DevOps Lab

Assignment 8:

Aim: To learn Dockerfile instructions, build an image for a sample web application using Dockerfile which will be hosted using nginx and apache2.

Theory & Execution:

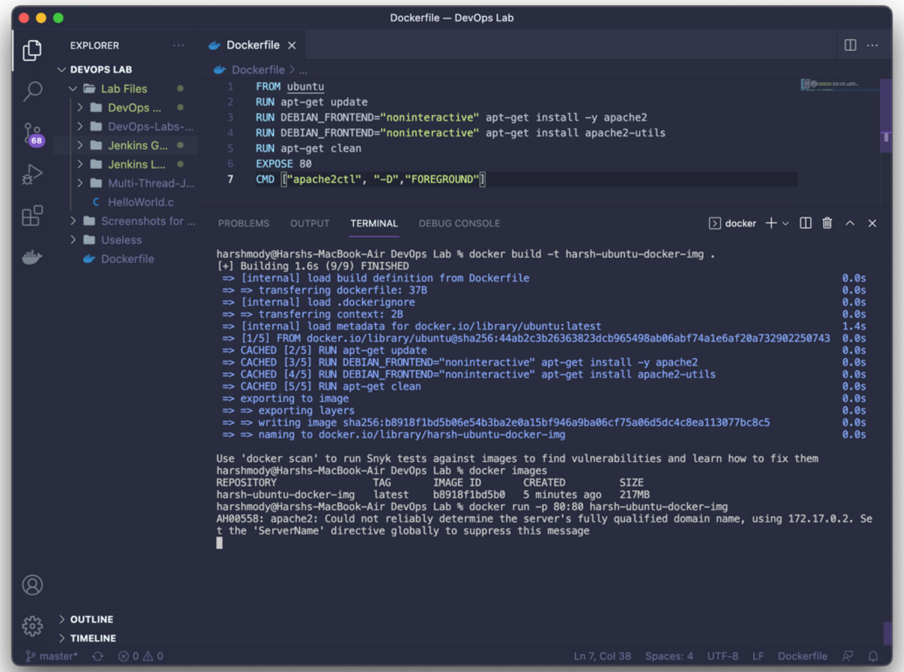
Docker can build images automatically by reading the instructions from a Dockerfile.
A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image. Using docker build users can create an automated build that executes several command-line instructions in succession.

A Docker image consists of read-only layers each of which represents a Dockerfile instruction. The layers are stacked and each one is a delta of the changes from the previous layer. Consider this Dockerfile:

Each instruction creates one layer:

- FROM creates a layer from the ubuntu:18.04 Docker image.
- COPY adds files from your Docker client's current directory.
- RUN builds your application with make.
- CMD specifies what command to run within the container.

