Docker Networking and Volume Management

Objective:

This assignment is designed to develop your skills in Docker container orchestration, focusing on container networking, data sharing, and persistent storage. You will create two Docker containers where one container will generate data and the other will access this data over a Docker network and store it using Docker volumes. The emphasis is on understanding Docker's network communications, volume management, and overall container orchestration with Docker Compose, not on the complexity of the applications themselves.

PS: Keep in mind that in this assignment, we do not care about the actual application. You can generate logs or random texts in one container, and read it from the other (as simple as that). What I care about here is you understanding how to create the images, manage them through docker compose, connect them through docker networks, and store data using volumes.

Requirements:

- **Docker**: Ensure Docker is installed on your machine.
- **Docker Compose**: Docker Compose must be available for orchestrating the containers.

Assignment Details:

1. **Develop Two Simple Applications (You choose the functionality of each container):**
    - **Data Producer Application (Container 1)**: This application will continuously generate simple data, such as random logs or numbers. It should provide this data over a Docker network, possibly through a simple REST API or by writing to a socket.
    - **Data Consumer Application (Container 2)**: This application will listen for data over the network from the Data Producer and store it locally using a Docker volume.

2. **Dockerfiles:**
    - Write a Dockerfile for each application that includes all necessary dependencies and setup instructions (Add comments to explain what each layer does).

3. **Docker Compose Configuration:**
    - Create a `docker-compose.yml` that will:
        - Define both services (Data Producer and Data Consumer).
        - Set up a custom Docker network for the two containers to communicate.
        - Configure volumes for the Data Consumer to ensure data persistence.

4. **Networking and Data Transfer:**
    - Configure the Docker network to facilitate communication between the two containers.
    - The Data Producer should send data to the Data Consumer using this network.

5. **Data Storage:**
    - Use Docker volumes in the Data Consumer to store received data persistently. Ensure the data remains available even if the container restarts.

6. **GitHub Repository Setup:**
    - Initialize a Git repository in your project directory.
    - Include all Dockerfiles, the `docker-compose.yml` file, and any scripts or application code.
    - Provide a README.md with detailed instructions on how to run the project, including building the images and starting the containers with Docker Compose.

Submission Guidelines:

- Submit the URL of your GitHub repository containing the Docker files and source code.
- Your repository must include:
    - Dockerfiles for both the Data Producer and Data Consumer.
    - `docker-compose.yml` file.
    - Source code or scripts for the applications.
    - A README.md explaining the purpose of each component in the repository, how to build the images, and how to run the containers.

Evaluation Criteria:

- Functionality and correctness of the Dockerfiles and docker-compose configuration.
- Effective use of Docker networking to enable communication between containers.
- Proper implementation of Docker volumes for persistent data storage.
- Quality and clarity of the GitHub repository documentation, ensuring it is straightforward for others to understand and deploy the project.