Experiment 10

Aim: To learn Dockerfile instructions, build an image for a sample web application using DockerFile.

THEORY:

What is Docker?

Docker is an open-source platform that enables developers to build, ship, and run applications inside containers. These containers are lightweight, portable, and contain everything the software needs to run: code, runtime, system libraries, and configurations.

Docker helps eliminate the problem of "it works on my machine" by ensuring the application behaves the same in development, testing, and production.

Benefits of Docker

1. Portability
   - Docker containers can run on any platform that supports Docker (Windows, macOS, Linux, cloud environments).
   - Applications behave consistently across different environments.
2. Efficiency
   - Containers share the host operating system’s kernel, reducing overhead compared to virtual machines.
   - They start quickly and use less memory.
3. Isolation
   - Each container runs in its own isolated environment, preventing conflicts between applications.
4. Scalability
   - Applications can be scaled horizontally by launching multiple containers.
   - Useful for microservices-based architectures and load balancing.

5. Consistency

- Docker ensures that the same code runs in the same environment across all stages of development and deployment.

Understanding Dockerfiles

A **Dockerfile** is a plain text script containing a set of **instructions** used to create a Docker image. These instructions are executed sequentially by Docker Engine when the image is built.

Although directives in a Dockerfile are case-insensitive, it is a good practice to write them in **UPPERCASE** for clarity and standardization.

Steps to Build a Dockerfile for a Python Flask App

We'll build a Dockerfile for a simple **Flask-based web application** that displays "Hello, World!" on the homepage.

1. Specify Base Image

```
FROM python:3.11-slim
```

This specifies the base image. We use a minimal Python 3.11 image. The base image includes the Python interpreter and essential libraries.

2. Set Working Directory

```
WORKDIR /app
```

Sets /app as the working directory inside the container. All subsequent commands (like RUN, COPY, CMD) operate from here.

3. Install Dependencies

```
RUN pip install Flask==2.2.2
```

Installs the Flask framework required by the application.

4. Copy Application Files

COPY . /app

Copies the project files from your host system to the /app directory in the container.

5. Set Environment Variable

ENV FLASK_APP=app.py

Tells Flask which file contains the application instance.

6. Define Default Command

CMD ["flask", "run", "--host=0.0.0.0", "--port=5000"]

Specifies the command to run the Flask app. --host=0.0.0.0 ensures the app is accessible externally via the container. --port=5000 runs the server on port 5000.

7. Create a .dockerignore File

Before building the image, create a .dockerignore file to prevent unnecessary files (like the Dockerfile itself) from being copied into the image.

Dockerfile

This improves security and reduces image size.

Building and Running the Docker Image

1. **Build the Docker image**

   docker build -t sample-flask-app:v1.

   (-t tags the image as sample-flask-app with version v1.)

Aim: To learn Dockerfile instructions, build an image for a sample web application using DockerFile.

2. **Verify Image Creation**

```
docker images
```

3. Run the Container

```
docker run -d -p 5000:5000 sample-flask-app:v1
```

-d: runs container in detached mode (background)

-p: maps port 5000 of the host to port 5000 of the container

4. **Check Running Containers**

```
docker ps
```

5. **Access the App**
   Open a web browser and navigate to:

