**Experiment 10 - Docker Image**

| Roll No. | 37 |
| --- | --- |
| Name | Mikil Lalwani |
| Class | D15-B |
| Subject | DevOps Lab |
| LO Mapped | LO1: To understand the fundamentals of DevOps engineering and be fully proficient with DevOps terminologies, concepts, benefits, and deployment options to meet your business requirements.  LO5: To understand the concept of containerization and Analyze the Containerization of OS images and deployment of applications over Docker. |
|

**Aim**:

To learn Dockerfile instructions, and build an image for a sample web application using Dockerfile.

**Introduction**:

What Is a Dockerfile?

So, our first question is simply what is a Dockerfile? When you run the Docker run command and specify WordPress, Docker uses this file to build the image. The Dockerfile is essentially the build instructions to create the image.

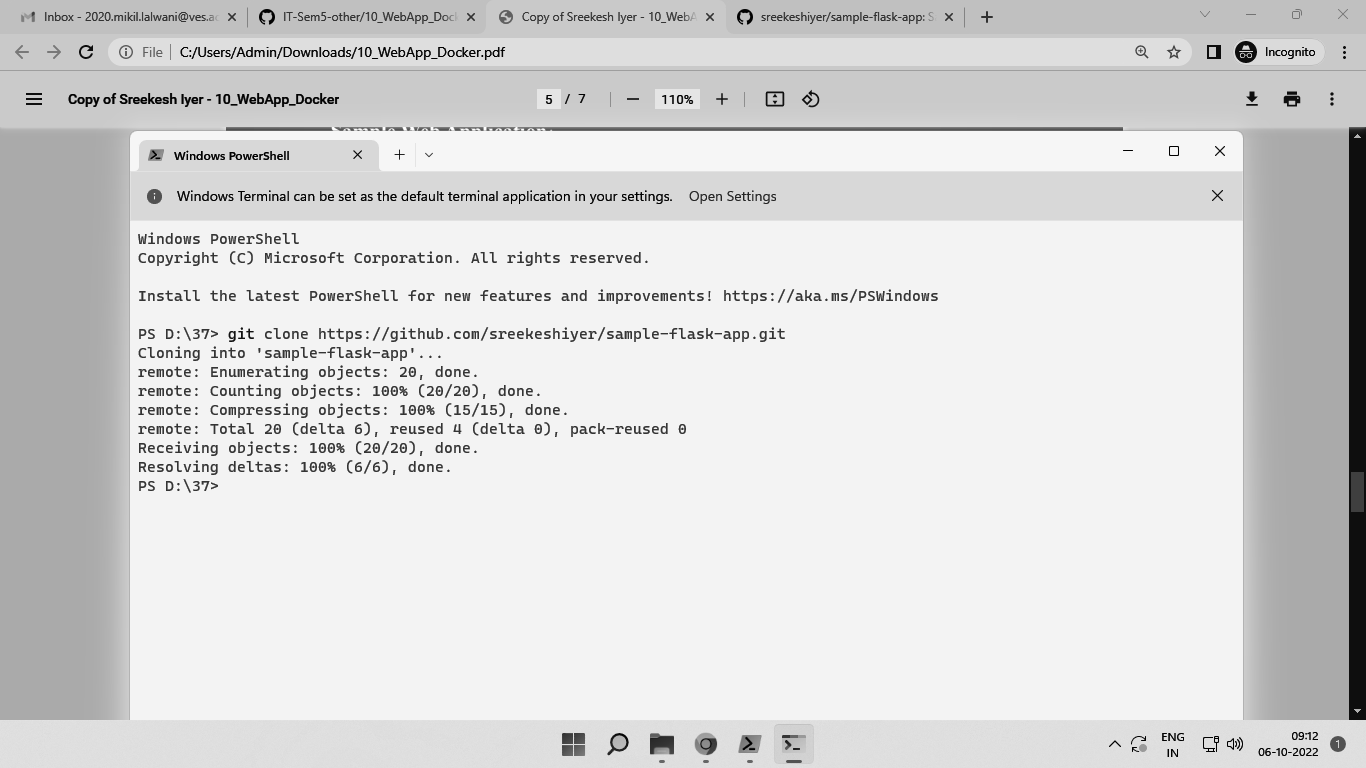
The advantage of a Dockerfile over just storing the binary image (or a snapshot/template in other virtualization systems) is that the automatic builds will ensure you have the latest version available. This is a good thing from a security perspective, as you want to ensure you’re not installing any vulnerable software.

