Lab Exercise 5- Building a Docker Image for an HTML App Using Nginx

Name : Raman Boora

SapID: 500109408

Roll no: R2142221160

1. Setup

You will need:

- Docker installed on your machine.
- A simple HTML file for the app.

2. Step 1: Create the HTML File

Create a directory for your HTML app and place an index.html file in it.

```
mkdir docker-app

cd docker-app
```

Inside the nginx-html-app directory, create the HTML file.

```
touch index.html
```

Edit the index.html file with the following content (or any custom HTML content you want):

```
<!DOCTYPE html>
<html>
<head>
    <title>Welcome to My Nginx HTML App</title>
</head>
<body>
    <h1>Hello, Nginx Docker!</h1>
        <p>This is a simple HTML app served by Nginx in a Docker container.</p>
```

```
</body>
</html>
```

3. Step 2: Create a Dockerfile

In the same directory, create a Dockerfile. This file will define how to build the Docker image using Nginx as the base image.

```
touch Dockerfile
```

Edit the Dockerfile and add the following content:

```
FROM nginx:latest
COPY index.html /usr/share/nginx/html/
EXPOSE 80
```

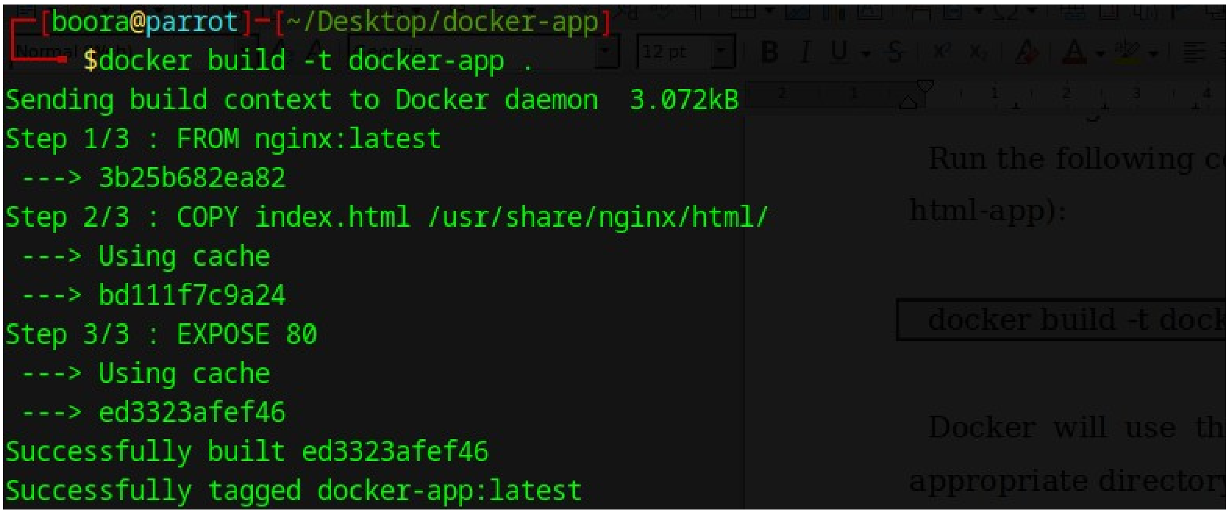
4. Step 3: Build the Docker Image

Now that you have the Dockerfile and index.html, it’s time to build the Docker image.

Run the following command to build the image, giving it a tag (e.g., nginx-html-app):

```
docker build -t docker-app .
```

Docker will use the Nginx base image, copy your index.html into the appropriate directory, and build the image.



