

Lab Exercise 21

Building a Docker Image for an HTML App Using Nginx

Name- Misha

Batch-2 (DevOps)

SAP ID- 500119679

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 nginx-html-app
```

```
cd nginx-html-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>

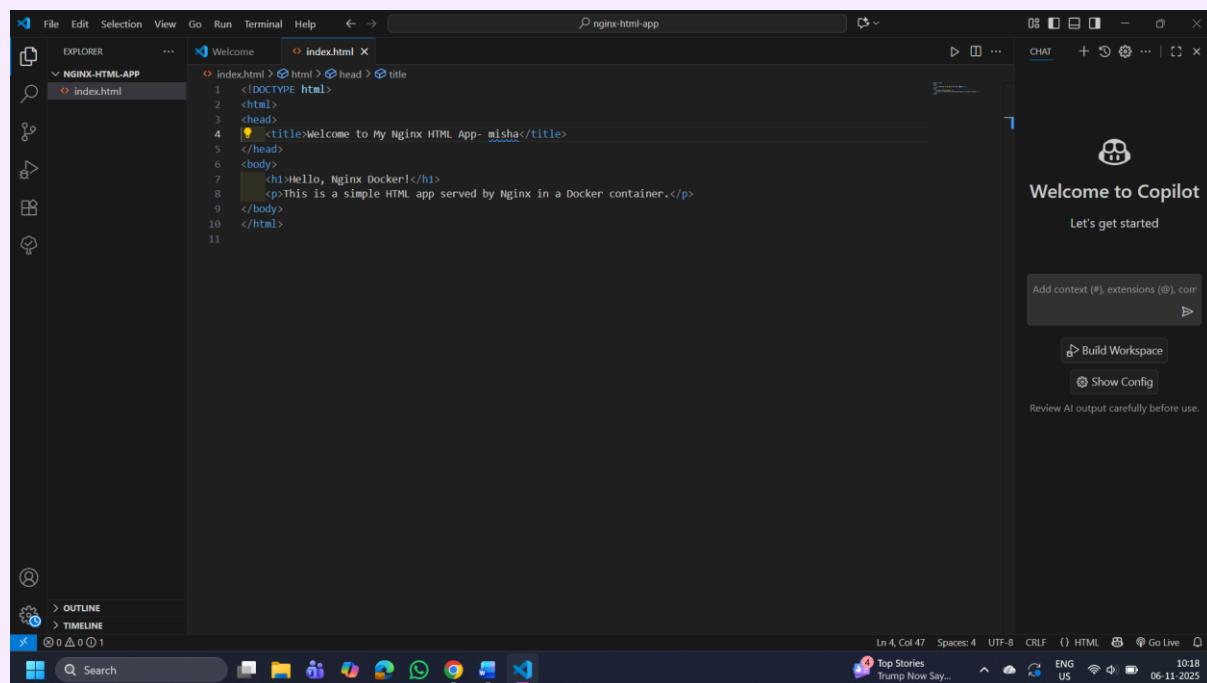
    <h1>Hello, Nginx Docker!</h1>

    <p>This is a simple HTML app served by Nginx in a Docker container.</p>

</body>

</html>

```



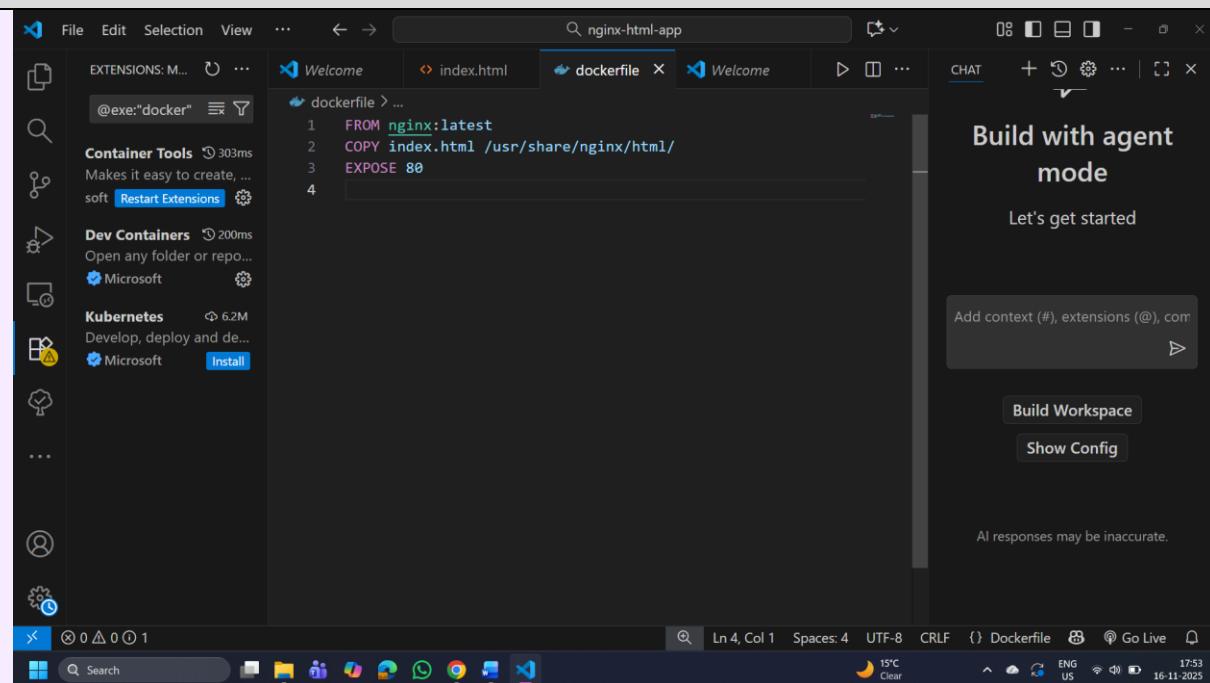
3. Step 2: Create a Dockerfile

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 nginx-html-app .
```

```

PS C:\Users\Misha> cd C:\Terraform\nginx-html-app
PS C:\Terraform\nginx-html-app> docker build -t nginx-html-app .
ERROR: error during connect: Head "http://%2F%2Fpipe%2FdockerDesktopLinuxEngine/_ping": open //./pipe/dockerDesktopLinuxEngine/_ping: no such file or directory
linuxEngine: The system cannot find the file specified.
PS C:\Terraform\nginx-html-app> docker build -t nginx-html-app .
[+] Building 13.1s (5/7)
   => [internal] load build definition from dockerfile
   => transferring dockerfile: 107B
   => [internal] load metadata for docker.io/library/nginx:latest
