> [1/1] FROM docker.io/library/nginx:latestsha256:84c52df455c467e12ef85cad6a25c0990564f03c4850799b41d673873 0.0s
=> sha256:84c52df455c467e12ef85cad6a25c0990564f03c4850799b41d6738738081f 9.54kB / 9.54kB
=> sha256:402f99b0d4759e726c20d426842f2f0881c5d079612d27ec36a360132d4 2.24kB / 2.24kB
=> sha256:c57ee500b0d1345a3ee668479ad110326e2274d9a5a7855909d1a26394463 29.15MB / 29.15MB
=> sha256:f24ae6f5277877f9731f1848b8024e614f4d4e2547bba350fc802c87f86d77 627B / 627B
=> sha256:908016325c0874cc35a3090884715be2a301a4d67a334396473683fb15c580833 41.39MB / 41.39MB
=> sha256:9f3589a5f5b0a6b3d47513b58f8a1d8f235aee6ba583679034cedf94ade74f 955B / 955B
=> sha256:f0b09a74d4c76b0a21c01172f32be31b04c881ccc6d4b991e1d4d409 365B / 365B
=> sha256:398157bc5c51a9e2e4c0552c190bee781b157c9ef4760f76ed3d6d34164fe71 1.21kB / 1.21kB
=> sha256:1ef1ca36ec2a46e2e803471d661f479a5a28666c553c4cccc26f222548b307 1.40kB / 1.40kB
=> extracting sha256:c57ee500b0d1345a3ee668479ad110326e2274d9a5a7855909d1a26394463 4.4s
