Egyptian E-Learning University

Faculty of Computers & Information Technology

Attendance System by Face recognition

**By**

|  |  |
| --- | --- |
| Name | ID |
| Mohamed Mostafa Ibrahim | 2102214 |
| Mohamed Osman Younes | 2101189 |
| Youssef Mohamed Mahmoud | 2101748 |
| Mohamed Abdelmonam Mohamed | 2101223 |
| Salma Abdelhameed Mohamed | 2101541 |
| Shahd Hossam Elden Hussein | 2101611 |
| Eman Abdo Abdelrahman | 2100622 |

Supervised by

**Prof. Khaled Wassef**

Assistant

Eng.Dalia Masoud

[Assiut]-2025

Contents

[Abstract 3](#_Toc197463970)

[Acknowledgments 4](#_Toc197463971)

[Introduction 5](#_Toc197463972)

[Literature Review / Related Work 11](#_Toc197463973)

[Proposed system 14](#_Toc197463974)

[Implementation 29](#_Toc197463975)

[Testing & Evaluation 33](#_Toc197463976)

[Results & Discussion 39](#_Toc197463977)

[Conclusion & Future Work 43](#_Toc197463978)

[References 46](#_Toc197463979)

[Appendices 48](#_Toc197463980)

Abstract

The “Attendance System by Face Recognition” is a graduation project developed for EELU, aiming to automate the process of student attendance using facial recognition technology. The system consists of a web application built with Angular for the frontend and ASP.NET Core for the backend, allowing administrators to manage attendance records in real-time. In addition, a desktop application has been developed to capture attendance offline, ensuring the system works seamlessly even in the absence of internet connectivity. This dual-mode solution enhances reliability, minimizes manual errors, and offers a contactless, secure method of tracking student presence in educational environments.

Acknowledgments

We would like to express our sincere gratitude to our academic supervisor for his continuous support, insightful guidance, and valuable feedback throughout the development of this graduation project.

We also thank the faculty members and administration at EELU for providing a rich educational environment and the technical resources that helped us bring our project to life.

Our heartfelt appreciation goes to our colleagues and peers who supported us with their ideas, collaboration, and encouragement.

Finally, we are deeply thankful to our families and friends for their patience, motivation, and unwavering belief in us during this journey.

Chapter 1

Introduction

## 1.1 Introduction

Traditional methods of recording student attendance often rely on manual entry or card-based systems, which are prone to human errors, manipulation, and inefficiency. In educational institutions, accurate attendance tracking is crucial for both administrative purposes and student performance monitoring.

This project introduces a modern solution: an intelligent attendance system based on facial recognition. By utilizing computer vision techniques and AI-based face detection models, the system can identify and verify students with high accuracy and speed. The proposed system includes a web-based application for real-time attendance management and a desktop application designed to capture offline attendance, ensuring flexibility and continuity in various classroom environments.

This documentation outlines the design, implementation, and testing of the system, which leverages technologies such as Angular, ASP.NET Core, and machine learning models to provide a secure, automated, and user-friendly solution.

## 1.2Background and motivation for the project.

Attendance is a critical aspect of classroom management and academic tracking in educational institutions. Traditionally, institutions rely on manual attendance methods such as calling out names or using ID cards, which are time-consuming and prone to errors or misuse. With the increasing adoption of technology in education, there's a growing need for smarter and more reliable attendance solutions.

The motivation behind this project stems from the desire to automate the attendance process using biometric technology, especially facial recognition, which offers a fast, secure, and contactless alternative. Additionally, the implementation of both online and offline attendance methods ensures flexibility and continuity in various learning environments.

## 1.3Importance of the problem being addressed.

Manual attendance systems can lead to significant administrative inefficiencies, inaccurate data collection, and even intentional fraud such as "proxy attendance." These issues can negatively impact academic records and institutional decision-making. By using facial recognition, the proposed system eliminates the potential for impersonation and reduces human effort, providing a reliable solution for daily attendance tracking.

Furthermore, integrating both online and offline modes ensures that institutions without consistent internet access can still benefit from the system, increasing its usability and impact.

## 1.4Problem Statement

Many educational institutions still rely on traditional attendance-taking methods that are inefficient, error-prone, and lack security. These methods often require manual input, are time-consuming, and susceptible to manipulation.

The problem addressed in this project is the lack of an automated, reliable, and secure system for student attendance that works both online and offline. Solving this problem is important to improve accuracy in student tracking, reduce administrative workload, and enable real-time reporting and analytics.

## 1.5Objectives

**Main Objective:**

To design and implement an intelligent attendance system that uses facial recognition technology to automate and secure the process of student attendance in both online and offline environments.