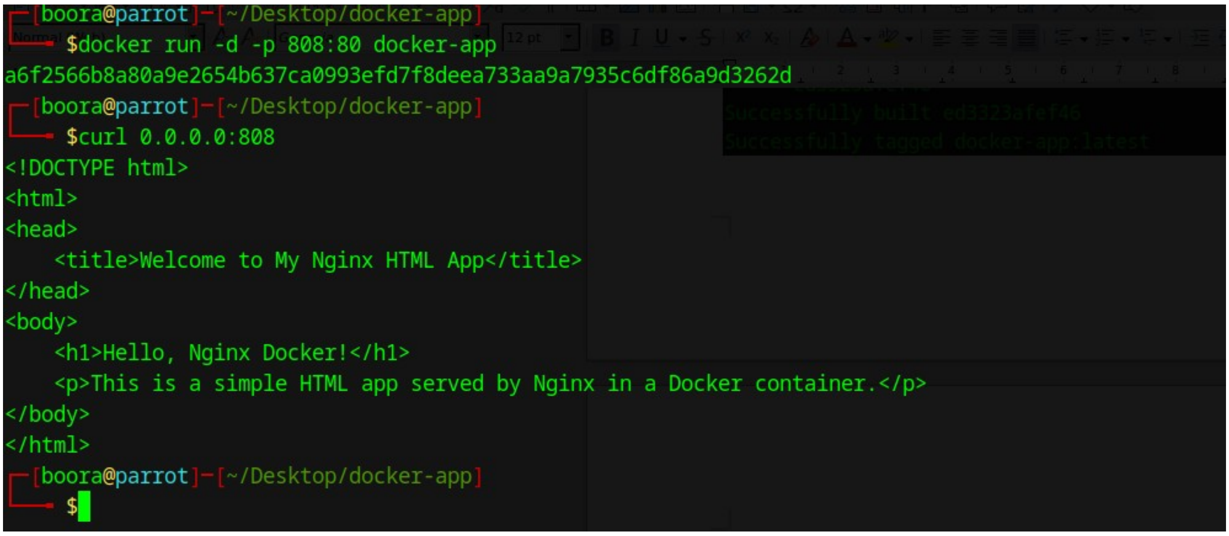
```
┌─[boora@parrot]─[~/Desktop/docker-app]
└──╼ $docker build -t docker-app .
Sending build context to Docker daemon  3.072kB
Step 1/3 : FROM nginx:latest
 ---> 3b25b682ea82
Step 2/3 : COPY index.html /usr/share/nginx/html/
 ---> Using cache
 ---> bd111f7c9a24
Step 3/3 : EXPOSE 80
 ---> Using cache
 ---> ed3323afef46
Successfully built ed3323afef46
Successfully tagged docker-app:latest
```

5. Step 4: Run the Docker Container

After building the image, you can run the container with the following command:

```
docker run -d -p 8080:80 docker-app
```

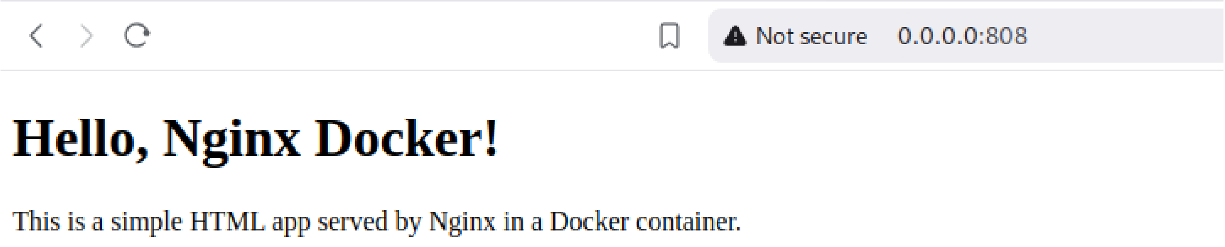
This command runs the container in detached mode (-d) and maps port 8080 on your host machine to port 80 inside the container, where Nginx is serving your HTML app.



```
┌─[boora@parrot]─[~/Desktop/docker-app]
└──╼ $docker run -d -p 808:80 docker-app
a6f2566b8a80a9e2654b637ca0993efd7f8deea733aa9a7935c6df86a9d3262d
┌─[boora@parrot]─[~/Desktop/docker-app]
└──╼ $curl 0.0.0.0:808
<!DOCTYPE html>
<html>
<head>
    <title>Welcome to My Nginx HTML App</title>
</head>
<body>
    <h1>Hello, Nginx Docker!</h1>
    <p>This is a simple HTML app served by Nginx in a Docker container.</p>
</body>
</html>
┌─[boora@parrot]─[~/Desktop/docker-app]
└──╼ $
```

6. Step 5: Verify

Open a browser and go to http://localhost:8080. You should see your HTML page with the message “Hello, Nginx Docker!”.



Hello, Nginx Docker!

This is a simple HTML app served by Nginx in a Docker container.

