Full Stack Developer Coding Test

Note to Candidates: We understand that each candidate may have varying levels of expertise in different areas of the test. Please feel free to focus on the sections where you feel most confident in showcasing your skills. You are welcome to solve one or more parts of the problem based on your strengths and expertise. The solutions provided are examples for reference, and you are encouraged to use your preferred methods and tools to achieve the objectives.

Sections:

Part 1: Image Processing and Machine Learning

Part 2: Backend and API Development

Part 3: Data Handling and Augmented Reality Integration

You can choose to solve any or all parts of the test. Completing multiple sections may provide more comprehensive insights into your capabilities, but we value the depth and quality of solutions over quantity.

*Submission Timeline: You have **one week** from receiving this test to complete and submit your solutions. Please ensure that your code is well-documented and organized, and include any necessary instructions in a README.md file.*

Part 1: Image Processing and Machine Learning with Python

Task: Develop a Python script using OpenCV and a pre-trained machine learning model to perform object detection on images of shelves.

Requirements:

- Use OpenCV for image pre-processing (e.g., resizing, normalization).
- Leverage a pre-trained model like YOLO or Faster R-CNN for object detection.
- Output bounding box coordinates for detected products are provided in the sample images.

Evaluation Criteria:

- Correct application of OpenCV for image processing.
- Implementation and integration of a pre-trained model for object detection.
- Code quality, readability, and documentation.

Part 2: Backend and API Development with Python

Task: Create a Flask-based RESTful API to manage a collection of product information, supporting operations for adding, retrieving, updating, and deleting records.

Requirements:

- Store data in an in-memory structure for testing purposes.
- Implement endpoints with appropriate request/response formats and status codes.
- Ensure the API is capable of handling concurrent requests gracefully.

Evaluation Criteria:

- API design, structure, and documentation.
- Implementation of CRUD operations.
- Error handling and response formatting.

Part 3: Data Handling and Augmented Reality Integration

Task: Write a Python script to simulate the integration of detected objects with augmented reality by overlaying text on images.

Requirements:

- Use OpenCV to draw bounding boxes and overlay text on images based on object detection results from Part 1.
- Simulate a simple AR effect by adding basic animations or effects using OpenCV (e.g., changing colors, adding simple graphics).

Evaluation Criteria:

- Effective manipulation of images using OpenCV.
- Creativity in simulating AR overlays.
- Code efficiency and modularity.

Code Submission:

- Organize your code in a GitHub repository.
- Include a README.md file with detailed instructions on setting up and executing each part.
- Ensure that your code is clean, well-structured, and documented appropriately.
- Include any necessary sample data or images required for testing the solutions.

Demonstration Video:

- Record a short video demonstrating the functionality of your solution.
- For each section you solved, show the code in action and explain your approach.
- Upload the video to a platform like YouTube (unlisted video) or Google Drive and provide the link in your submission.

Submission Details:

- Send the GitHub repository link and the video link to hr@fyndme.net.
- Ensure that your submission is completed within one weeks of receiving this test.