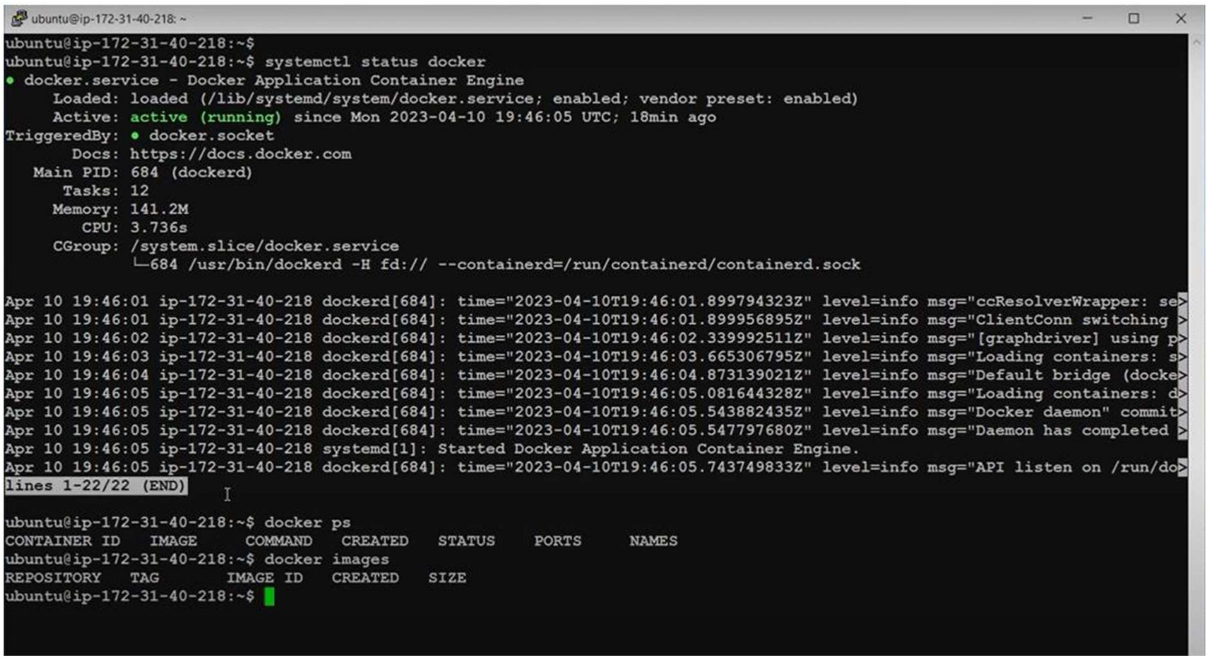
   http://localhost:5000

   You should see: **Hello, World!**

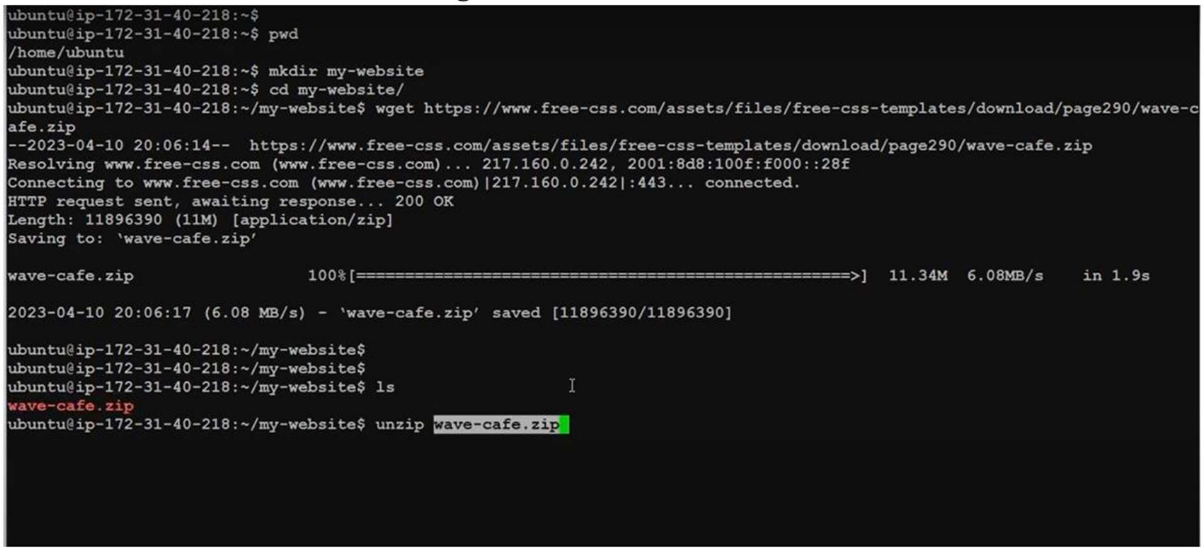
Experiment 10

Aim: To learn Dockerfile instructions, build an image for a sample web application using DockerFile.