=> extracting sha256:f0b016325c0874cc35a3090884715be2a301a4d67a334396473683fb15c580833 4.4s
=> extracting sha256:f24ae6f5277877f9731f1848b8024e614f4d4e2547bba350fc802c87f86d77 0.0s
=> extracting sha256:9f3589a5f5b0a6b3d47513b58f8a1d8f235aee6ba583679034cedf94ade74f 0.0s
=> extracting sha256:f0b09a74d4c76b0a21c01172f32be31b04c881ccc6d4b991e1d4d409 0.0s
=> extracting sha256:398157bc5c51a9e2e4c0552c190bee781b157c9ef4760f76ed3d6d34164fe71 0.0s
=> extracting sha256:1ef1ca36ec2a46e2e803471d661f479a5a28666c553c4cccc26f222548b307 0.0s
=> exporting to image                                           0.0s

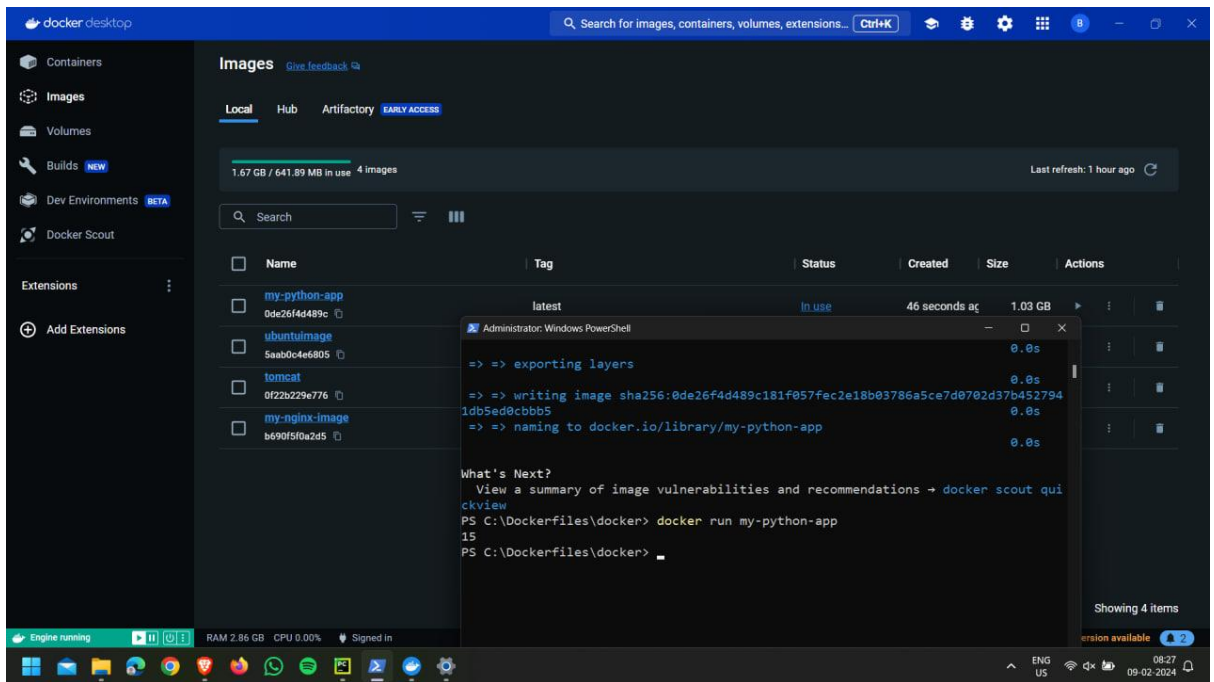
PS C:\Dockerfiles> docker run -d --name my-nginx-container -p 80:80 my-nginx-image
2a8b1a01e787f97a14602977bcd78-0fb919ea67d22769a088c5680c31515e3d
PS C:\Dockerfiles>
```



9. We can also use Dockerfile to automate the process of above steps
10. Example of Dockerfile

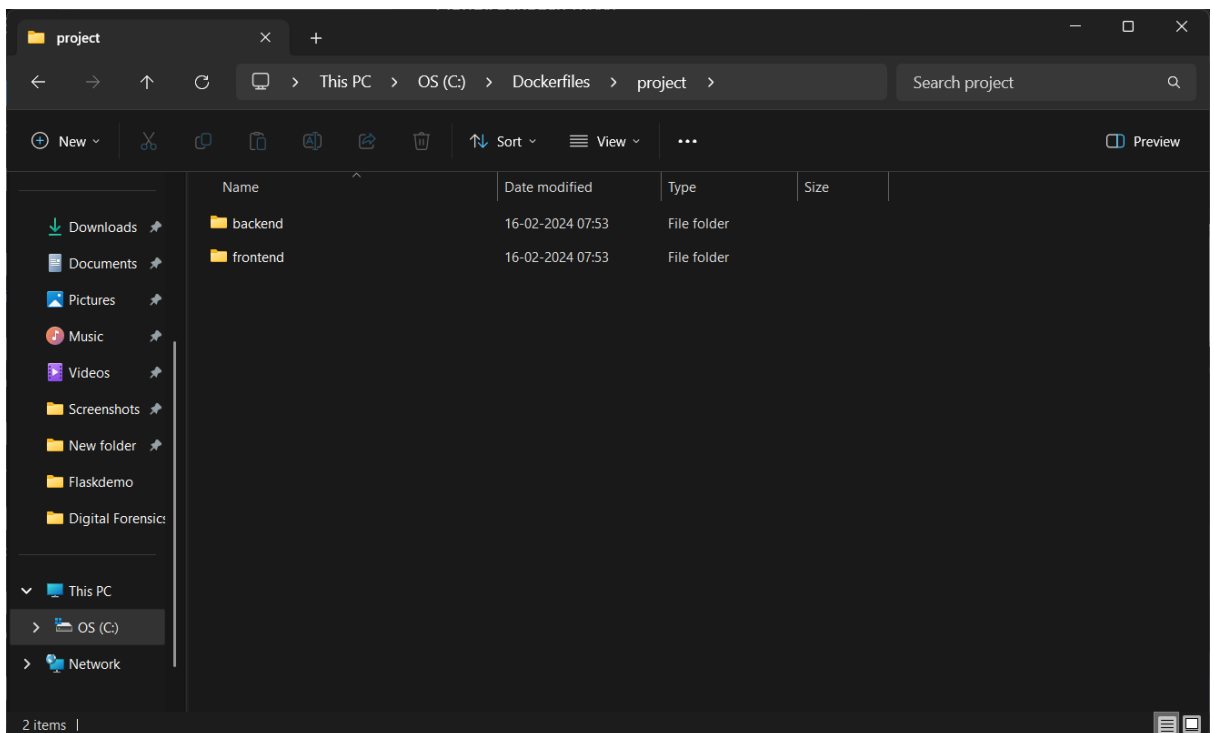


After running the image it runs the app.py file to give output.