**Building a docker image:**

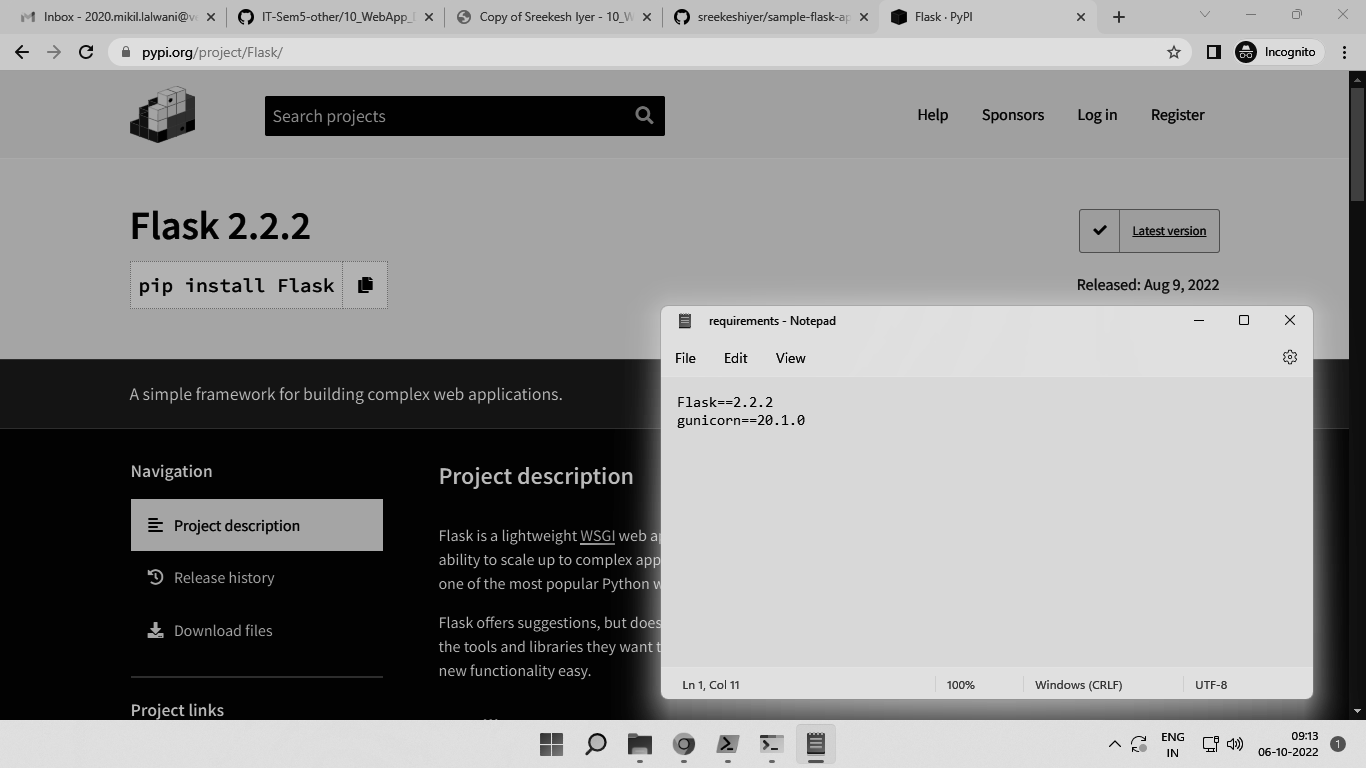
In this experiment, we will create a docker image for a sample flask API which on connection returns a greeting. You can clone this application from this [GitHub repository](https://github.com/sreekeshiyer/sample-flask-app.git).