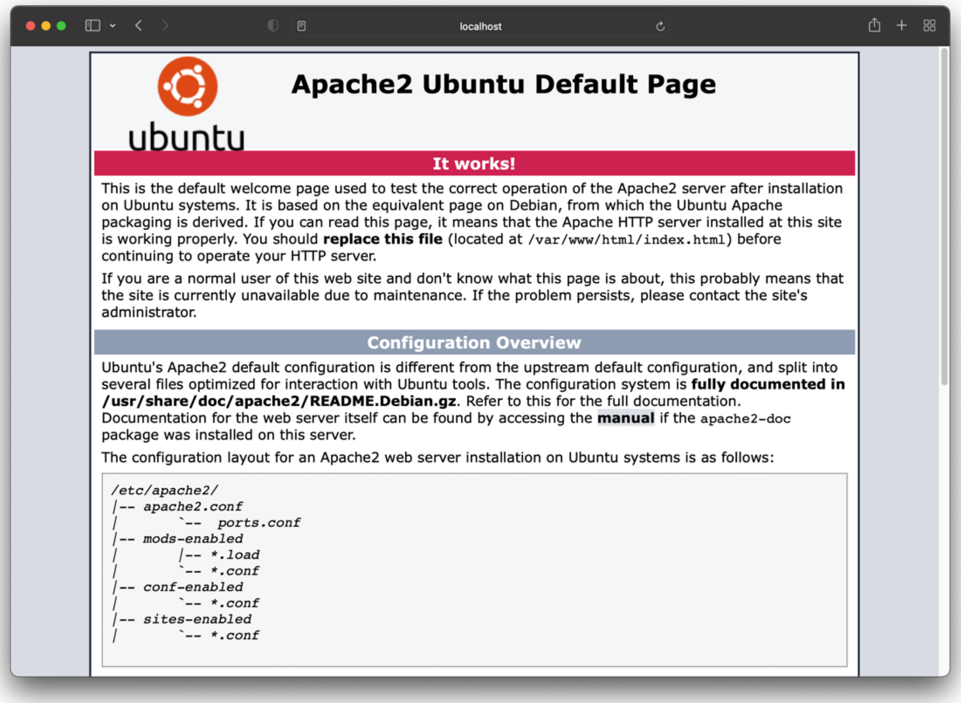
To run a web app using apache2, we used an Ubuntu image as our base image and wrote a custom Dockerfile to host the sample apache web server homepage.



```
FROM ubuntu
RUN apt-get update
RUN DEBIAN_FRONTEND="noninteractive" apt-get install -y apache2
RUN DEBIAN_FRONTEND="noninteractive" apt-get install apache2-utils
RUN apt-get clean
EXPOSE 80
CMD ["apache2ctl", "-D","FOREGROUND"]
```

```
harshmody@Harshs-MacBook-Air DevOps Lab % docker build -t harsh-ubuntu-docker-img .
[+] Building 1.6s (9/9) FINISHED
 => [internal] load build definition from Dockerfile                                          0.0s
 => => transferring dockerfile: 37B                                                           0.0s
 => [internal] load .dockerignore                                                             0.0s
 => => transferring context: 2B                                                               0.0s
 => [internal] load metadata for docker.io/library/ubuntu:latest                              1.4s
 => [1/5] FROM docker.io/library/ubuntu@sha256:44ab2c3b26363823dcb965498ab06abf74a1e6af20a732902250743  0.0s
 => CACHED [2/5] RUN apt-get update                                                           0.0s
 => CACHED [3/5] RUN DEBIAN_FRONTEND="noninteractive" apt-get install -y apache2              0.0s
 => CACHED [4/5] RUN DEBIAN_FRONTEND="noninteractive" apt-get install apache2-utils           0.0s
 => CACHED [5/5] RUN apt-get clean                                                            0.0s
 => exporting to image                                                                        0.0s
 => => exporting layers                                                                       0.0s
 => => writing image sha256:b8918f1bd5b06e54b3ba2e0a15bf946a9ba06cf75a06d5dc4c8ea113077bc8c5  0.0s
 => => naming to docker.io/library/harsh-ubuntu-docker-img                                    0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
harshmody@Harshs-MacBook-Air DevOps Lab % docker images
REPOSITORY                TAG       IMAGE ID       CREATED         SIZE
harsh-ubuntu-docker-img   latest    b8918f1bd5b0   5 minutes ago   217MB
harshmody@Harshs-MacBook-Air DevOps Lab % docker run -p 80:80 harsh-ubuntu-docker-img
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.17.0.2. Se
t the 'ServerName' directive globally to suppress this message
```

We can check if our container was created successfully by visiting http://localhost:80/