Specific Objectives:

* Develop a web application with Angular and ASP.NET Core for real-time attendance tracking and administration.
* Build a desktop application for capturing attendance offline using facial recognition.
* Integrate a facial recognition model capable of identifying and verifying students with high accuracy.
* Design a central database to store student records, attendance logs, and related data.
* Ensure synchronization between the online and offline systems when internet connectivity is restored.
* Provide an easy-to-use interface for administrators and staff.

## 1.6 Brief overview of the proposed solution.

The proposed system consists of two main components: a web application and a desktop application. The web application, developed with Angular and ASP.NET Core, allows administrators to manage students, view attendance records, and access reports. The desktop application enables attendance capturing in offline environments using face recognition technology.

Both systems are integrated with a centralized database that synchronizes attendance data. The facial recognition feature ensures secure and contactless attendance, making the system suitable for a wide range of educational settings. This hybrid solution improves efficiency, accuracy, and reliability in attendance management.

Chapter 2

Literature Review / Related Work

## 2.1 Summary of existing research and technologies related to your project.

Face recognition technology has been widely explored and applied in areas such as security, surveillance, and access control. In the educational field, recent research has demonstrated the feasibility of using facial recognition for automating attendance systems. Most solutions rely on computer vision libraries such as OpenCV, Dlib, or deep learning models like FaceNet and MTCNN to detect and recognize faces with high accuracy.

Modern attendance systems are often integrated with web technologies or mobile applications to enhance accessibility. Web-based systems allow real-time attendance tracking, while some institutions have begun incorporating biometric attendance devices. Many academic papers highlight the effectiveness of facial recognition systems in reducing fraud and saving time

## 2.2 Gaps in current solutions that your project aims to fill.

While several face recognition attendance systems exist, most of them have significant limitations:

* They rely solely on online functionality and cannot operate in offline environments.
* Many lack real-time synchronization between offline and online records.
* Some systems are not user-friendly or require expensive dedicated hardware.
* Security and data privacy are often overlooked in open-source implementations.

These gaps present an opportunity for developing a more practical and robust solution that ensures both **connectivity independence** and **scalable integration** into educational environments.

## 2.3 Summary

In summary, while face recognition technologies are not new, their application in attendance systems still has room for improvement, especially when addressing reliability in low-connectivity settings and providing flexible administration tools.

The proposed system distinguishes itself by offering a hybrid model that functions both **online and offline**, ensuring accurate attendance tracking using facial recognition, with a centralized web platform for administration and analytics. This helps bridge existing gaps and brings a more holistic solution to student attendance management.

Chapter 3

Proposed system

## 3.1 Approach used to solve the problem

The proposed solution is a hybrid attendance system that uses facial recognition to automate student attendance in both online and offline modes. The system consists of:

* A web application for administrative tasks, built with Angular and ASP.NET Core.
* A desktop application that captures attendance offline and syncs data when internet is available.
* A facial recognition model that identifies students through camera input.  
  This dual-system architecture ensures flexibility, security, and scalability for educational institutions.

## 3.2 System architecture overview.

A diagram of a software application

AI-generated content may be incorrect.

Figure 3.1: System Architecture Overview

The proposed Smart Attendance System consists of three main layers:

* **Frontend Application**: Includes the Angular-based Website App used by instructors and attendees to access system functionalities online, and a Desktop App (built with Tkinter) used primarily for offline attendance capturing.
* **Backend Server**: A .NET 8 Web API serves as the core application layer, handling requests from both the web and desktop applications. It processes facial data, stores images in a file system, and manages interactions with the database.
* **Classification Service**: The face recognition and classification module is integrated into the backend to identify and verify attendees.
* **SQL Database**: All attendance records, user data, and system logs are stored in a centralized SQL Server database, which can be accessed and updated by the backend API.

Use Case Diagram

A diagram of a system

AI-generated content may be incorrect.

Figure 3.2: Use Case Diagram for Attendance Management System.

The use case diagram illustrates how different types of users (Admin, Instructor, Attendee) interact with the system. The main functional areas are:

* **Authentication / Authorization:** Users can register, log in, and choose their role. Registration includes uploading a personal image used for face recognition.
* **Attendance Component:** Instructors can take attendance, which includes recognizing a face or scanning a QR code, and can generate reports.
* **Attendees Component:** Attendees can view their subjects and attendances, and manage their own subject enrollment.
* **Instructors Component:** Instructors can manage subjects they teach, including viewing, adding, and deleting subjects.
* The use case model helps visualize system requirements from a user interaction perspective.

