Report Challenge 1

Setting Up a Simple Web Server

Introduction:

In this tutorial, we'll walk through the process of using Docker to serve static web pages with an NGINX web server. By the end of this challenge, you'll learn the basics of Docker, how to build a Docker image, and run a container to serve your static website.

Prerequisites:

Docker installed on your machine (Windows/MacOS/Linux)

Basic understanding of terminal or command prompt commands

A text editor (e.g., VSCode, Sublime Text)

Step 1: Prepare the Project Directory

Create a new folder where you want to store your project. We'll refer to this as the project directory.

Inside the project directory, create a subfolder named public. This is where you'll store your web assets.

Step 2: Add Web Content

Within the public folder, create an index.html file. This file will be your homepage.

Edit index.html to include your name and student ID number. Style the page as desired using HTML/CSS.

You can add more assets (images, CSS files) to the public folder as needed.

Step 3: Creating the Dockerfile

In the root of your project directory, create a file named Dockerfile. Note: The file should not have any extension.

Open the Dockerfile in your editor and add the following content:

----------Dockerfile----------

```
# Use the official image for NGINX
FROM nginx:alpine

# Copy static assets to NGINX public folder
COPY ./public /usr/share/nginx/html

# Expose port 80
EXPOSE 80

# Start NGINX
CMD ["nginx", "-g", "daemon off;"]
```

----------Dockerfile----------

This tells Docker to use the lightweight nginx:alpine image, copy your public folder to the nginx serving directory, and expose port 80.

Step 4: Build the Docker Image

Open a terminal or command prompt.

Navigate to your project directory.

Run the following command to build your Docker image:

----------cmd----------

```
docker build -t webserver .
```

----------cmd----------

Replace webserver with a name of your choice for the image. Here I have used "docker build -t dockercl1 .".

Step 5: Run Your Docker Container

After the build completes, start your container with:

----------cmd----------

docker run -d -p 8080:80 webserver

----------cmd----------

Again replace webserver with a name of your choice for the image. Here I have used "docker run -d -p 8080:80 dockercl1".

This maps port 8080 on your host to port 80 in the container.

Step 6: Verify Your Web Server

Open a web browser and navigate to http://localhost:8080.

You should see your homepage displaying your name and student ID number.

Step 7: Publish Your Work

Initialize a Git repository in your project directory and commit your changes.

Create a private GitHub repository and push your project.

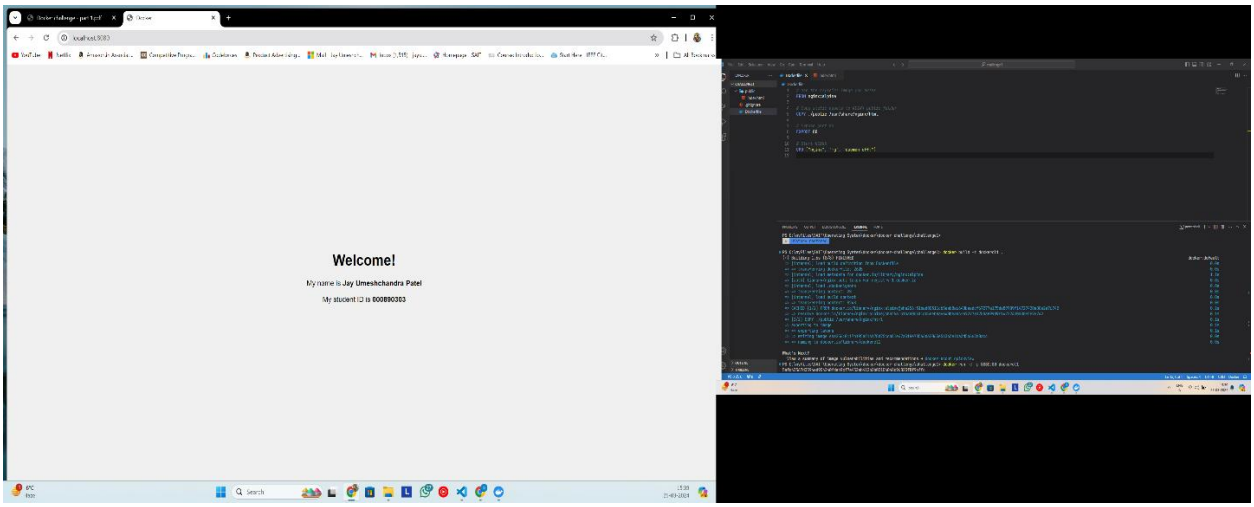
Conclusion:

Congratulations! You've successfully completed the challenge of setting up a simple web server using Docker and NGINX. This tutorial has introduced you to the basics of Docker, including how to build images and run containers.

Reference Links:

https://docs.docker.com/get-started/overview/

https://docker-curriculum.com/


Welcome!
My name is Jay Umeshchandra Patel
My student ID is 000890303