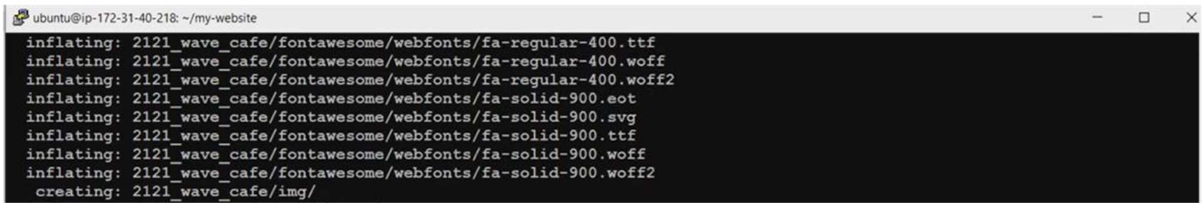
IMPLEMENTATION:



```
ubuntu@ip-172-31-40-218: ~
ubuntu@ip-172-31-40-218:~$
ubuntu@ip-172-31-40-218:~$ systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Mon 2023-04-10 19:46:05 UTC; 18min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 684 (dockerd)
      Tasks: 12
     Memory: 141.2M
        CPU: 3.736s
     CGroup: /system.slice/docker.service
             └─684 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Apr 10 19:46:01 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:01.899794323Z" level=info msg="ccResolverWrapper: se>
Apr 10 19:46:01 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:01.899956895Z" level=info msg="ClientConn switching >
Apr 10 19:46:02 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:02.339992511Z" level=info msg="[graphdriver] using p>
Apr 10 19:46:03 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:03.665306795Z" level=info msg="Loading containers: s>
Apr 10 19:46:04 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:04.873139021Z" level=info msg="Default bridge (docke>
Apr 10 19:46:05 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:05.081644328Z" level=info msg="Loading containers: d>
Apr 10 19:46:05 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:05.543882435Z" level=info msg="Docker daemon" commit>
Apr 10 19:46:05 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:05.547797680Z" level=info msg="Daemon has completed >
Apr 10 19:46:05 ip-172-31-40-218 systemd[1]: Started Docker Application Container Engine.
Apr 10 19:46:05 ip-172-31-40-218 dockerd[684]: time="2023-04-10T19:46:05.743749833Z" level=info msg="API listen on /run/do>
lines 1-22/22 (END)

ubuntu@ip-172-31-40-218:~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@ip-172-31-40-218:~$ docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
ubuntu@ip-172-31-40-218:~$
```



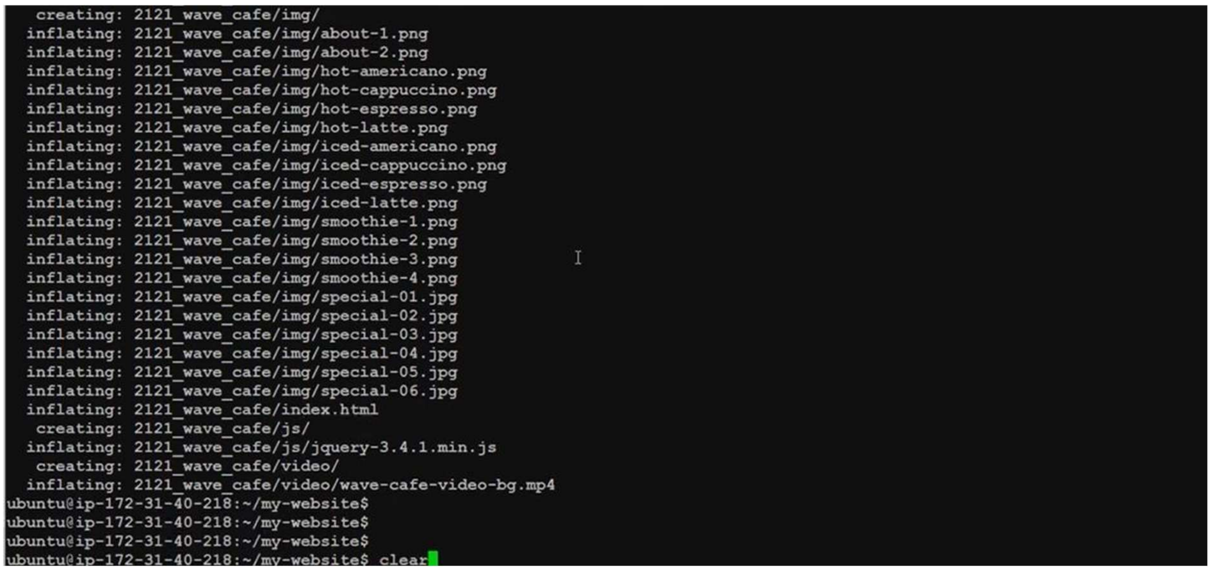
```
ubuntu@ip-172-31-40-218:~$
ubuntu@ip-172-31-40-218:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-40-218:~$ mkdir my-website
ubuntu@ip-172-31-40-218:~$ cd my-website/
ubuntu@ip-172-31-40-218:~/my-website$ wget https://www.free-css.com/assets/files/free-css-templates/download/page290/wave-c
afe.zip
--2023-04-10 20:06:14--  https://www.free-css.com/assets/files/free-css-templates/download/page290/wave-cafe.zip
Resolving www.free-css.com (www.free-css.com)... 217.160.0.242, 2001:8d8:100f:f000::28f
Connecting to www.free-css.com (www.free-css.com)|217.160.0.242|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 11896390 (11M) [application/zip]
Saving to: 'wave-cafe.zip'

wave-cafe.zip                 100%[===================================================>]  11.34M  6.08MB/s    in 1.9s

2023-04-10 20:06:17 (6.08 MB/s) - 'wave-cafe.zip' saved [11896390/11896390]

ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$ ls
wave-cafe.zip
ubuntu@ip-172-31-40-218:~/my-website$ unzip wave-cafe.zip
```



