Report on Creating a Custom Docker Image for a Static Website

By Anjali Dubey RA2211056010119
Saaz Bhargava RA2211056010114

1. Project Aim

The aim of this project is to create a **custom Docker image** for a simple **static website** and deploy it using **Docker**. By containerizing the static website, we ensure portability, consistency, and ease of deployment across different environments.

2. Introduction

A **Docker container** allows applications to run in an isolated environment, ensuring that dependencies and configurations remain consistent across different systems. In this project, we use **Nginx**, a high-performance web server, as the base image to serve the static website.

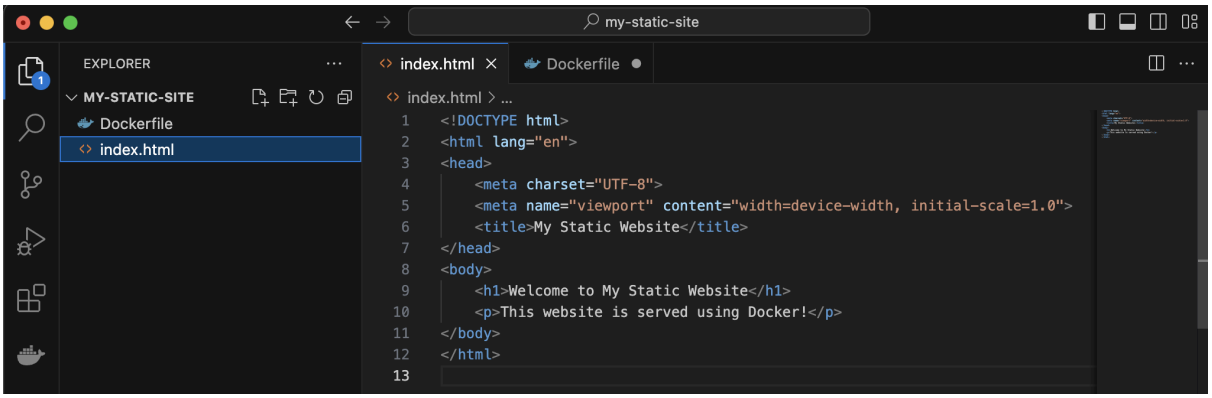
3. Detailed Procedures

Step 1: Setting Up the Project

1.Open **VS Code** and create a new folder:
mkdir docker-static-website

cd docker-static-website

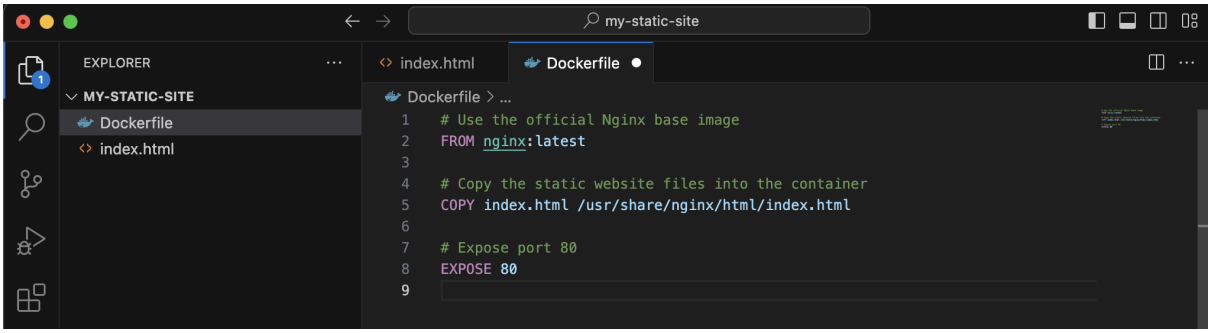
2.Create an **HTML file** to serve as the website's homepage:
touch index.html



Step 2: Creating the Dockerfile

1.Inside the project folder, create a Dockerfile:
touch Dockerfile

2.Open Dockerfile and add the following:



Step 3: Building the Docker Image

Run the following command to build the Docker image:

```
docker build -t my-static-website .
```

This command:

- Builds an image using the Dockerfile.
- Tags the image as my-static-website.