Sequence Diagram 1

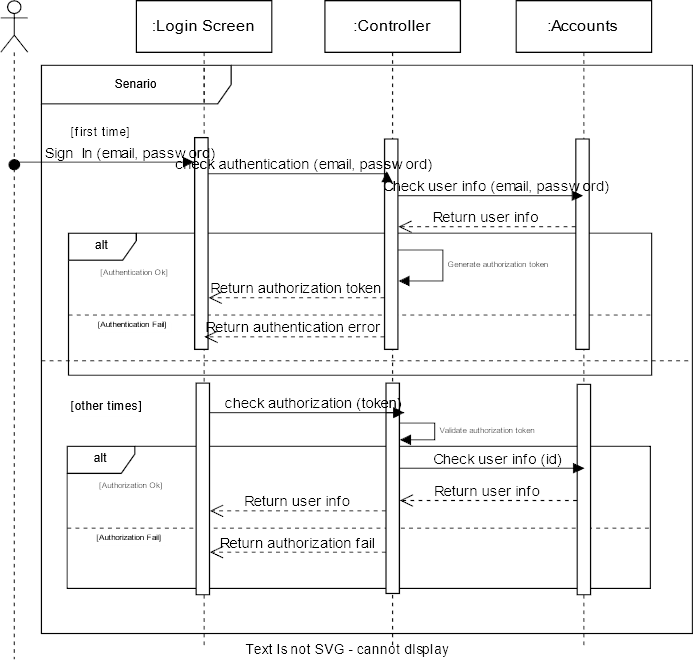


Figure 3.3: Sequence Diagram – Login and Authorization Flow

Sequence Diagram 2

A diagram of a face recognition method

AI-generated content may be incorrect.

Figure 3.4: Sequence Diagram for Attendance Registration via Face Recognition

This diagram illustrates the step-by-step interaction between system components during the attendance process using facial recognition. The flow is as follows:

1. The **user** initiates the attendance process through the attendance screen.
2. The image is sent to the **controller**, which forwards it to the **Face Recognition Method**.
3. The method attempts to identify the individual based on the provided image.
4. If the identity is **found**, the system registers the attendance in the **database** and returns a successful status.
5. If the identity is **not found**, a failure status is returned.
6. This sequence ensures secure and reliable attendance marking using biometric data, and reflects the core logic of the offline and online systems alike.

Class Diagram

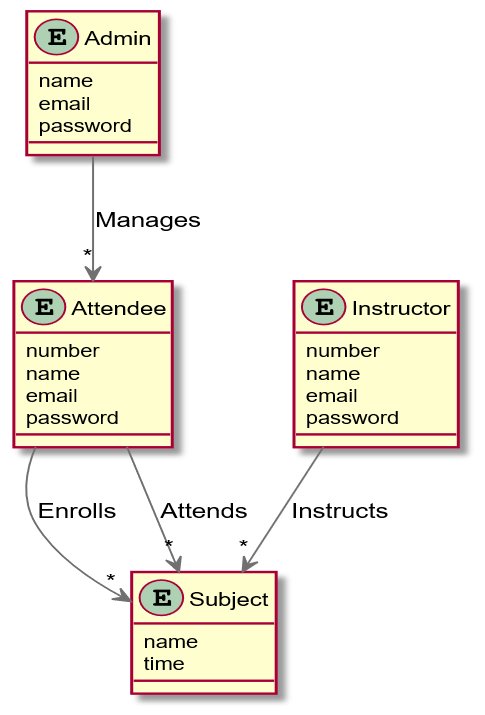


Figure 3.5: Class Diagram of the Attendance Management System

The class diagram presents the core entities of the system and their relationships:

* **Admin** manages multiple attendees.
* **Attendees** can enroll in and attend multiple subjects.
* **Instructors** are responsible for teaching subjects.
* **Subjects** serve as the central entity linked to both attendees and instructors.

Each class contains relevant attributes such as name, email, and password. The relationships (e.g., *manages*, *enrolls*, *instructs*) reflect the real-world interactions between users and the system’s academic structure.

Database Scheme

A computer screen shot of a computer

AI-generated content may be incorrect.

Figure 3.6: Database Schema of the Attendance System

The schema includes core tables such as AspNetUsers, Attendees, Instructors, Subjects, and Attendances. It uses ASP.NET Identity for authentication and authorization, which is visible through tables like AspNetRoles, AspNetUserRoles, and related claims tables.

Relationships between entities:

* Attendees and Subjects have a many-to-many relation via AttendeeSubjects.
* Each Instructor can be assigned to one or more Subjects.
* Attendances track attendance per Attendee and Subject.
* SubjectDates define the schedule of each subject.