Apache2 Ubuntu Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

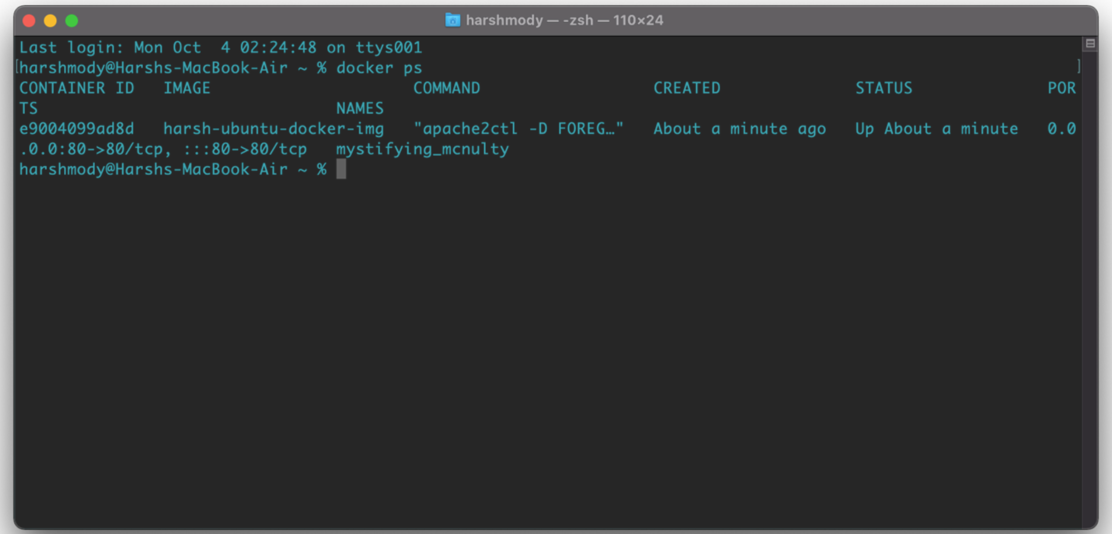
If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

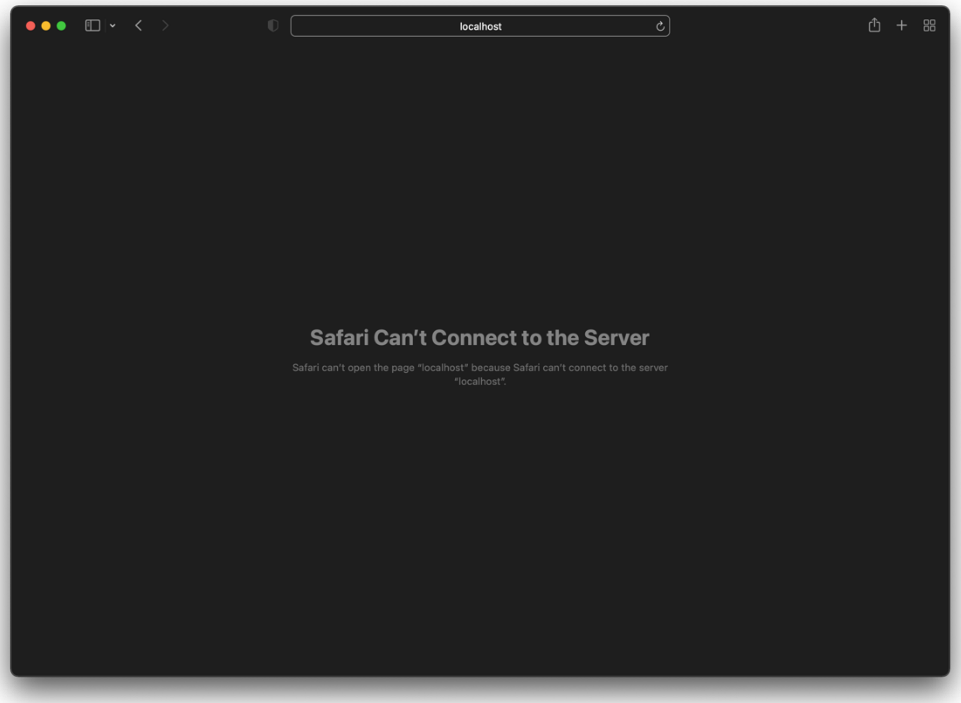
The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```



```
Last login: Mon Oct  4 02:24:48 on ttys001
harshmody@Harshs-MacBook-Air ~ % docker ps
CONTAINER ID   IMAGE                    COMMAND                  CREATED              STATUS              POR
TS                                NAMES
e9004099ad8d   harsh-ubuntu-docker-img   "apache2ctl -D FOREG…"   About a minute ago   Up About a minute   0.0
.0.0:80->80/tcp, :::80->80/tcp   mystifying_mcnulty
harshmody@Harshs-MacBook-Air ~ % 
```

