**Use Case Document: Dockerized Nginx Application with Docker Compose**

**Objective:**

You are required to **create a simple Dockerized application** using **Nginx** and run it via **Docker Compose**. The goal is to simulate a production-ready static content server where Nginx serves a custom HTML page.

**Application Requirements:**

1. **Web Server**: Nginx
2. **Static Content**: A custom index.html page containing a welcome message (e.g., "Welcome to the Docker Nginx App!")
3. **Dockerization**:
   * Dockerfile to build a custom Nginx image.
   * Docker Compose configuration to run the container with a defined network.
4. **Container Naming**: Container must be named nginx.
5. **Port Mapping**: Host port 8080 should be mapped to container port 80.

**Expected Directory Structure:**

docker-nginx-app/

│

├── docker-compose.yml

├── Dockerfile

└── html/

└── index.html

**1️⃣ Step 1: Create index.html**

Create a directory named html/ and inside it, create an index.html file with the following content:

<!DOCTYPE html>

<html>

<head>

<title>Welcome</title>

</head>

<body>

<h1>Welcome to the Docker Nginx App!</h1>

</body>

</html>

**Step 2: Create Dockerfile**

In the project root directory, create a Dockerfile to build a custom image:

**Dockerfile**

User the nginx:alpine

Copy to tests folder

This Dockerfile uses a lightweight Nginx base image and copies your static content into the container.

**Step 3: Create docker-compose.yml**

In the same root directory, create the Docker Compose file:

Map it to port 8081

**How to Build and Run the Application:**

1. **Open terminal in project root (/docker-nginx-app/)**
2. **Build and run using Docker Compose:**

**docker-compose up --build -d**

1. **Verify that the container is running:**

**docker ps**

You should see a container named nginx running and exposing port 8081.

1. **Access the application in your browser:**

Open: <http://localhost:8080>

You should see the welcome message from index.html.

**What Will Be Evaluated:**

* Correct directory and file structure
* Proper Dockerfile and Compose configuration
* Accessibility of the application on http://localhost:8081
* Container name must be nginx
* Nginx must serve the custom HTML content

**Steps to run the projects**

1. Open CMD type the command pip install docker .
2. You can use the command docker compose command docker-compose up --build -d to deploy the application
3. After the application is deployed your application you will need to run the testcases
4. In Cmd run the python file run\_tests.py
5. Wait for the python file to run the testcase
6. Next step is push the code to github

**Steps to push the code**

**Push the code the repository**

**You can run test cases as many numbers of times and at any stage of Development, to check how many test cases are passed/failed and accordingly refactor your code.**

1. **Make sure before final submission you commit all changes to git**. For that open the project folder available on desktop

A computer screen with icons on it

AI-generated content may be incorrect.

* 1. Right click in folder and open Git Bash

A screenshot of a computer

AI-generated content may be incorrect.

* 1. In Git bash terminal, run following commands
  2. git status

A screenshot of a computer

AI-generated content may be incorrect.

* 1. git add .

A screenshot of a computer

AI-generated content may be incorrect.

e . git commit -m “First commit”

(You can provide any message every time you commit)

A black screen with icons on it

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.F .git push