Steps:

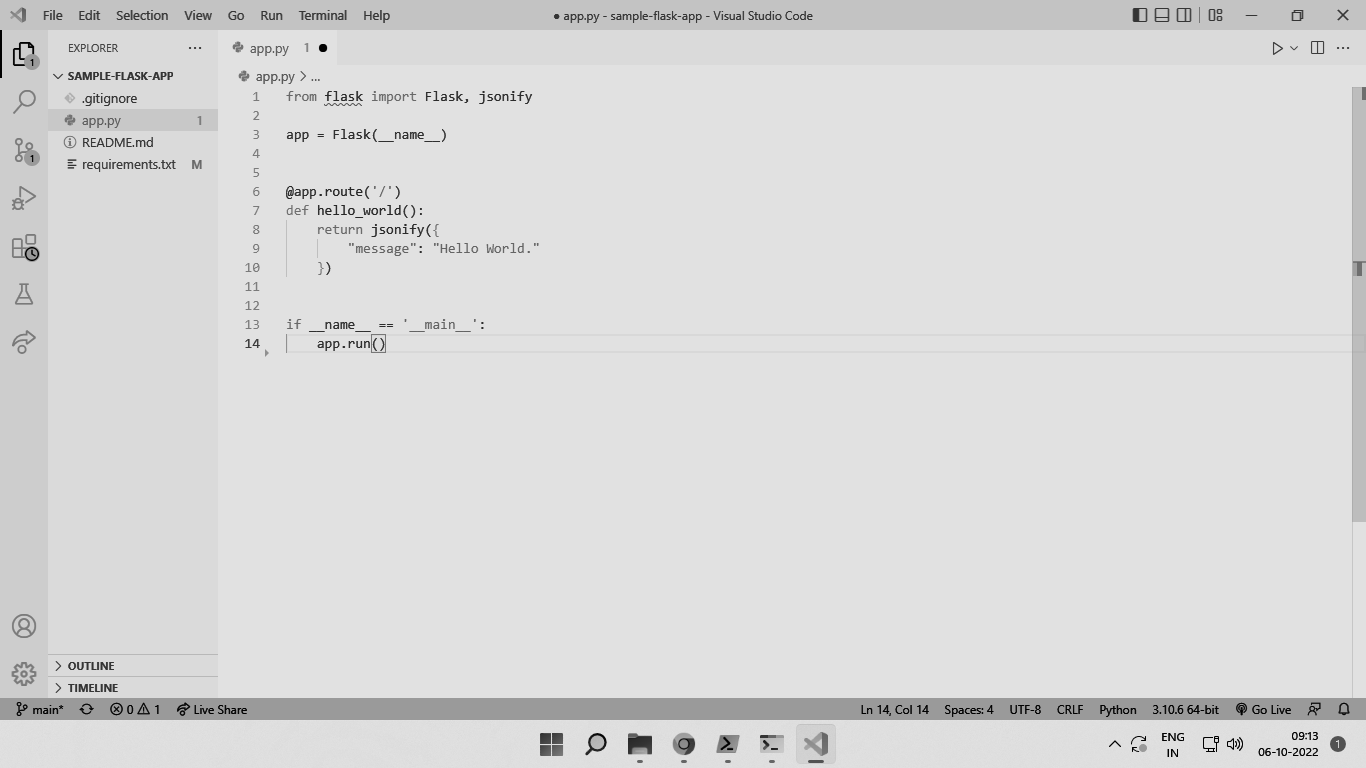
1. Clone the GitHub Repository.