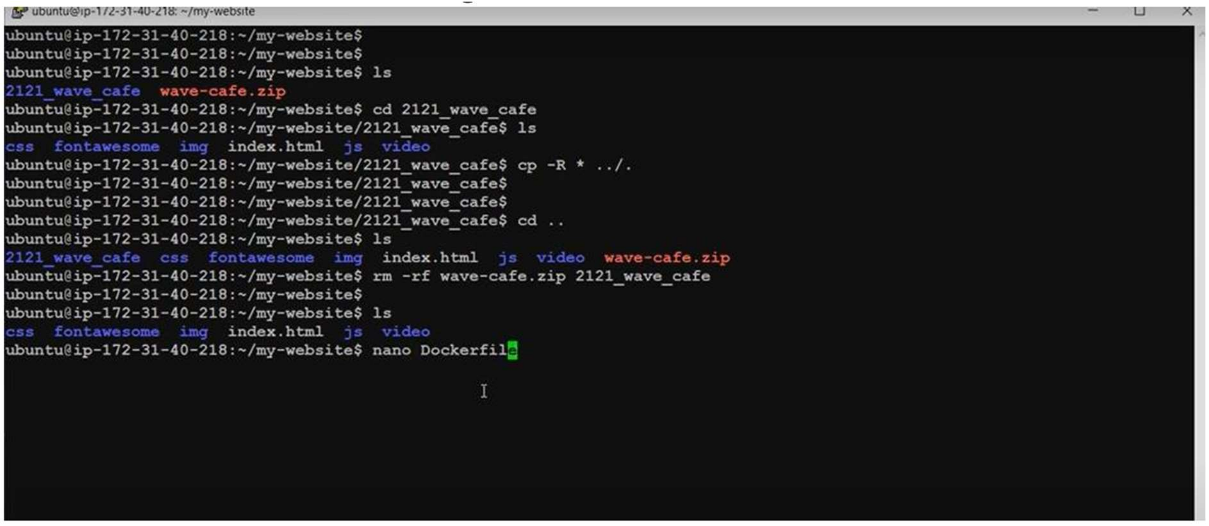
```
ubuntu@ip-172-31-40-218: ~/my-website
  inflating: 2121_wave_cafe/fontawesome/webfonts/fa-regular-400.ttf
  inflating: 2121_wave_cafe/fontawesome/webfonts/fa-regular-400.woff
  inflating: 2121_wave_cafe/fontawesome/webfonts/fa-regular-400.woff2
  inflating: 2121_wave_cafe/fontawesome/webfonts/fa-solid-900.eot
  inflating: 2121_wave_cafe/fontawesome/webfonts/fa-solid-900.svg
  inflating: 2121_wave_cafe/fontawesome/webfonts/fa-solid-900.ttf
  inflating: 2121_wave_cafe/fontawesome/webfonts/fa-solid-900.woff
  inflating: 2121_wave_cafe/fontawesome/webfonts/fa-solid-900.woff2
   creating: 2121_wave_cafe/img/
```

Experiment 10

Aim: To learn Dockerfile instructions, build an image for a sample web application using DockerFile.