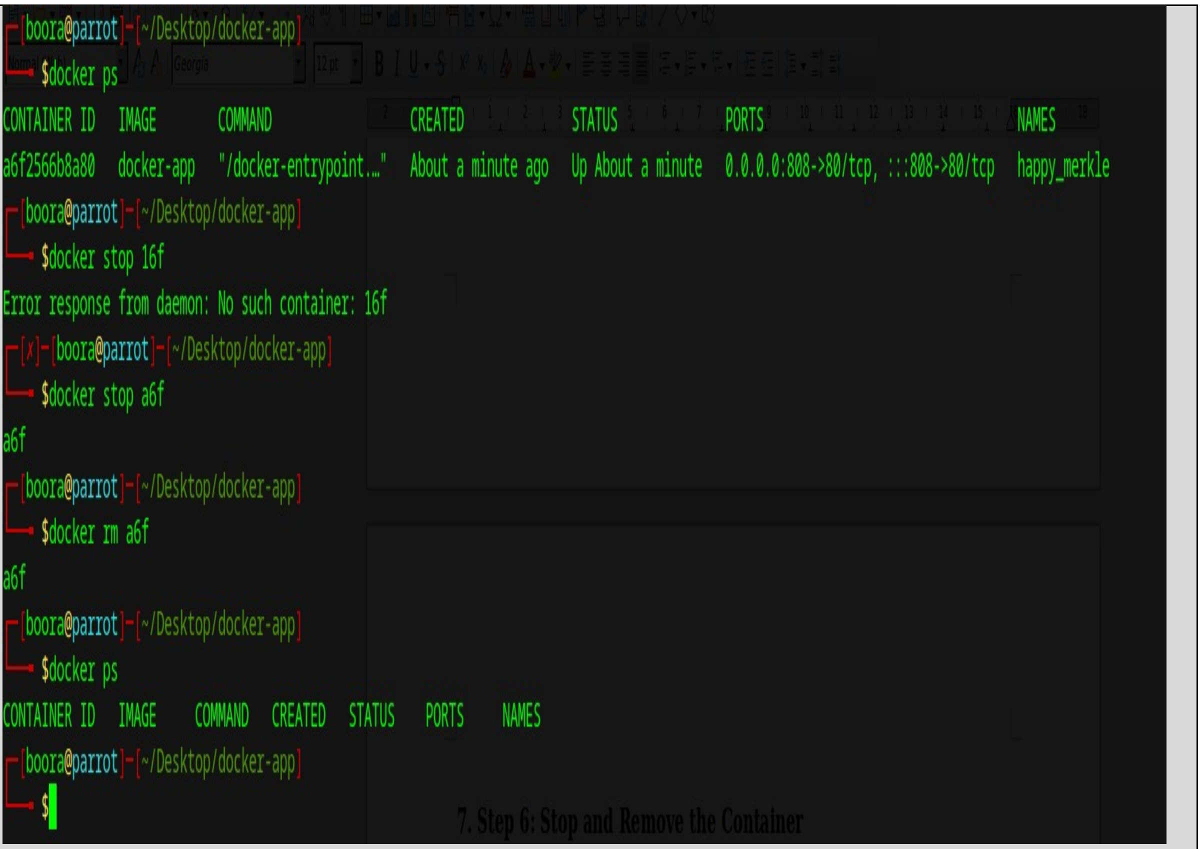
7. Step 6: Stop and Remove the Container

Once you're done, you can stop and remove the container:

```
docker ps  # to see running containers

docker stop <container-id>

docker rm <container-id>
```



```
┌─[boora@parrot]─[~/Desktop/docker-app]
└──╼ $docker ps
CONTAINER ID   IMAGE        COMMAND                  CREATED          STATUS          PORTS                                 NAMES
a6f2566b8a80   docker-app   "/docker-entrypoint.…"   About a minute ago   Up About a minute   0.0.0.0:808->80/tcp, :::808->80/tcp   happy_merkle
┌─[boora@parrot]─[~/Desktop/docker-app]
└──╼ $docker stop 16f
Error response from daemon: No such container: 16f
┌─[✗]─[boora@parrot]─[~/Desktop/docker-app]
└──╼ $docker stop a6f
a6f
┌─[boora@parrot]─[~/Desktop/docker-app]
└──╼ $docker rm a6f
a6f
┌─[boora@parrot]─[~/Desktop/docker-app]
└──╼ $docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
┌─[boora@parrot]─[~/Desktop/docker-app]
└──╼ $
```

7. Step 6: Stop and Remove the Container