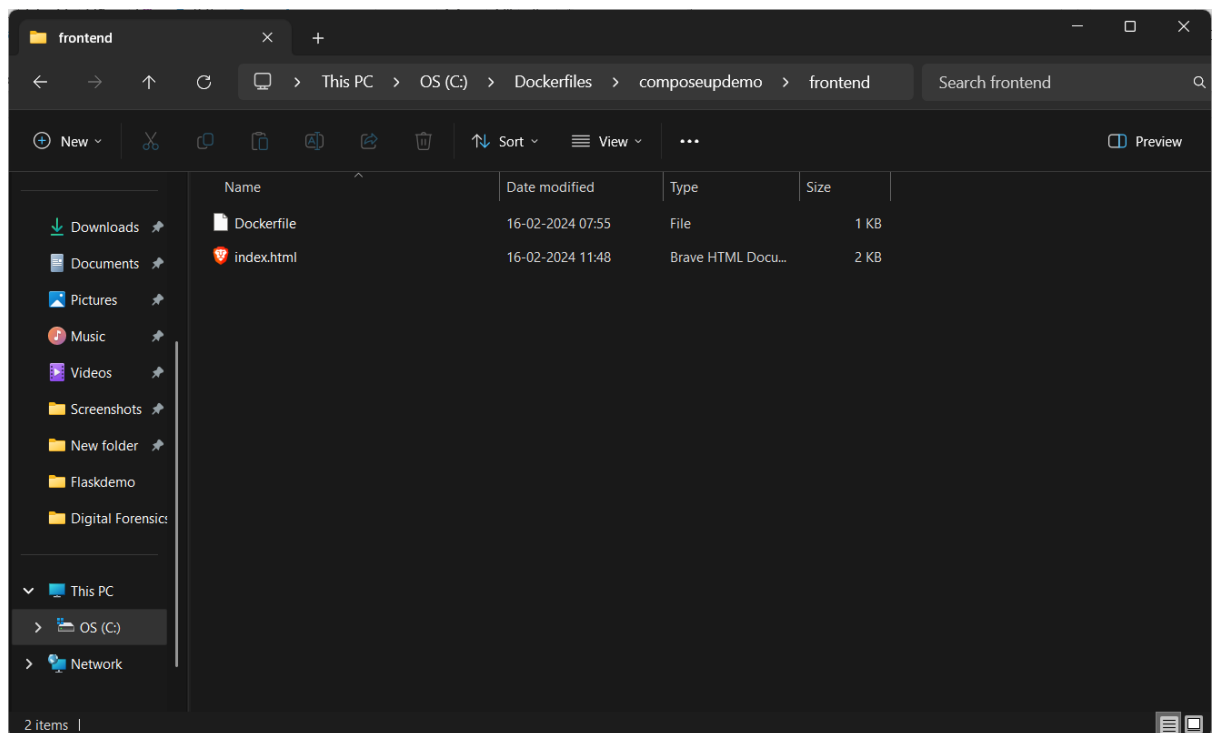