2. Check the requirements.txt file to confirm installing the latest flask version.

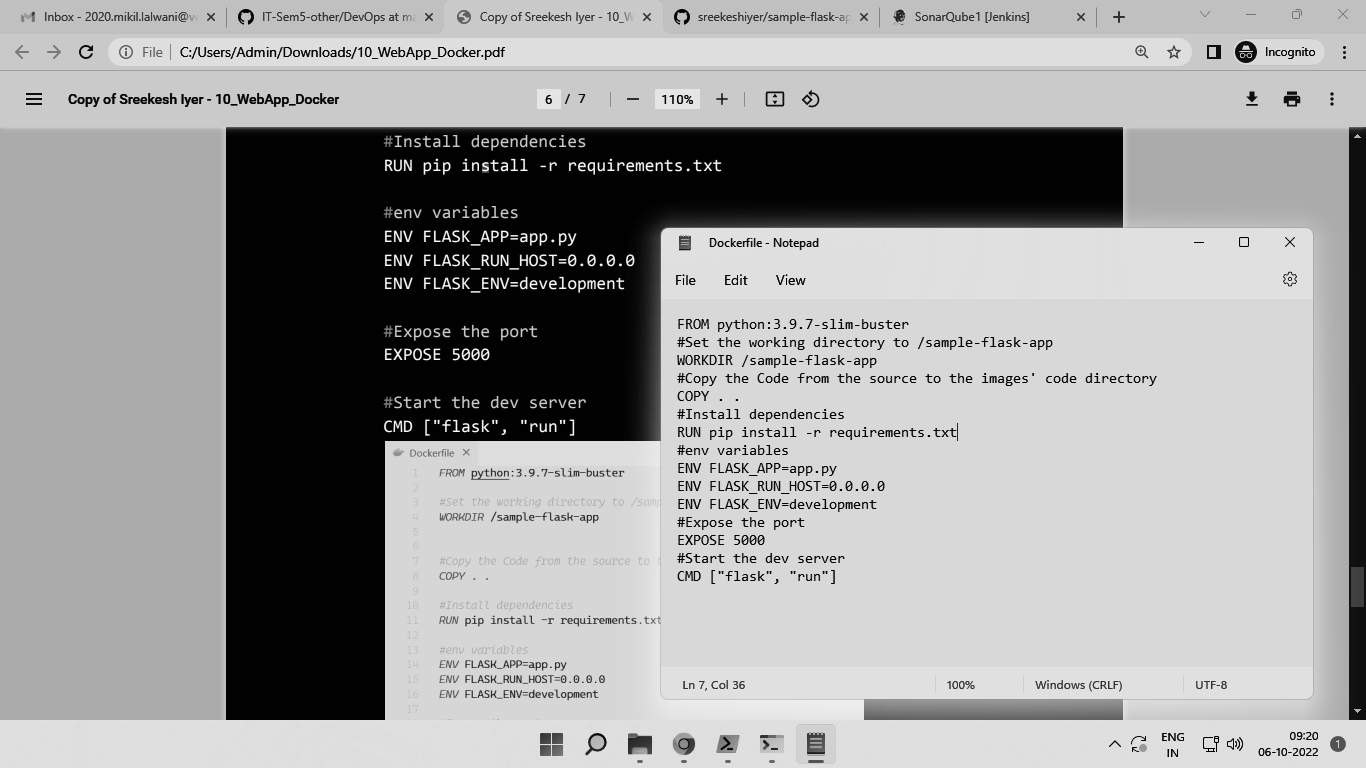


3. You can view the code in app.py



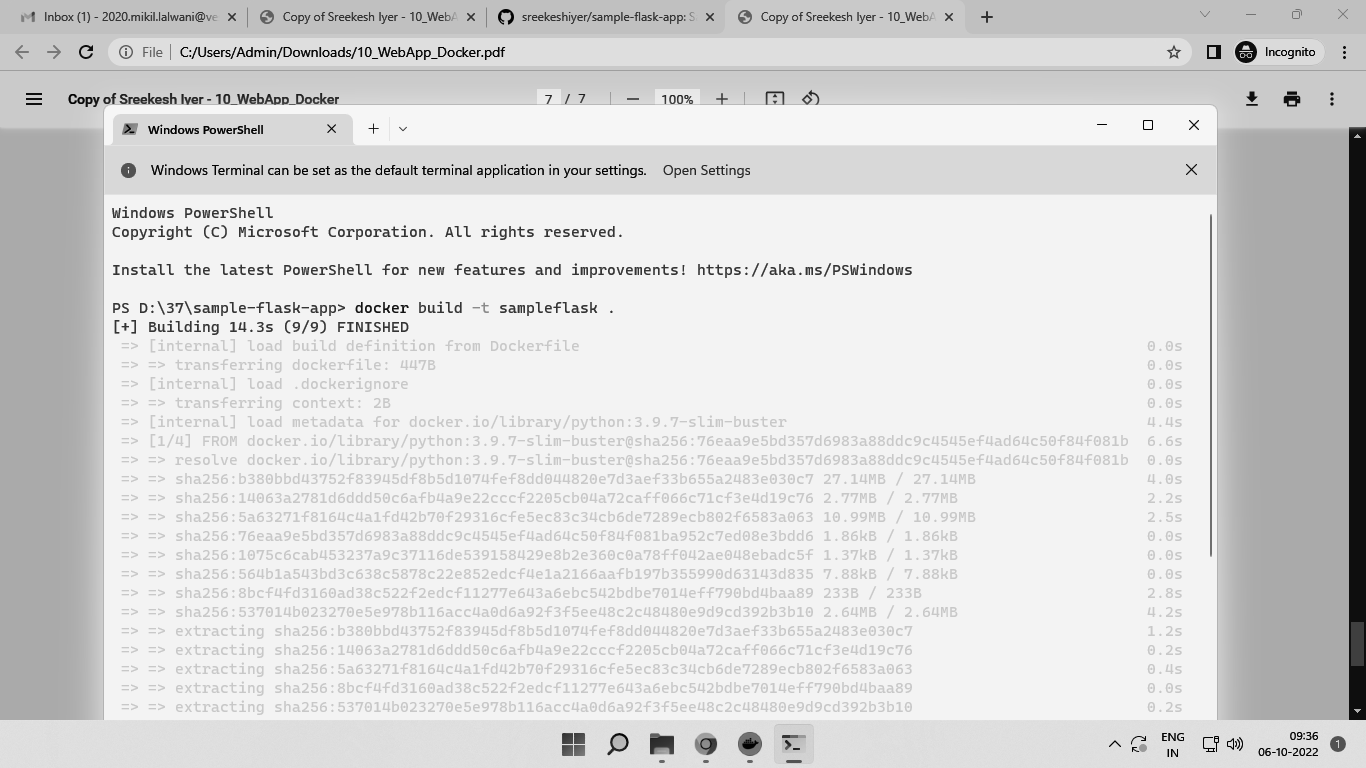
As you can see, we are creating a flask app that simply returns a greeting when you run it.