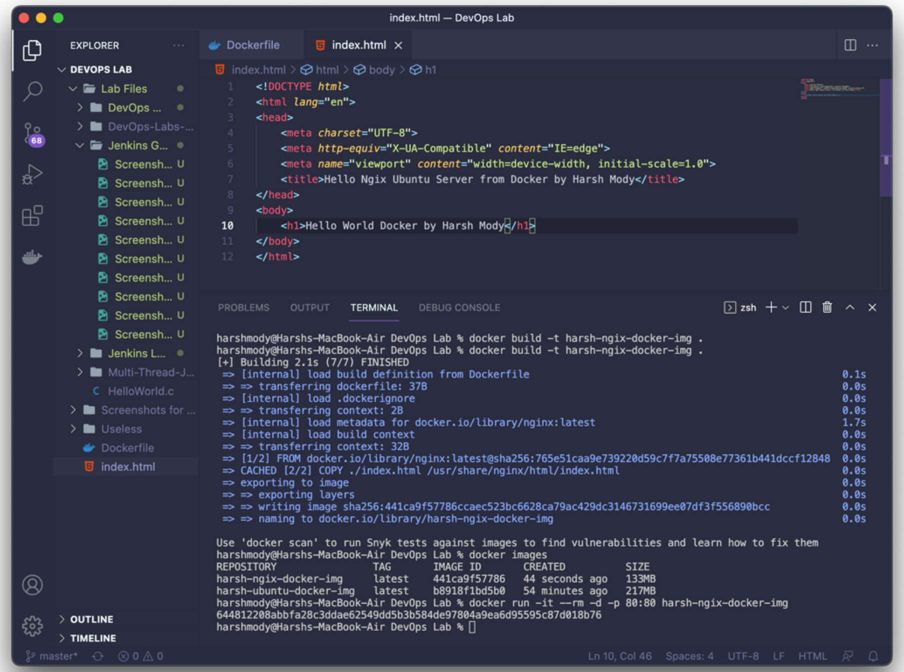
After terminating the image container, the hosted webpage is no longer visible.



Safari Can't Connect to the Server

Safari can't open the page "localhost" because Safari can't connect to the server "localhost".

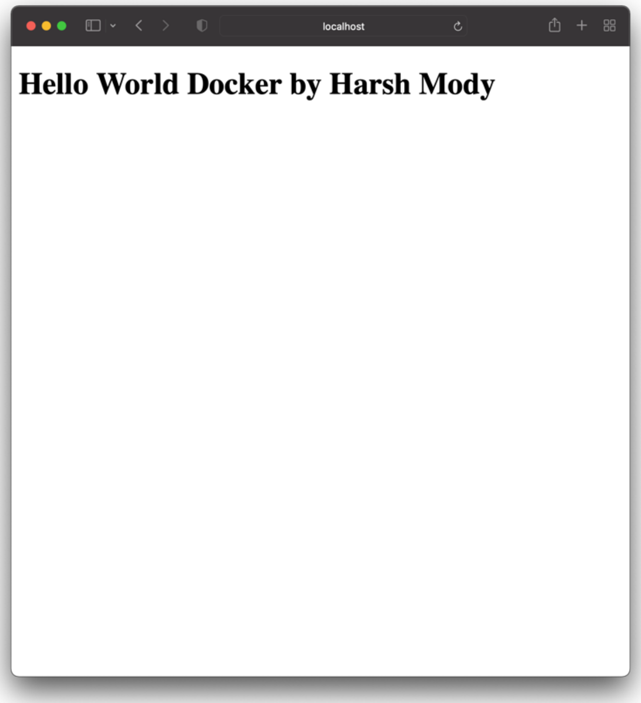
To run a web app using nginx, we create a simple custom index.html and use default nginx image as our base image to host our app.



