Project Description: Image Embedding and Semantic Similarity Web API and Client Application

Objective

Develop a two-part project consisting of a Web API and a client application that leverages a machine learning (ML) model to generate image embeddings for semantic similarity analysis. The project aims to demonstrate proficiency in building ML-powered web services and client applications using Python, FastAPI, PyTorch, and Docker.

Part A: Inference Web API

Task Description:

- **Web API Development:** Create a Web API using Python and FastAPI that accepts one or more images as input and returns a single embedding per image. You can choose any request format (e.g. JSON, binary, gRPC).
- **ML Inference:** Utilize PyTorch for ML inference to create embeddings for the input images. You may choose any pre-trained ML embedding model suitable for the task.
- **Containerization:** Ensure the Web API is containerized using Docker for ease of deployment and scalability.
- **Endpoints:** The API should have at least one endpoint that accepts image uploads and returns their corresponding embeddings in a structured format (e.g. JSON, binary, gRPC).

Technical Requirements:

- Python 3.x
- FastAPI
- PyTorch
- Docker
- Any pre-trained ML embedding model of your choice

Part B: Client Script

Task Description:

- **Application Development:** Write a simple command line client script in Python that can load multiple images, specifying a single reference image. The image paths can be hardcoded.
- **Embedding Generation:** Utilize the Web API developed in Part A to create embeddings for each of the images, including the reference image.
- **Similarity Analysis:** Calculate and output the semantic similarity of each image in comparison to the reference image.

Technical Requirements:

- Python 3.x
- Requests (or any other HTTP library for Python)

Submission Guidelines

- **GitHub Repository:** Submit your work through a GitHub repository. Your submission should include both parts of the project (the Web API and the client application).

- **README File:** Include a detailed README file that provides instructions on how to set up and run your project. The README should cover system requirements, installation steps, and usage examples.

Evaluation Criteria

Your project will be evaluated based on the following criteria:

- **Functionality:** The Web API and client scripts work as described, handling image inputs and producing embeddings and similarity analysis correctly.
- **Code Quality:** Clean, readable, and well-documented code following Python best practices.
- **Use of Technologies:** Effective use of Python, FastAPI, PyTorch, and Docker as specified in the project requirements.
- **Documentation:** Clear and comprehensive documentation, including setup, running instructions, and examples of use in the README file.

Discussion in Interview

Objective: After reviewing your submission, we'll have a detailed discussion about your approach and solutions. We are particularly interested in understanding your thought process and decision-making. Key Focus: We will explore what additional work would be needed to make your project production-ready. This includes discussing scalability, security, and maintainability aspects.