   => [auth] library/nginx:pull token for registry-1.docker.io
   => [internal] load .dockerignore
   => [internal] transfer context: 2B
   => [internal] load build context
   => [internal] transfer context: 275B
   => [1/2] FROM docker.io/library/nginx:latest@sha256:1beed3ca46acebe9d3fb62e9067f03d05d5bfa97a00f30938a0a35805632 7.2s
   => resolve docker.io/library/nginx:latest@sha256:1beed3ca46acebe9d3fb62e9067f03d05d5bfa97a00f30938a0a35805632 0.0s
   => sha256:e2f8e296d9df1dd5e2ddc81e5e758f9762fd9b32e982ac6873e36692c3e3c983 0.2s
   => sha256:52bc359bcd74bb3d1b94cf3c6d94bcf9bd2d3e150483fb978124ceddb9ca57 0.0s
   => sha256:9def903993e4ef9a3faa02bb893b0382768a4d466d51247bf-f1ea80b119377a1 1.40kB / 1.40kB 0.0s
   => sha256:d921c57c6a81addac6ca451906699ca6ee8c01fd708805a928181c5370b0a30c 1.21kB / 1.21kB 1.6s
   => sha256:320b6949be89766f76a8746f1971021a8e8c84928af00454c0f9c6e38ebf54c 956B / 956B 1.6s
   => sha256:266626526d42cf7fe5f56b933db3bf459c0596b7e2c3a556ba5ec4981daf3e9d 628B / 628B 1.0s
   => sha256:d7ecded7702a5dbf6d0f79a71edc34b534d08f3051980e2c948fba72db3197fc 11.53MB / 11.53MB 5.2s
   => sha256:d7ecded7702a5dbf6d0f79a71edc34b534d08f3051980e2c948fba72db3197fc 18.87MB / 18.87MB 5.2s