4. Create a new file in the same folder, named ‘Dockerfile’. Add the following contents to the file, like so.

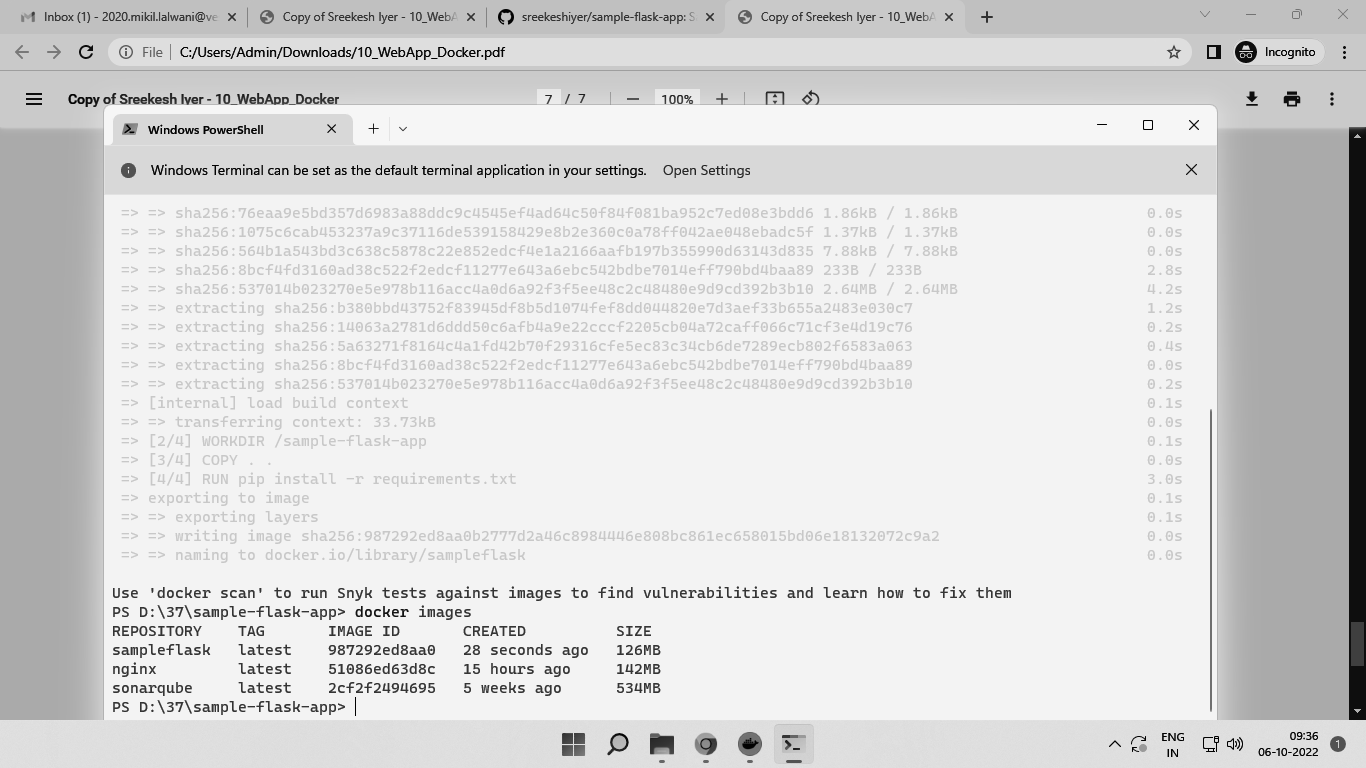


This Dockerfile will be used to create a Docker image for our sample app.