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Hello Ngix Ubuntu Server from Docker by Harsh Mody</title>
</head>
<body>
    <h1>Hello World Docker by Harsh Mody</h1>
</body>
</html>
```

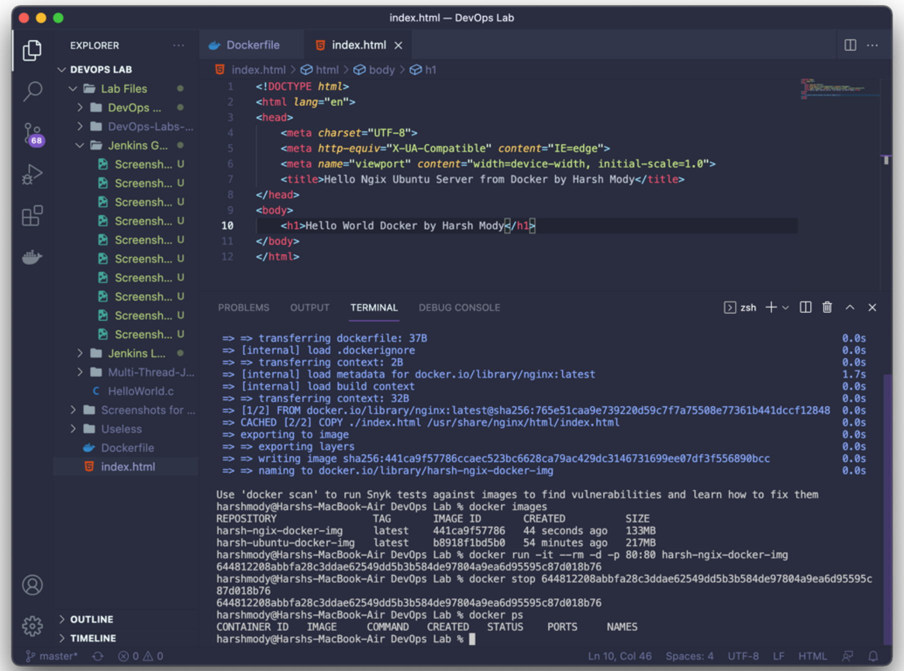
```
harshmody@Harshs-MacBook-Air DevOps Lab % docker build -t harsh-ngix-docker-img .
harshmody@Harshs-MacBook-Air DevOps Lab % docker build -t harsh-ngix-docker-img .
[+] Building 2.1s (7/7) FINISHED
 => [internal] load build definition from Dockerfile                                                   0.1s
 => => transferring dockerfile: 37B                                                                    0.0s
 => [internal] load .dockerignore                                                                      0.0s
 => => transferring context: 2B                                                                        0.0s
 => [internal] load metadata for docker.io/library/nginx:latest                                        1.7s
 => [internal] load build context                                                                      0.0s
 => => transferring context: 32B                                                                       0.0s
 => [1/2] FROM docker.io/library/nginx:latest@sha256:765e51caa9e739220d59c7f7a75508e77361b441dccf12848  0.0s
 => CACHED [2/2] COPY ./index.html /usr/share/nginx/html/index.html                                    0.0s
 => exporting to image                                                                                 0.0s
 => => exporting layers                                                                                0.0s
 => => writing image sha256:441ca9f57786ccaec523bc6628ca79ac429dc3146731699ee07df3f556890bcc           0.0s
 => => naming to docker.io/library/harsh-ngix-docker-img                                               0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
harshmody@Harshs-MacBook-Air DevOps Lab % docker images
REPOSITORY               TAG       IMAGE ID       CREATED          SIZE
harsh-ngix-docker-img    latest    441ca9f57786   44 seconds ago   133MB
harsh-ubuntu-docker-img  latest    b8918f1bd5b0   54 minutes ago   217MB
harshmody@Harshs-MacBook-Air DevOps Lab % docker run -it --rm -d -p 80:80 harsh-ngix-docker-img
644812208abbfa28c3ddae62549dd5b3b584de97804a9ea6d95595c87d018b76
harshmody@Harshs-MacBook-Air DevOps Lab % 
```

We can check if our container was created successfully by visiting http://localhost:80/



Hello World Docker by Harsh Mody

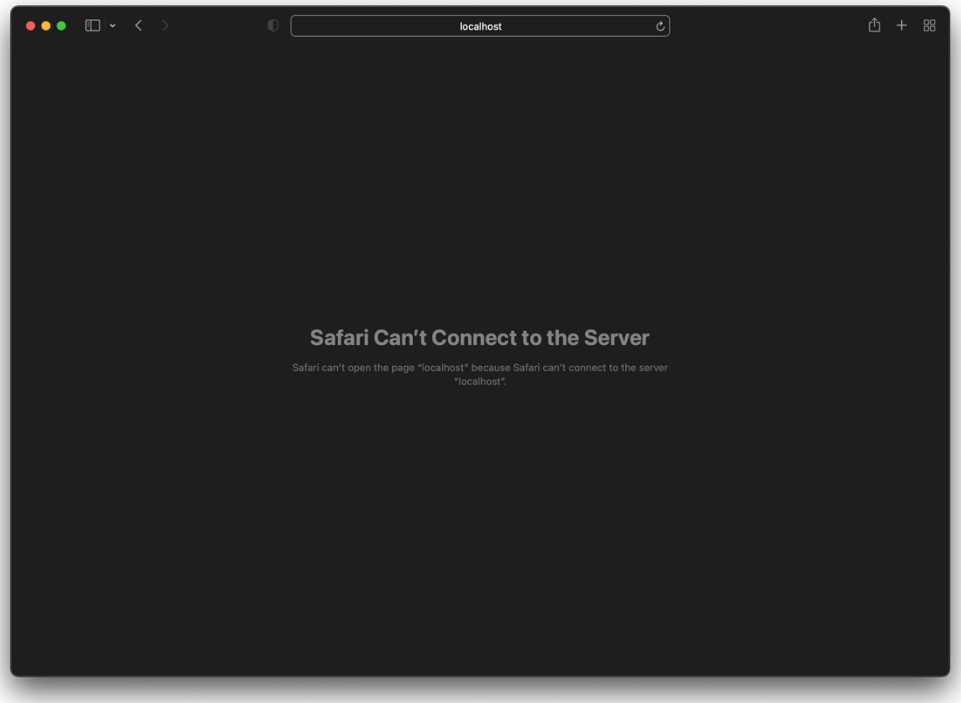
We can then stop the docker container and run docker prune to delete all images to save storage space.