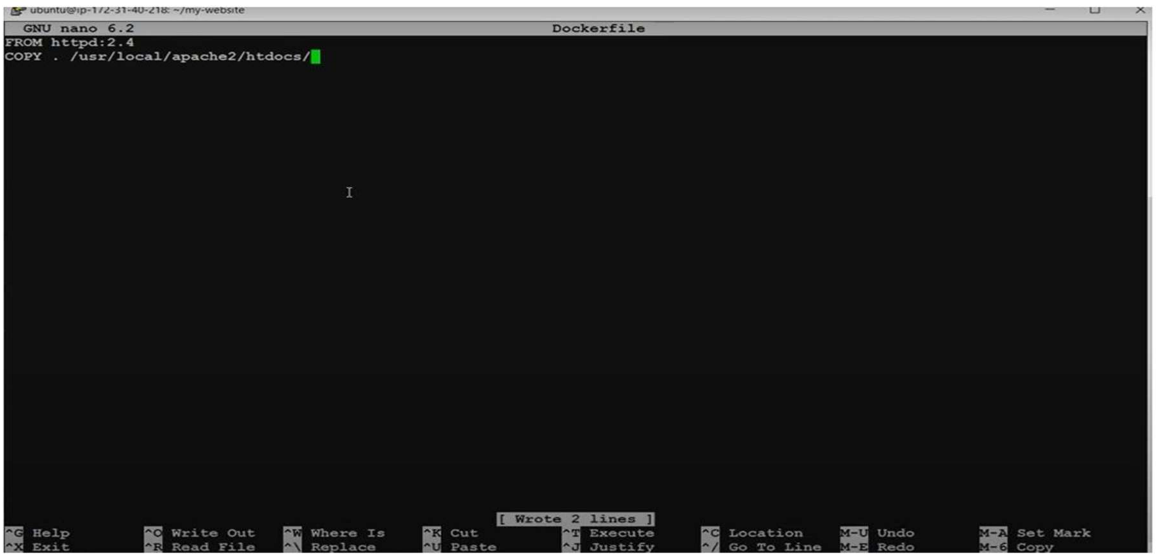


```
   creating: 2121_wave_cafe/img/
  inflating: 2121_wave_cafe/img/about-1.png
  inflating: 2121_wave_cafe/img/about-2.png
  inflating: 2121_wave_cafe/img/hot-americano.png
  inflating: 2121_wave_cafe/img/hot-cappuccino.png
  inflating: 2121_wave_cafe/img/hot-espresso.png
  inflating: 2121_wave_cafe/img/hot-latte.png
  inflating: 2121_wave_cafe/img/iced-americano.png
  inflating: 2121_wave_cafe/img/iced-cappuccino.png
  inflating: 2121_wave_cafe/img/iced-espresso.png
  inflating: 2121_wave_cafe/img/iced-latte.png
  inflating: 2121_wave_cafe/img/smoothie-1.png
  inflating: 2121_wave_cafe/img/smoothie-2.png
  inflating: 2121_wave_cafe/img/smoothie-3.png
  inflating: 2121_wave_cafe/img/smoothie-4.png
  inflating: 2121_wave_cafe/img/special-01.jpg
  inflating: 2121_wave_cafe/img/special-02.jpg
  inflating: 2121_wave_cafe/img/special-03.jpg
  inflating: 2121_wave_cafe/img/special-04.jpg
  inflating: 2121_wave_cafe/img/special-05.jpg
  inflating: 2121_wave_cafe/img/special-06.jpg
  inflating: 2121_wave_cafe/index.html
   creating: 2121_wave_cafe/js/
  inflating: 2121_wave_cafe/js/jquery-3.4.1.min.js
   creating: 2121_wave_cafe/video/
  inflating: 2121_wave_cafe/video/wave-cafe-video-bg.mp4
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$ clear
```



ubuntu@ip-172-31-40-218: ~/my-website

```
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$ ls
2121_wave_cafe  wave-cafe.zip
ubuntu@ip-172-31-40-218:~/my-website$ cd 2121_wave_cafe
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ ls
css  fontawesome  img  index.html  js  video
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ cp -R * ../.
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ cd ..
ubuntu@ip-172-31-40-218:~/my-website$ ls
2121_wave_cafe  css  fontawesome  img  index.html  js  video  wave-cafe.zip
ubuntu@ip-172-31-40-218:~/my-website$ rm -rf wave-cafe.zip 2121_wave_cafe
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$ ls
css  fontawesome  img  index.html  js  video
ubuntu@ip-172-31-40-218:~/my-website$ nano Dockerfile
```

