

ANSHIKA SRIVASTAVA
ROLL NUMBER – R2142220907
SAP ID – 500107049
DEVSECOPS BATCH B1 HONS.

EXPERIMENT – 5

Building a Docker Image for an HTML App

Using Nginx

1. Setup

- 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 nginx-html-app
```

```
cd nginx-html-app
```

```
anshi@HP MINGW64 /d
$ mkdir nginx-html-app

anshi@HP MINGW64 /d
$ cd nginx-html-app
```

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

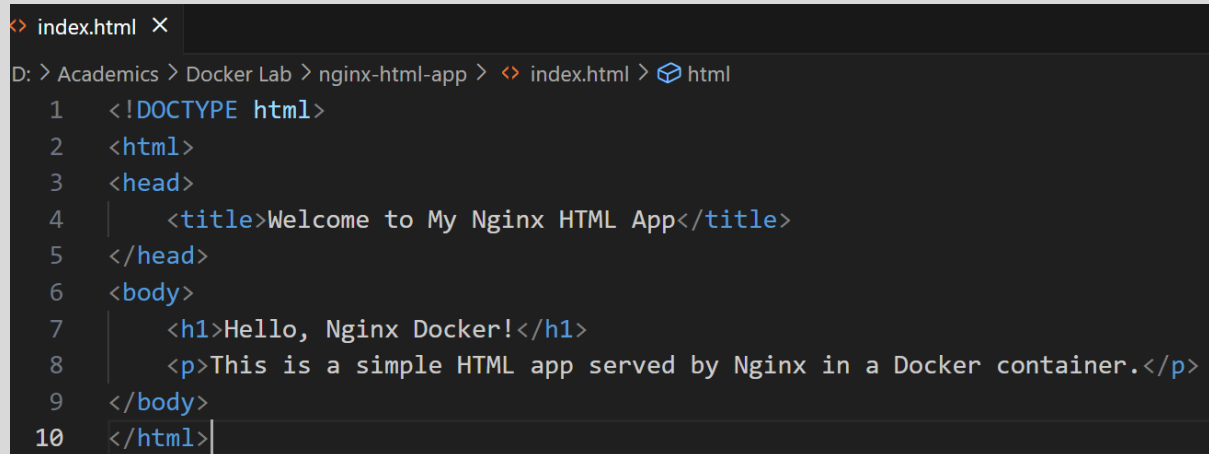
```
touch index.html
```

```
anshi@HP MINGW64 /d/nginx-html-app
$ touch index.html

anshi@HP MINGW64 /d/nginx-html-app
$ |
```

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>
```

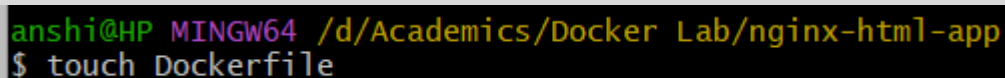
A screenshot of a code editor window. The title bar shows 'index.html' with a close button. The editor content shows the same HTML code as the previous block, with line numbers 1 through 10 on the left. The file path in the top bar is 'D: > Academics > Docker Lab > nginx-html-app > index.html'.

```
index.html ×
D: > Academics > Docker Lab > nginx-html-app > index.html > html
1  <!DOCTYPE html>
2  <html>
3  <head>
4    <title>Welcome to My Nginx HTML App</title>
5  </head>
6  <body>
7    <h1>Hello, Nginx Docker!</h1>
8    <p>This is a simple HTML app served by Nginx in a Docker container.</p>
9  </body>
10 </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.

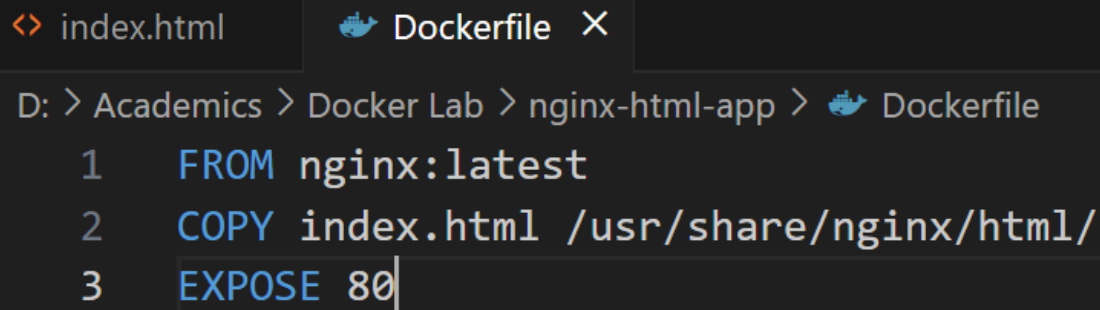
touch Dockerfile

A screenshot of a terminal window. The prompt is 'anshi@HP MINGW64' and the command being entered is 'touch Dockerfile'. The file path in the prompt is '/d/Academics/Docker Lab/nginx-html-app'.