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Hello Ngix Ubuntu Server from Docker by Harsh Mody</title>
</head>
<body>
    <h1>Hello World Docker by Harsh Mody</h1>
</body>
</html>
```

```
=> => transferring dockerfile: 37B                                                          0.0s
=> [internal] load .dockerignore                                                            0.0s
=> => transferring context: 2B                                                              0.0s
=> [internal] load metadata for docker.io/library/nginx:latest                              1.7s
=> [internal] load build context                                                            0.0s
=> => transferring context: 32B                                                             0.0s
=> [1/2] FROM docker.io/library/nginx:latest@sha256:765e51caa9e739220d59c7f7a75508e77361b441dccf12848  0.0s
=> CACHED [2/2] COPY ./index.html /usr/share/nginx/html/index.html                          0.0s
=> exporting to image                                                                       0.0s
=> => exporting layers                                                                      0.0s
=> => writing image sha256:441ca9f57786ccaec523bc6628ca79ac429dc3146731699ee07df3f556890bcc 0.0s
=> => naming to docker.io/library/harsh-ngix-docker-img                                     0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
harshmody@Harshs-MacBook-Air DevOps Lab % docker images
REPOSITORY              TAG       IMAGE ID       CREATED          SIZE
harsh-ngix-docker-img     latest    441ca9f57786   44 seconds ago   133MB
harsh-ubuntu-docker-img   latest    b8918f1bd5b0   54 minutes ago   217MB
harshmody@Harshs-MacBook-Air DevOps Lab % docker run -it --rm -d -p 80:80 harsh-ngix-docker-img
644812208abbfa28c3ddae62549dd5b3b584de97804a9ea6d95595c87d018b76
harshmody@Harshs-MacBook-Air DevOps Lab % docker stop 644812208abbfa28c3ddae62549dd5b3b584de97804a9ea6d95595c
87d018b76
644812208abbfa28c3ddae62549dd5b3b584de97804a9ea6d95595c87d018b76
harshmody@Harshs-MacBook-Air DevOps Lab % docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
harshmody@Harshs-MacBook-Air DevOps Lab % 
```

After terminating the image container, the hosted webpage is no longer visible.



localhost

Safari Can't Connect to the Server

Safari can't open the page "localhost" because Safari can't connect to the server "localhost".

<u>Conclusion:</u> Thus, successfully understood the importance of Containerization tools like Docker and learnt the use of Dockerfile to create and build custom Docker containers.