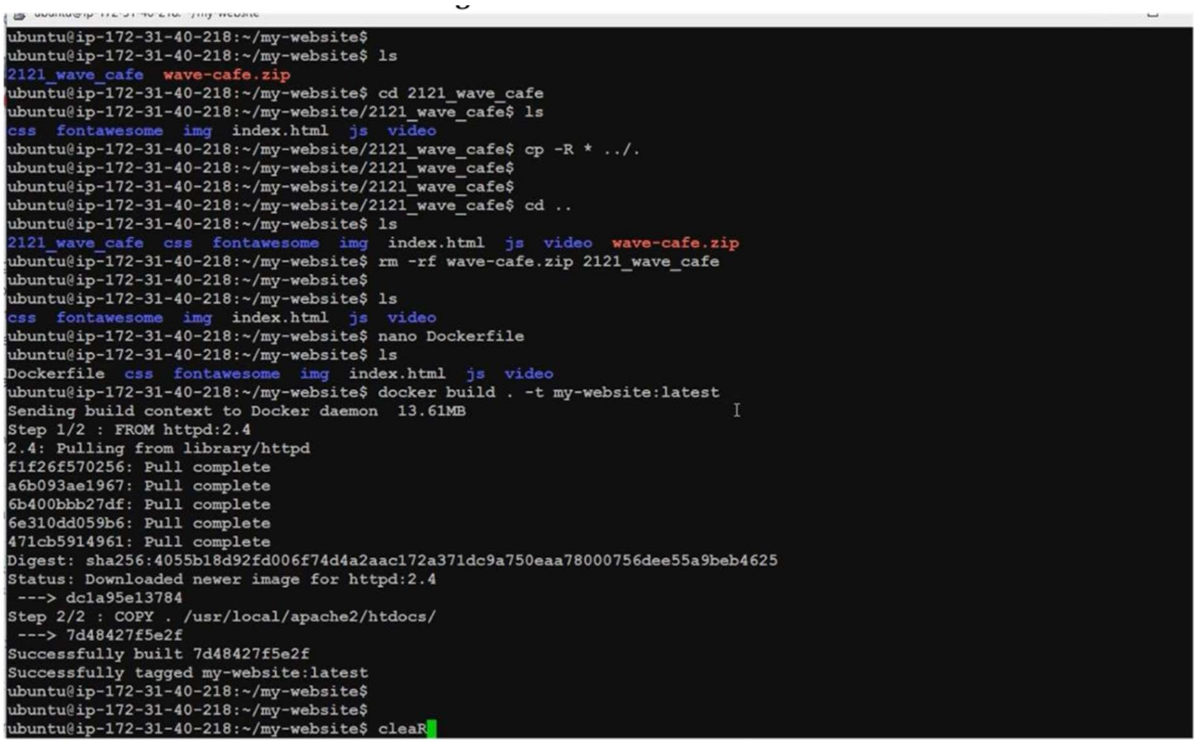


ubuntu@ip-172-31-40-218: ~/my-website

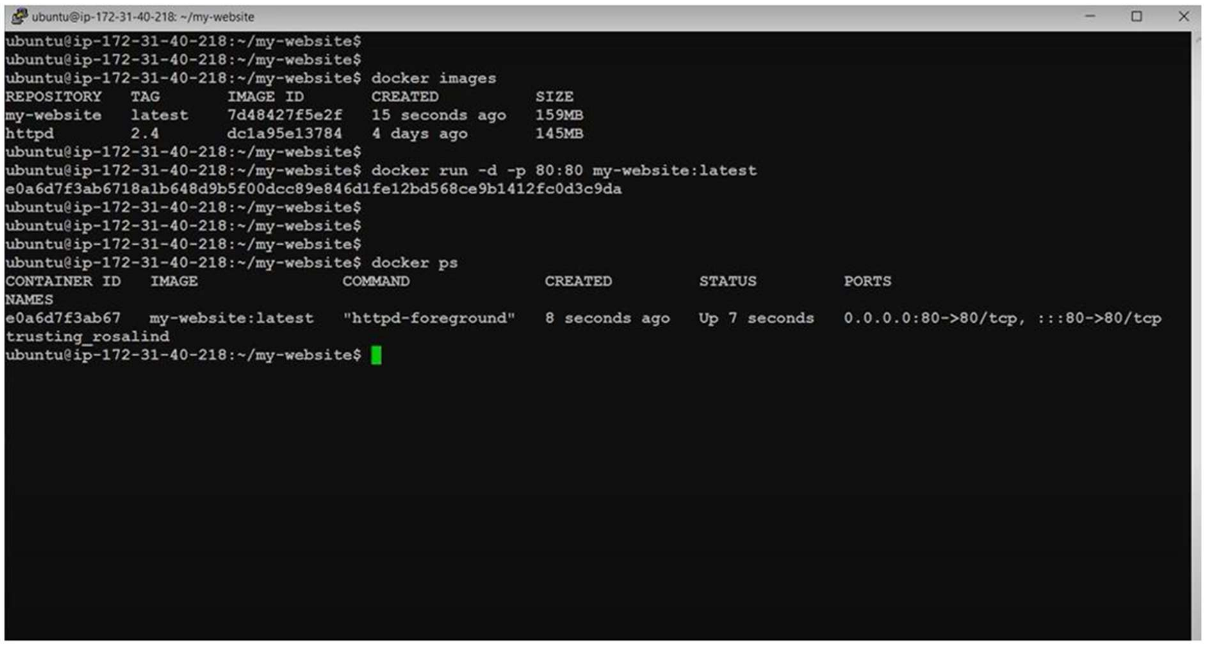
```
  GNU nano 6.2                            Dockerfile
FROM httpd:2.4
COPY . /usr/local/apache2/htdocs/

                                       [ Wrote 2 lines ]
^G Help        ^O Write Out   ^W Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy
```

Experiment 10

Aim: To learn Dockerfile instructions, build an image for a sample web application using DockerFile.



```
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$ ls
2121_wave_cafe  wave-cafe.zip
ubuntu@ip-172-31-40-218:~/my-website$ cd 2121_wave_cafe
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ ls
css  fontawesome  img  index.html  js  video
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ cp -R * ../.
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$
ubuntu@ip-172-31-40-218:~/my-website/2121_wave_cafe$ cd ..
ubuntu@ip-172-31-40-218:~/my-website$ ls
2121_wave_cafe  css  fontawesome  img  index.html  js  video  wave-cafe.zip
ubuntu@ip-172-31-40-218:~/my-website$ rm -rf wave-cafe.zip 2121_wave_cafe
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$ ls
css  fontawesome  img  index.html  js  video
ubuntu@ip-172-31-40-218:~/my-website$ nano Dockerfile
ubuntu@ip-172-31-40-218:~/my-website$ ls
Dockerfile  css  fontawesome  img  index.html  js  video
ubuntu@ip-172-31-40-218:~/my-website$ docker build . -t my-website:latest
Sending build context to Docker daemon  13.61MB
