Project Name: Online Face Recognition Attendance System

Developer's Information:

Name: Md. Fardin Ahsan

Student Id: 221-15-5496

Section: 61_Q

Institution: Daffodil International University

Extend Guideline: Rule of Optimized Performance

This extend addresses to the Rule of Optimized Execution, centering on:

o Minimizing Rebuilds

o Optimizing Rendering

o Network and Information Optimization

o Memory Management

o UI Responsiveness

o Device Compatibility

o Continuous Checking and Optimization

Project Design Patterns:

To proficiently address common issues in computer program plan, this venture utilizes the MVC and MVP building patterns:

MVC Pattern:

The Model-View-Controller (MVC) design is a program plan design that isolates an application into three primary components:

- Model: Handles state administration and commerce rationale. It specifically oversees the information, rationale, and rules of the application.

- View: Capable for rendering UI components. It shows the information to the client and sends client commands to the controller.

- Controller: Reacts to UI activities and upgrades the see in like manner. It acts as an mediator between the demonstrate and the view.

Example:

In a confront acknowledgment framework, the Demonstrate would handle putting away and preparing facial information, the See would show the user's confront and participation status, and the Controller would oversee intelligent such as enlisting a unused confront or logging attendance.

MVP Pattern:

The Model-View-Presenter (MVP) design is a subsidiary of the MVC design, fundamentally utilized for organizing the client interface. It isolates the application rationale from the UI, making it simpler to test and maintain:

- Model: Administers commerce behaviors and state administration. It handles the information portion of the application.

- View: Renders UI components and communicates with the moderator through an interface. It is capable for showing information to the user.

• Presenter: Acts as a broker between the show and the see, dealing with the trade rationale. It recovers information from the demonstrate and applies rationale to choose what to show in the view.

Example:

In the confront acknowledgment framework, the Demonstrate would oversee the facial acknowledgment information, the See would show the acknowledgment comes about, and the Moderator would handle the information and upgrade the see with the participation information.

Planning and Requirements:

The extend will at first be based on the MVP engineering due to its center on UI plan and introduction layer responsibilities.

Key Features:

Face Discovery: The framework can identify any prepared face.

Attendance Logging: Essential structure for logging participation (future work).

User Interface: Plan a user-friendly UI for ease of use.

Development Instruments, Strategies & Environment Explanation:

1. Improvement Tools:

Programming Dialect: Python

Framework: Django for backend advancement and application structure.

Face Acknowledgment Library: OpenCV and face_recognition libraries for confront discovery and recognition.

Database: PostgreSQL for putting away client information and confront acknowledgment data.

Front-end Advances: HTML, CSS, and JavaScript for the client interface.

Version Control: GitHub for source code management.

IDE: VSCode with expansions like Python, live-server, auto-indentation, etc.

Design Instruments: Figma for UI/UX design.

2. Methods:

Development Strategy: Dexterous strategy for iterative and incremental advancement. This includes normal overhauls, criticism cycles, and ceaseless change of the project.

Project Administration: Jira for following venture advance and collaboration. Assignments will be partitioned into sprints, permitting for clear breakthroughs and goals.

3. Environment Explanation:

The improvement environment will comprise of:

Local Improvement: Setting up a neighborhood server utilizing Django for improvement and testing. VSCode will be the essential IDE, prepared with vital expansions for effective coding.

Version Control: Utilizing GitHub for form control guarantees that changes are followed, and different group individuals can collaborate effectively.

Testing Environment: Partitioned testing environment to run unit and integration tests, guaranteeing the application capacities accurately some time recently deployment.

Future Highlights and Advancement of the Project:

Future upgrades will be based on client input and advancing requirements.

Upcoming Features:

Attendance Tally: Actualize usefulness to number and log participation based on confront recognition.

Push Notices: Inform clients of their participation status through thrust notifications.

Profile Administration: Clients can make and oversee their profiles.

Login Framework: Secure login with e-mail and password.

Forgot Secret word: Secret word recuperation choice with mail verification.

Email Confirmation: Confirm client emails with a 6-digit PIN.

Edit Profile: Clients can upgrade their profile data and profile image.

Calendar Integration: Clients can see participation history on a calendar.

Device Compatibility: Guarantee the framework works over different devices.

Data Analytics: Give analytics and reports on participation data.

Overview of Completely Created Venture and Conclusion:

The Online Confront Acknowledgment Participation Framework will begin with the MVP engineering for introductory improvement, centering on the client interface and confront location. Over time, it will join MVC engineering designs to handle more complex commerce rationale and state management.

Using Python and Django will give a vigorous system for creating a versatile and viable application. The introductory dispatch will center on the web stage, with future plans to expand usefulness and back extra highlights as depicted. In spite of introductory confinements, the extend will be completely actualized and kept up for ideal execution and client satisfaction.

This venture points to give a strong, proficient, and user-friendly confront acknowledgment participation framework, streamlining participation administration through progressed innovation and plan standards. The combination of MVP and MVC designs guarantees a versatile, viable, and testable engineering, improving the system's generally unwavering quality and convenience.