## This normalized schema ensures data consistency, integrity, and scalability for both real-time and offline attendance operations.

## 3.3 Algorithms or frameworks used.

## 3.3.1 Face Recognition Algorithm

The face recognition process in this system is powered by a **pre-trained Efficient-B2 model**, which is known for its balance between speed and accuracy. The overall pipeline includes:

1. **Face Detection** – Faces are detected in real-time from webcam input using OpenCV.
2. **Face Embedding** – The Efficient-B2 model is used to extract deep facial features and generate embeddings for each captured face.
3. **Face Matching** – The generated embedding is compared against stored embeddings using Euclidean distance. A match is confirmed when the distance is below a defined threshold.

This lightweight and optimized model allows for fast and reliable identification, suitable for both desktop and web environments.

## 3.3.2 Frameworks and Technologies Used

The system is built using a combination of modern technologies and frameworks:

* **Frontend:**
  + **Angular** – for building the responsive web interface.
* **Backend:**
  + **ASP.NET Core 8** – for building secure RESTful APIs.
  + **Entity Framework Core** – for database access and migration management.
  + **SQL Server** – for storing all user, attendance, and facial embedding data.
* **Desktop Application:**
  + **Python (with Tkinter)** – for building the offline attendance app.
  + **OpenCV / Dlib** – for image capture and face detection.
  + **Face Recognition library** – for face encoding and comparison.
* **Security and Auth:**
  + **ASP.NET Identity** – handles user registration, login, role-based access, and token management.
* **Others:**
  + **File System Storage** – used to store captured face images if required.
  + **Sync Mechanism** – ensures offline attendance records are pushed to the server once online.

Chapter 4

Implementation

## 4.1 Technologies, tools, and programming languages used.

The system was developed using a range of tools and technologies, each selected based on performance, compatibility, and ease of integration. The main technologies used include:

* **Frontend:**
  + Angular 15 (TypeScript, HTML, SCSS)
  + Angular Material for UI components
* **Backend:**
  + ASP.NET Core 8 (C#)
  + Entity Framework Core for ORM
  + SQL Server for relational database
  + ASP.NET Identity for user authentication and role management
* **Desktop Application:**
  + Python 3 with Tkinter for GUI
  + OpenCV for image capture
  + **Efficient-B2** pre-trained model for face embedding
* **Tools:**
  + Visual Studio 2022
  + Visual Studio Code
  + SQL Server Management Studio (SSMS)
  + Postman for API testing
  + Git and GitHub for version control

## 4.2 Key components/modules of the system.

The system is modular and divided into major components as follows:

1. **Frontend Module (Angular):**
   * Handles UI and UX for instructors and admins.
   * Manages login, user roles, student and subject views, and real-time attendance dashboards.
2. **Backend API (ASP.NET Core):**
   * Processes HTTP requests from frontend and desktop app.
   * Contains authentication, business logic, attendance operations, and database access.
3. **Face Recognition Module (Python):**
   * Detects and encodes facial features using the Efficient-B2 model.
   * Compares embeddings and returns match status to mark attendance.
4. **Desktop Application:**
   * Offers offline attendance capture.
   * Stores temporary records and syncs them with the server once internet is available.
5. **Database Module (SQL Server):**
   * Stores user accounts, roles, facial embeddings, subjects, and attendance logs.
   * Enforces relationships between entities such as instructors, attendees, and sessions.

## 4.3 Challenges faced and how they were resolved.

Developing a hybrid online-offline attendance system presented several technical and practical challenges:

**Offline Synchronization:**  
Ensuring that offline attendance data is properly synced with the server once the internet is available was a major concern.  
 Solution: Implemented a local buffer in the desktop app to store records and push them to the API upon reconnecting.

**Facial Recognition Accuracy:**  
Lighting conditions and webcam quality affected the accuracy of facial recognition.  
 Solution: Used the Efficient-B2 model for better feature extraction and applied preprocessing like image resizing and contrast adjustment.

**Data Security:**  
Sensitive user data, including facial embeddings, needed to be stored securely.  
 Solution: Applied hashing for passwords using ASP.NET Identity, and restricted access to image files via role-based authorization.

**Cross-technology Integration:**  
Integrating Python (for face recognition) with a .NET backend and Angular frontend required well-defined interfaces.  
 Solution: Used RESTful APIs to handle all communication, maintaining clear separation between components.