5. Open the terminal and run docker build -t .

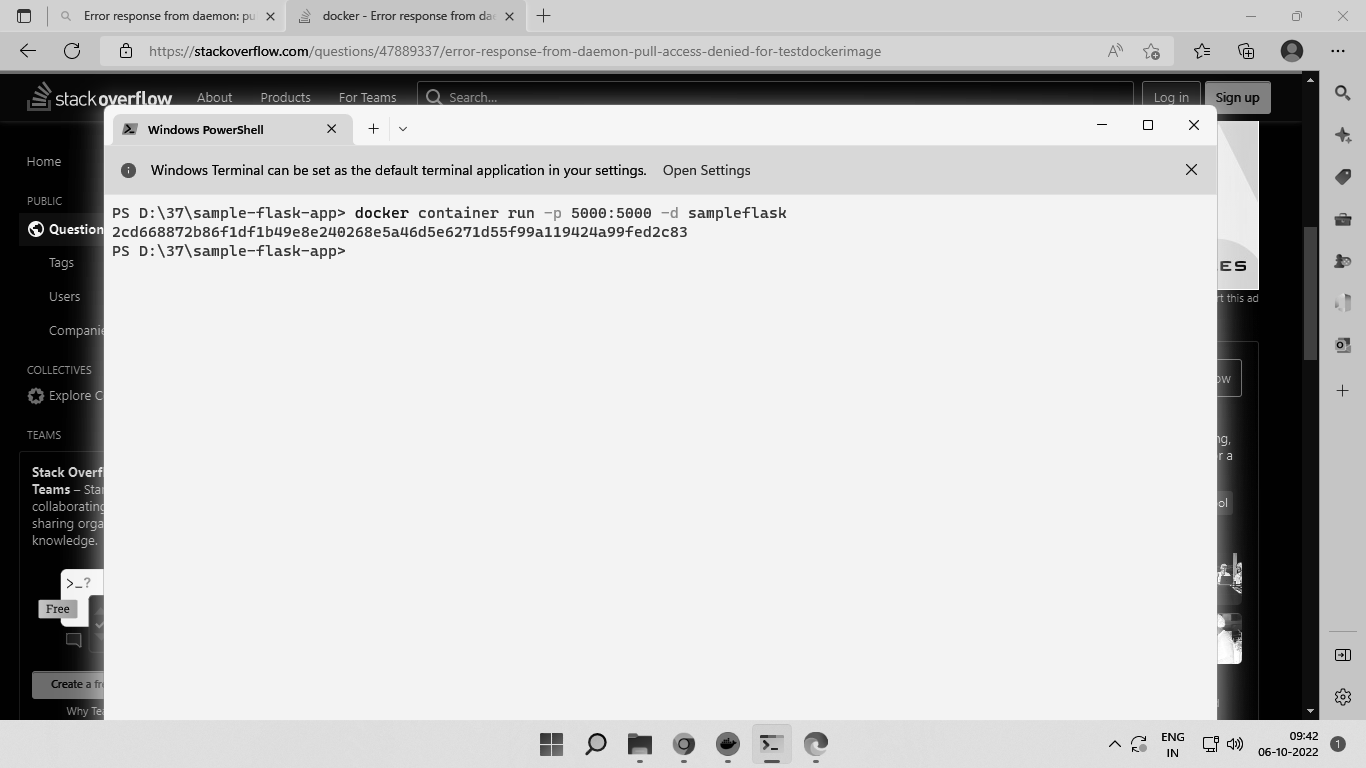


6. Once the image is successfully created, you can use docker images to check it.

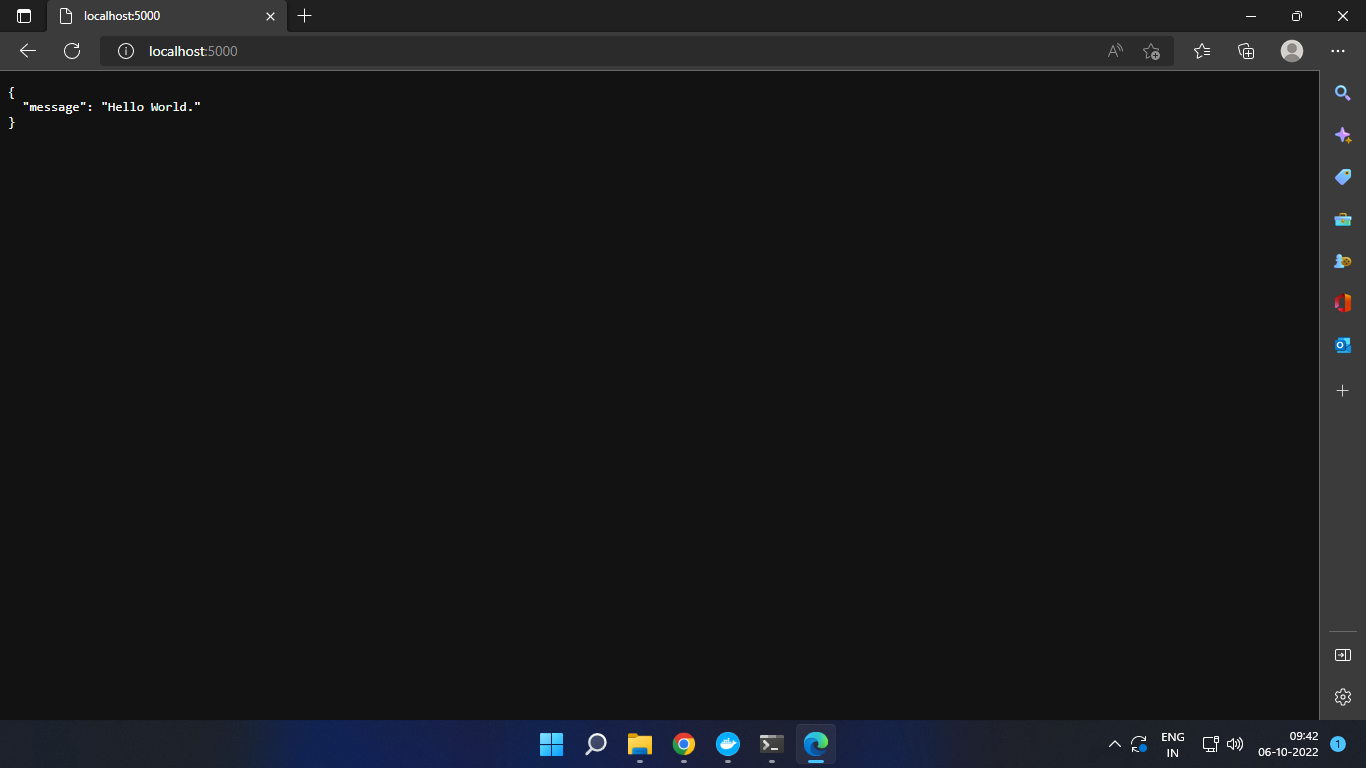


7. After that, we can use this image to run a container using -

docker container run -p 5000:5000 -d sampleflask



8. Check localhost:5000 in your browser and you can see your app running.



**Conclusion**:

Thus, we learned about Dockerfile, created and wrote Dockerfile for a sample Flask WebApp and built a Docker Image using it.