Task: Multi-Container Communication Using Docker Networks

Students will set up and run two Docker containers: one for a Flask web application and another for a Redis database. These containers will communicate with each other using Docker networks. The goal is to understand inter-container networking without using Docker Compose.

Objectives:

- Learn how to create and manage Docker networks.
- Build and run a containerized Flask web application.
- Deploy a Redis container and connect it to Flask.
- Understand environment variables and networking in Docker.

Task Steps:

Step 1: Clone the GitHub Repository

Students will clone the provided GitHub repository to their local machine and set up their working environment.

Step 2: Create a Docker Network

Students will create a custom Docker network to enable communication between containers.

Step 3: Run the Redis Container

Students will pull and run a Redis container, ensuring it runs in the created network.

Step 4: Develop a Flask Application

Students will build a simple Flask application that interacts with Redis. The application should:

- Act as a counter that tracks visits to the web page.

- Store and retrieve values from Redis.
- Expose an API endpoint that interacts with Redis data.

Step 5: Containerize the Flask Application

Students will create a Dockerfile to containerize their Flask application.

Step 6: Run the Flask Application in a Container

Students will run the Flask container, ensuring it connects to the Redis container using the Docker network.

Step 7: Push the Flask App to Docker Hub

Students will tag and push their Flask Docker image to their Docker Hub repository.

Step 8: Push the Code to GitHub

Students will commit and push their code to the main branch of the GitHub repository.

Requirements:

- Docker installed on the system.
- A GitHub account for repository management.
- A Docker Hub account for pushing the Flask app image.
- Basic knowledge of Python and Flask.
- Internet access to pull Docker images.

Deliverables:

- A GitHub repository with the completed implementation pushed to the main branch.
- A working Flask container that connects to the Redis container over a Docker network.

- The Flask app Docker image pushed to Docker Hub.
- Clear commit messages and a structured repository.