Experiment No. 1.3

Student Name: Ambika Sharma **UID: 22MCC20022**
Branch: MCA CC & Devops **Section/Group – 22MCD - 1/A**
Semester: III **Date of Performance: 25/09/23**
Subject Name: Containerization with Docker **Subject Code: - 22CAH-742**

1. Aims/Overview for the practical:

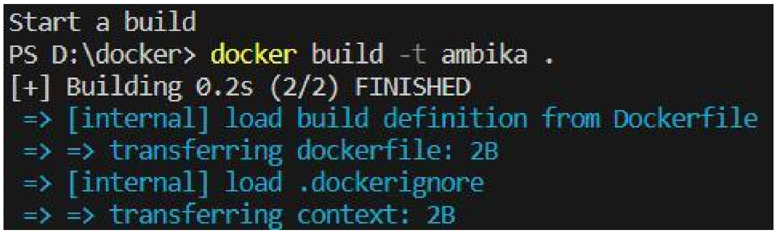
Deploying Docker images as Stateless Containers.

2. Steps for experiment/practical:

Step 1. Install Docker Desktop: If you haven't already, download and install Docker Desktop on your computer. You can find the installer for your specific operating system on the Docker website.
Step 2. **Create a Dockerfile:** To deploy a Docker image, you need to create a Dockerfile. This file contains instructions for building your container image. You can use a text editor to create a Dockerfile in the root directory of your

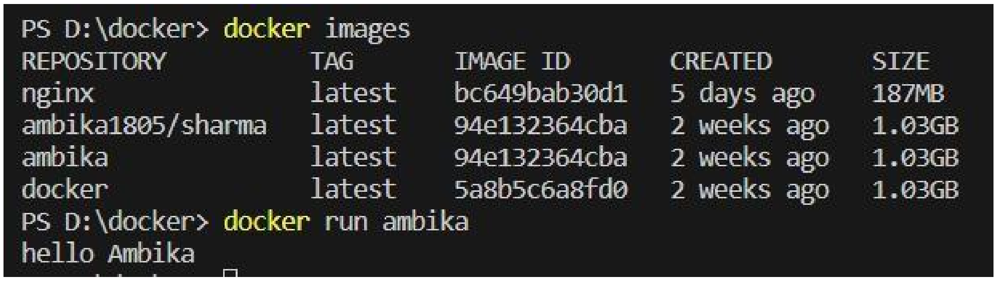
```
FROM python:3
ADD helloworld.py /
RUN pip install flask
RUN pip install flask_restful
EXPOSE 3333
CMD ["python", "./helloworld.py" ]
do
```

Step 3. Build the Docker Image: Open a terminal and navigate to the directory where your Dockerfile is located. Run the following command to build your Docker image, replacing my-app with a suitable name for your image:
docker build -t my-app .



```
Start a build
PS D:\docker> docker build -t ambika .
[+] Building 0.2s (2/2) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 2B
 => [internal] load .dockerignore
 => => transferring context: 2B
```

Step 4. Run Docker Containers: Once your image is built, you can run containers from it. Here's a basic command to run a stateless container: ```
docker run -d -p 8080:3000 my-app

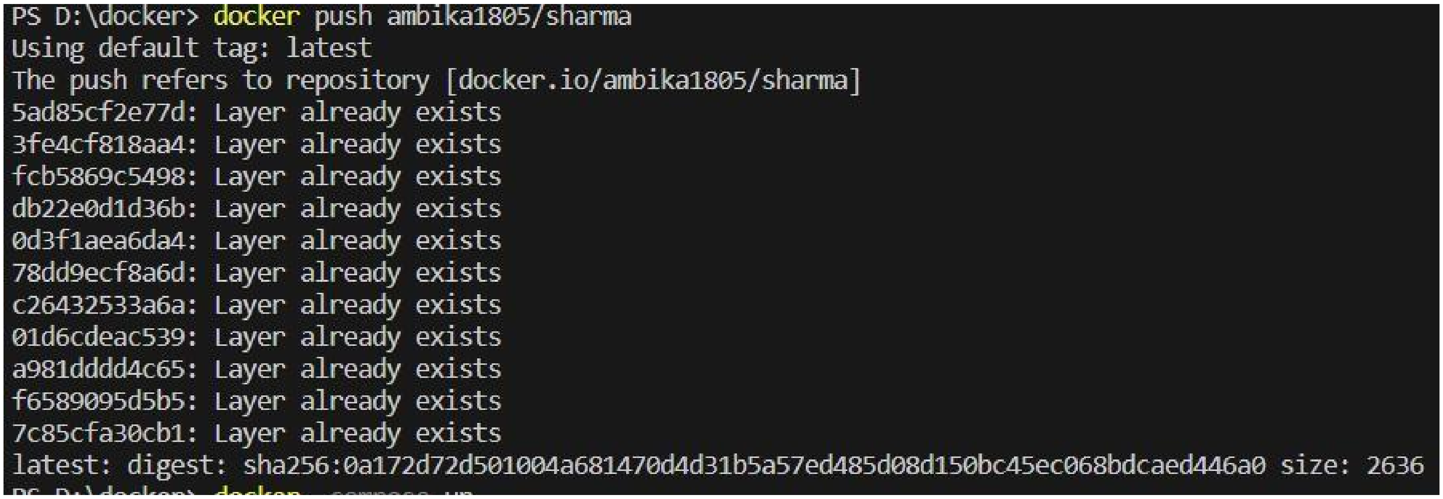


```
PS D:\docker> docker images
REPOSITORY          TAG       IMAGE ID       CREATED       SIZE
nginx               latest    bc649bab30d1   5 days ago    187MB
ambika1805/sharma   latest    94e132364cba   2 weeks ago   1.03GB
ambika              latest    94e132364cba   2 weeks ago   1.03GB
docker              latest    5a8b5c6a8fd0   2 weeks ago   1.03GB
PS D:\docker> docker run ambika
hello Ambika
```

Step 5. Access the Application: Open a web browser or a tool like curl to access your application at http://localhost:8080 if you mapped it to port 8080. ```

Step 6. Scaling and Managing Containers: To scale your application, you can run multiple containers from the same image, each using a different host port.