Step 4: Running the Docker Container

Run the following command to start a container using the image:

```
docker run -d -p 8080:80 --name static-container my-static-website
```

This command:

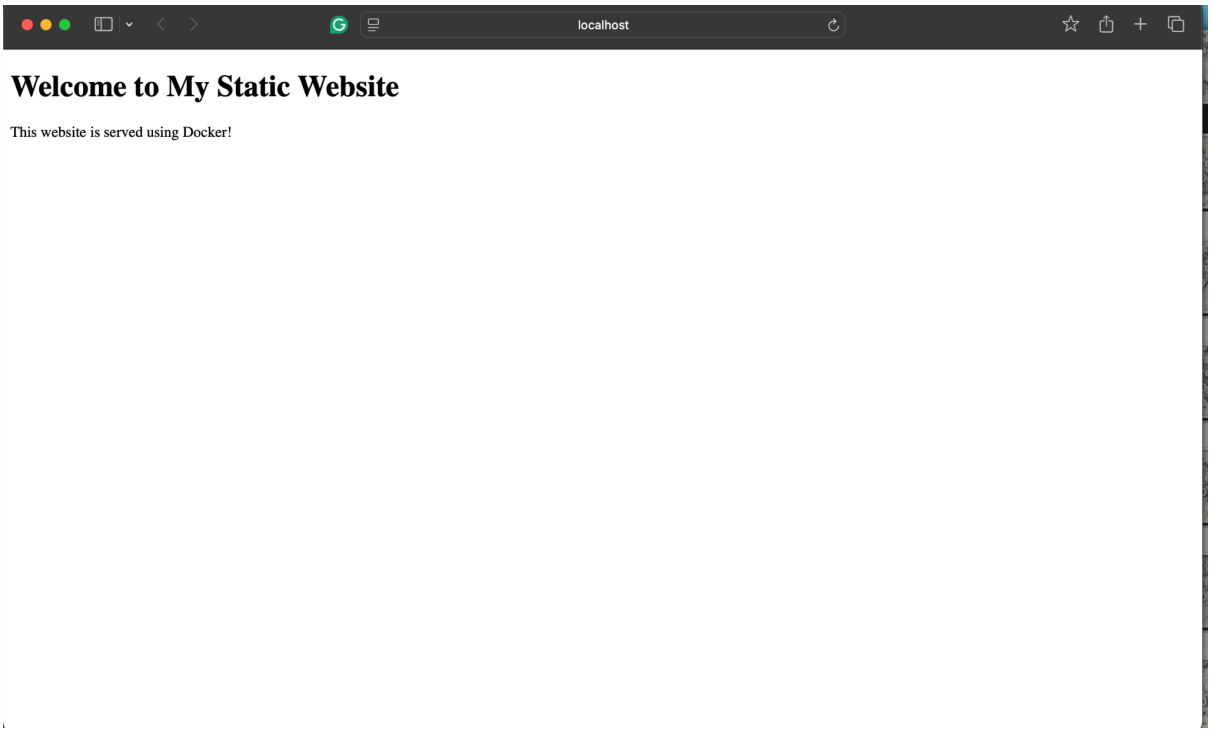
- Runs the container in the background (-d).
- Maps **port 8080 on the host** to **port 80 in the container**.
- Names the container static-container.

Step 5: Accessing the Website

Open a web browser and go to:

http://localhost:8080

The static website should now be visible.



4. Results and Observations

1. The **Docker image** was successfully built and contained the static website.
2. The **container ran without errors**, serving the website using **Nginx**.
3. The static website was accessible through **http://localhost:8080**.
4. The process was efficient, demonstrating the ease of **containerizing applications**.

5. Conclusion

This project successfully demonstrated how to **create, build, and run** a custom Docker image for a static website. Using **Nginx as the web server**, the website was served efficiently in a **containerized environment**. This approach ensures easy deployment and scalability of static web applications.