11. To create a development environment we can use **docker compose** to integrate both frontend and backend serves.

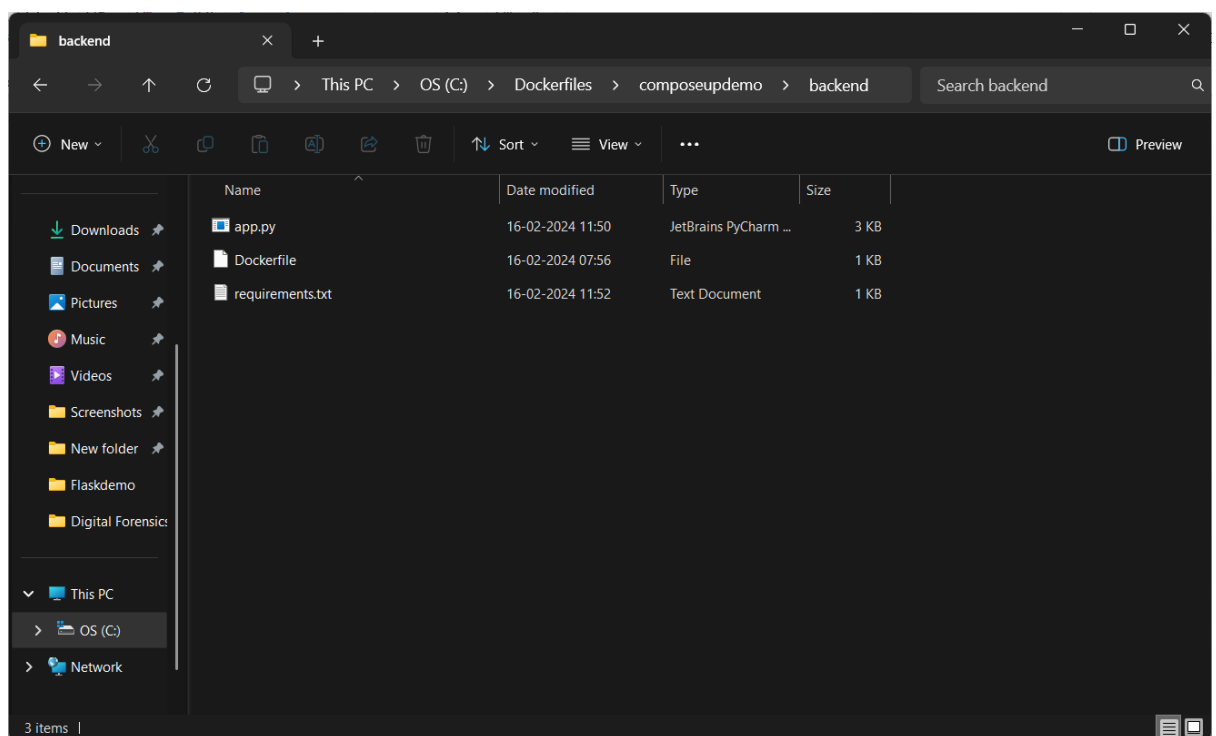
12. For that we need to create individual Dockerfile for both frontend and backend.



Frontend

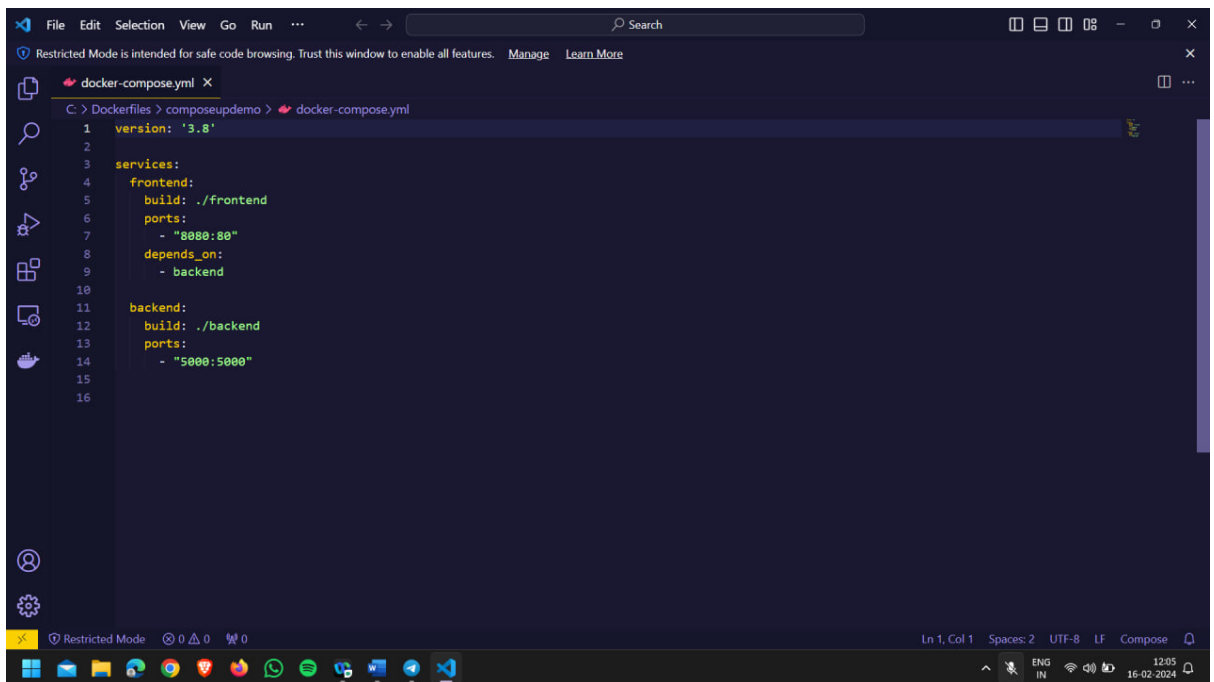