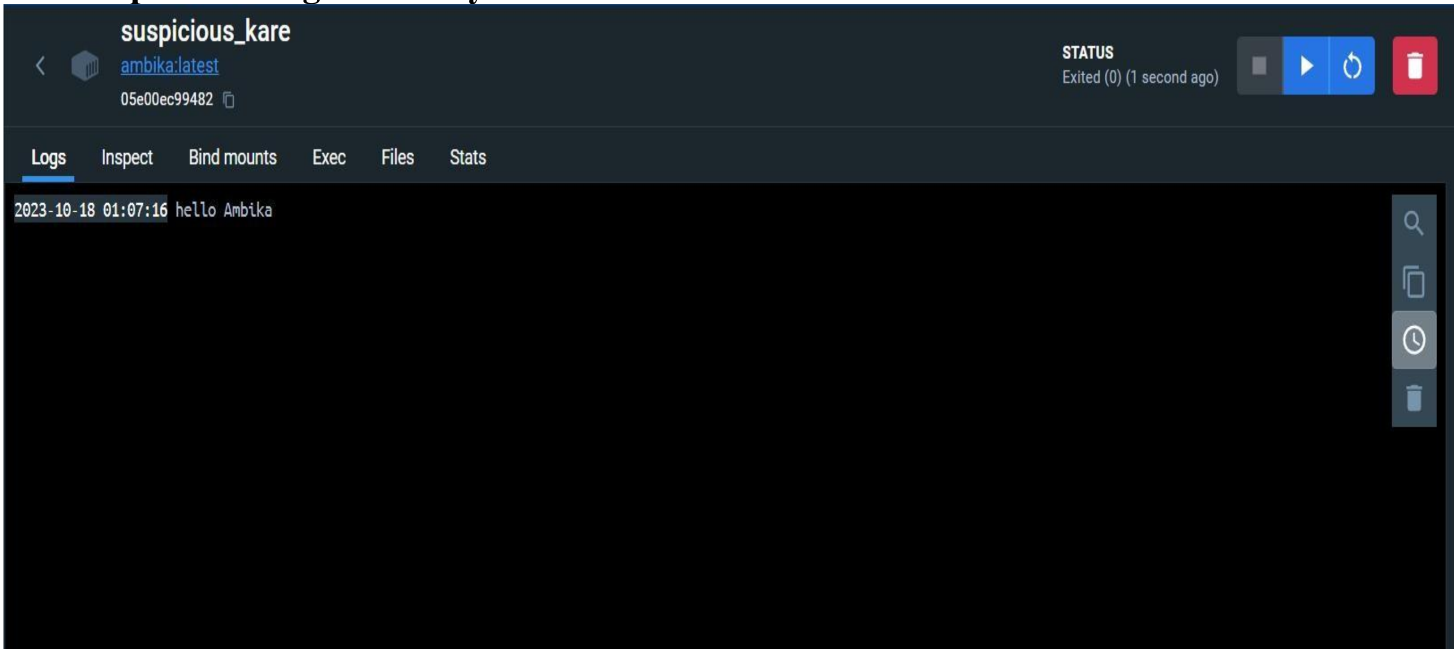
Step 7. Push Image to a Registry (Optional): If you plan to share your Docker image with others or deploy it on multiple hosts, you can push the image to a container registry like Docker Hub. Use the docker push command for this:
docker push your-registry/image-name:tag



```
PS D:\docker> docker push ambika1805/sharma
Using default tag: latest
The push refers to repository [docker.io/ambika1805/sharma]
5ad85cf2e77d: Layer already exists
3fe4cf818aa4: Layer already exists
fcb5869c5498: Layer already exists
db22e0d1d36b: Layer already exists
0d3f1aea6da4: Layer already exists
78dd9ecf8a6d: Layer already exists
c26432533a6a: Layer already exists
01d6cdeac539: Layer already exists
a981dddd4c65: Layer already exists
f6589095d5b5: Layer already exists
7c85cfa30cb1: Layer already exists
latest: digest: sha256:0a172d72d501004a681470d4d31b5a57ed485d08d150bc45ec068bdcaed446a0 size: 2636
```

Step 8. Deploy Stateless Containers:
=> Using Docker Compose (for local development and testing):
• Create a docker-compose.yml file that defines your service, including the image, environment variables, ports, and any volume mounts.
• Run the application using docker-compose up**.**

4. Result/Output/Writing Summary:



```
2023-10-18 01:07:16 hello Ambika
```

5. Learning outcomes (What I have learned):

a) Learned to build a docker image by creating a Dockerfile, a docker-compose.yml and other required files.
b) Learned to push image to the docker hub or private registry.
c) Learned to deploy docker image as stateless containers.