Step 1/2 : FROM httpd:2.4
2.4: Pulling from library/httpd
f1f26f570256: Pull complete
a6b093ae1967: Pull complete
6b400bbb27df: Pull complete
6e310dd059b6: Pull complete
471cb5914961: Pull complete
Digest: sha256:4055b18d92fd006f74d4a2aac172a371dc9a750eaa78000756dee55a9beb4625
Status: Downloaded newer image for httpd:2.4
 ---> dc1a95e13784
Step 2/2 : COPY . /usr/local/apache2/htdocs/
 ---> 7d48427f5e2f
Successfully built 7d48427f5e2f
Successfully tagged my-website:latest
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$ cleaR
```



ubuntu@ip-172-31-40-218: ~/my-website

```
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$ docker images
REPOSITORY   TAG       IMAGE ID       CREATED          SIZE
my-website   latest    7d48427f5e2f   15 seconds ago   159MB
httpd        2.4       dc1a95e13784   4 days ago       145MB
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$ docker run -d -p 80:80 my-website:latest
e0a6d7f3ab6718a1b648d9b5f00dcc89e846d1fe12bd568ce9b1412fc0d3c9da
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$
ubuntu@ip-172-31-40-218:~/my-website$ docker ps
CONTAINER ID   IMAGE               COMMAND              CREATED         STATUS         PORTS
NAMES
e0a6d7f3ab67   my-website:latest   "httpd-foreground"   8 seconds ago   Up 7 seconds   0.0.0.0:80->80/tcp, :::80->80/tcp
trusting_rosalind
ubuntu@ip-172-31-40-218:~/my-website$
```

Conclusion: Thus, we have successfully learnt Dockerfile instructions & build an image for a sample web application using DockerFile.