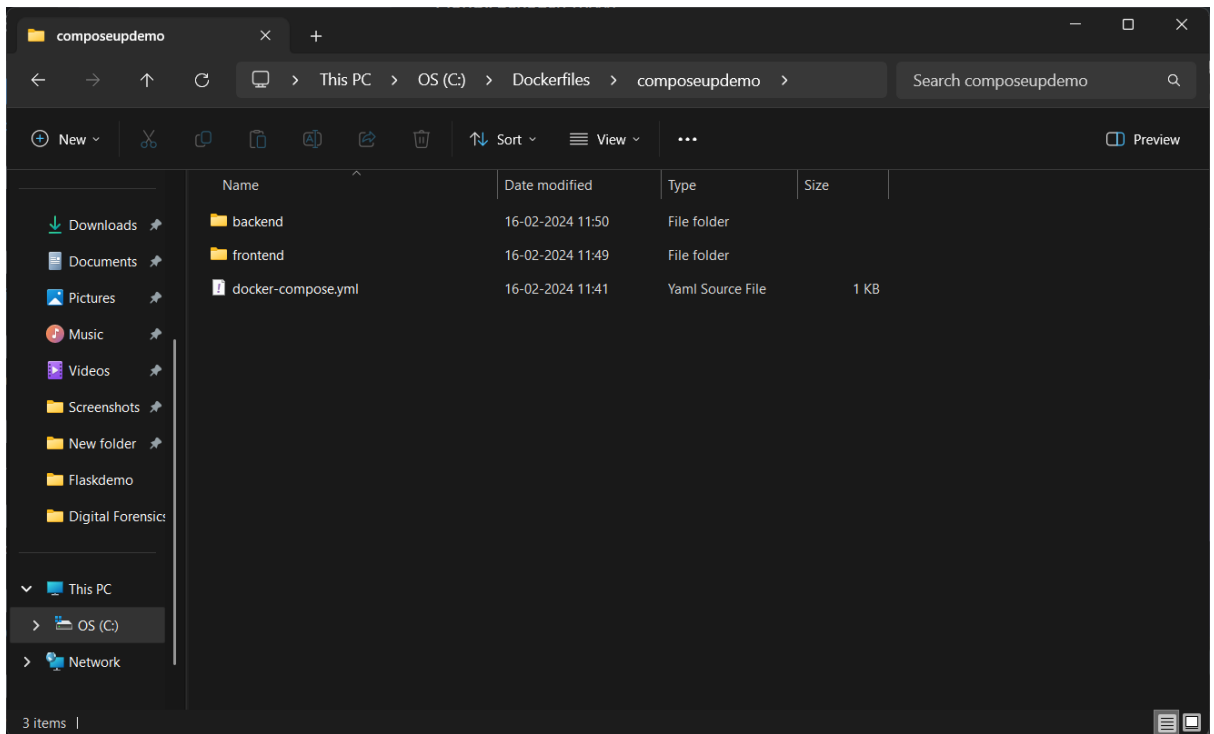


Backend



13. Compose.yml file

Create a compose.yml file.



14. After that use this to build image for both frontend and backend

docker build t <image_name> .

15. Now use this command

docker-compose up

16. After completion the output

This shows the containers running and the websites.