```

is may be inaccurate.

Ln 4, Col 1 Spaces: 4 UTF-8 CRLF () Dockerfile 🌐 Go Live 15°C Clear ENG US 18:13 16-11-2025

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 8006:80 nginx-html-app
```

```

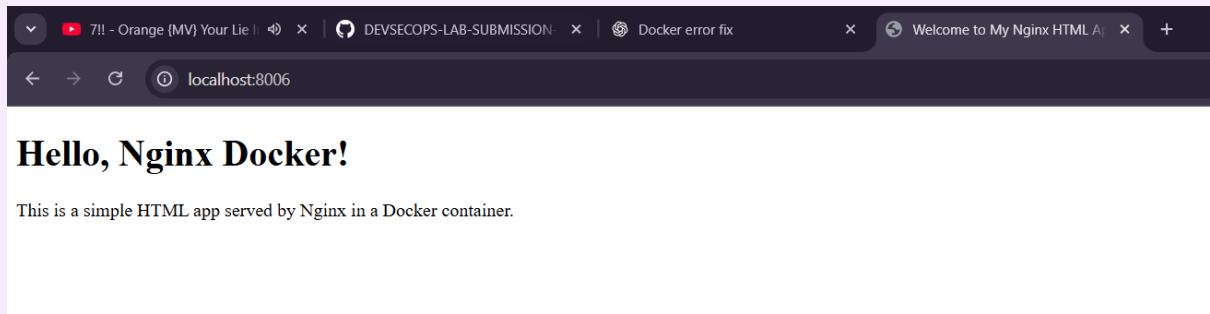
PS C:\Terraform\nginx-html-app> docker run -d -p 8006:80 nginx-html-app
b520c9a35feedb7793c205fbf4cd8b5a2872fae535ac2cd6a38cc0462e8f1f8f
PS C:\Terraform\nginx-html-app>

```

This command runs the container in detached mode (-d) and maps port 8006 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:8006>. You should see your HTML page with the message “Hello, Nginx Docker!”.



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
```

```
PS C:\Terraform\nginx-html-app> docker run -d -p 8006:80 nginx html app
b520c9a35feedb7793c205fbf4cd8b5a2872fae535ac2cd6a38cc0462e8f1f8f
PS C:\Terraform\nginx-html-app> docker ps # to see running containers
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS
 NAMES
b520c9a35fee      nginx-html-app   "/docker-entrypoint..."   About a minute ago   Up About a minute   0.0.0.0:8006->80/tcp,
[::]:8006->80/tcp  nice_black
PS C:\Terraform\nginx-html-app>
```

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

```
docker rm <container-id>
```

```
PS C:\Terraform\nginx-html-app> docker stop b520c9a35feedb7793c205fbf4cd8b5a2872fae535ac2cd6a38cc0462e8f1f8f
b520c9a35feedb7793c205fbf4cd8b5a2872fae535ac2cd6a38cc0462e8f1f8f
PS C:\Terraform\nginx-html-app> docker rm b520c9a35feedb7793c205fbf4cd8b5a2872fae535ac2cd6a38cc0462e8f1f8f
b520c9a35feedb7793c205fbf4cd8b5a2872fae535ac2cd6a38cc0462e8f1f8f
PS C:\Terraform\nginx-html-app>
```