```
anshi@HP MINGW64 /d/Academics/Docker Lab/nginx-html-app
$ touch Dockerfile
```

Edit the Dockerfile and add the following content:

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

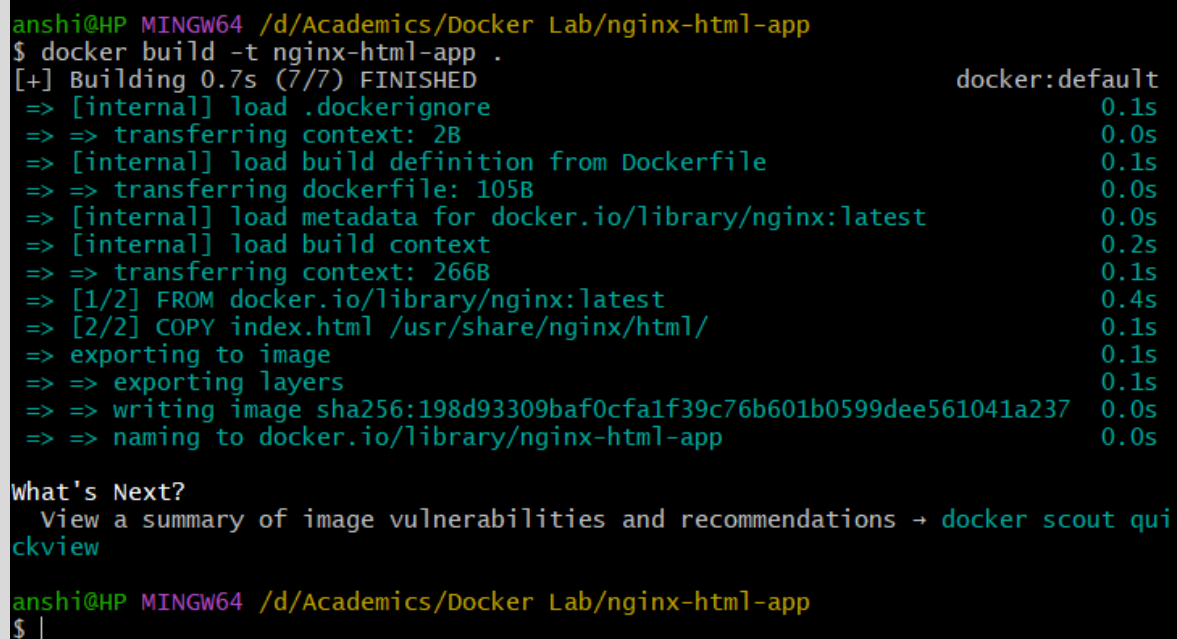
A screenshot of a code editor with two tabs: 'index.html' and 'Dockerfile'. The 'Dockerfile' tab is active, showing the following content:

```
D: > Academics > Docker Lab > nginx-html-app > Dockerfile
1  FROM nginx:latest
2  COPY index.html /usr/share/nginx/html/
3  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 nginx-html-app .
```

A screenshot of a terminal window showing the output of the 'docker build' command. The prompt is 'anshi@HP MINGW64 /d/Academics/Docker Lab/nginx-html-app'. The command executed is '\$ docker build -t nginx-html-app .'. The output shows the build progress with various steps and their durations, ending with 'naming to docker.io/library/nginx-html-app'. Below the output, there is a 'What's Next?' section suggesting to use 'docker scout quickview'.

```
anshi@HP MINGW64 /d/Academics/Docker Lab/nginx-html-app
$ docker build -t nginx-html-app .
[+] Building 0.7s (7/7) FINISHED                                docker:default
=> [internal] load .dockerignore                                0.1s
=> => transferring context: 2B                                    0.0s
=> [internal] load build definition from Dockerfile             0.1s
=> => transferring dockerfile: 105B                               0.0s
=> [internal] load metadata for docker.io/library/nginx:latest  0.0s
=> [internal] load build context                                0.2s
=> => transferring context: 266B                                   0.1s
=> [1/2] FROM docker.io/library/nginx:latest                   0.4s
=> [2/2] COPY index.html /usr/share/nginx/html/                 0.1s
=> exporting to image                                           0.1s
=> => exporting layers                                           0.1s
=> => writing image sha256:198d93309baf0cfa1f39c76b601b0599dee561041a237 0.0s
=> => naming to docker.io/library/nginx-html-app                0.0s

What's Next?
View a summary of image vulnerabilities and recommendations → docker scout quickview

anshi@HP MINGW64 /d/Academics/Docker Lab/nginx-html-app
$ |
```

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

5. Step 4: Run the Docker Container

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

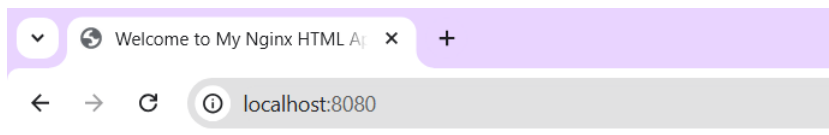
```
docker run -d -p 8080:80 nginx-html-app
```

```
anshi@HP MINGW64 /d/Academics/Docker Lab
$ docker run -d -p 8080:80 nginx-html-app
a1a776008584ffa87936a86978703d69bdb89b41cd707abca42beb645a9e49a3
anshi@HP MINGW64 /d/Academics/Docker Lab
$
```

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.

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
```

```
anshi@HP MINGW64 /d/Academics/Docker Lab
$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
a1a776008584   nginx-html-app "/docker-entrypoint..." About a minute ago Up About a minute 0.0.0.0:8080->80/tcp    stupefied_proskuriakova
anshi@HP MINGW64 /d/Academics/Docker Lab
$
```

```
docker stop <container-id>
```

```
anshi@HP MINGW64 /d/Academics/Docker Lab
$ docker stop ala
ala
```

docker rm <container-id>

```
anshi@HP MINGW64 /d/Academics/Docker Lab
$ docker rm ala
ala
```

```
anshi@HP MINGW64 /d/Academics/Docker Lab
$ |
```

```
anshi@HP MINGW64 /d/Academics/Docker Lab
$ docker ps -a
CONTAINER ID   IMAGE          COMMAND         CREATED        STATUS        PORTS          NAMES
anshi@HP MINGW64 /d/Academics/Docker Lab
$ |
```