Chapter 5

Testing & Evaluation

## 5.1 Testing strategies (unit testing, integration testing, user testing).

The system was tested using several testing approaches to ensure stability, correctness, and user satisfaction:

* **Unit Testing:**  
  Core backend services were unit tested using xUnit and Moq to validate business logic such as attendance marking, subject assignment, and user role verification.
* **Integration Testing:**  
  RESTful API endpoints were tested using Postman to verify request/response cycles, database interactions, and error handling between frontend, backend, and desktop components.
* **User Testing (UAT):**  
  A group of instructors and students tested the system in a simulated classroom environment. Feedback was collected regarding usability, accuracy, and speed, which informed multiple UI/UX improvements.
* **Offline Testing:**  
  The desktop app was tested in complete offline scenarios. Attendance records were verified for correct syncing when reconnected to the network.

## 5.2 Performance metrics (accuracy, speed, scalability, etc.).

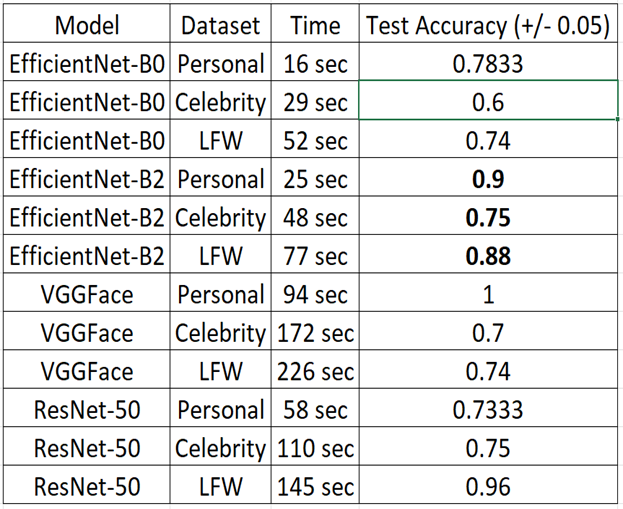
Several aspects of system performance were evaluated using real-world datasets and benchmark models:

**Face Recognition Accuracy:**  
The EfficientNet-B2 model demonstrated **excellent accuracy across different datasets**, outperforming many alternatives in terms of both speed and precision:

**EfficientNet-B2 achieved 0.90 accuracy on personal datasets while maintaining relatively low processing time (25–77 seconds) compared to heavier models like VGGFace and ResNet-50, making it a well-balanced choice for real-time educational environments.**

* **Response Time (Web API):  
  Average API response time for attendance submission: ~400ms**
* **Offline Sync Speed:  
  Attendance records from the desktop app sync with the server within 2–3 seconds of internet reconnection.**
* **System Scalability:  
  The system is capable of managing thousands of records across users, subjects, and attendances without significant performance degradation, thanks to efficient SQL indexing and modular architecture.**

5.3 A brief comparison with common market or academic attendance solutions:



A graph of different colored lines

AI-generated content may be incorrect.A graph of different colored lines

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.

The proposed system outperforms existing methods by offering **face-based attendance with offline functionality**, enhanced accuracy, and centralized data management — while still being cost-effective.

Chapter 6

Results & Discussion

## 6.1 Introduction

This chapter presents the outcomes of the system implementation and testing. It highlights the key findings, evaluates the effectiveness of the proposed solution, discusses whether the project objectives were met, and acknowledges any limitations encountered during development or deployment.

## 6.2 Summary of findings.

Based on testing and real-world evaluation, the following findings were observed:

* The face recognition module using EfficientNet-B2 achieved high accuracy on personal datasets (up to **90%**) with reasonable processing speed.
* The web application successfully handled user authentication, attendance management, and reporting with low latency.
* The desktop application functioned reliably in offline mode and synchronized data correctly upon reconnection.
* Administrators and instructors were able to manage students, subjects, and attendances with minimal training.
* User feedback indicated the interface was user-friendly and the process of capturing attendance was fast and intuitive

## 6.3 Interpretation of results (Did the project meet its objectives?).

The results clearly indicate that the system met its primary and specific objectives:

* **Main Objective:** Automating attendance through facial recognition was achieved, and both online and offline modes were successfully integrated.
* **Specific Objectives:** Each task (web app, desktop app, face model integration, database design, syncing) was implemented and tested with positive outcomes.

The system not only fulfilled the functional requirements but also introduced reliability, security, and a modern user experience that addressed the inefficiencies of traditional attendance systems

## 6.4 Limitations of the proposed solution.

Despite the successful implementation, the system has a few limitations:

* **Face Recognition Sensitivity:** Performance may degrade in poor lighting conditions, side profiles, or when facial features are obstructed.
* **Hardware Dependency:** The accuracy and speed of recognition depend on the quality of the webcam used, which may vary across devices.
* **Scalability with High Traffic:** While the system handles normal load efficiently, large-scale concurrent usage was not tested under real institutional scale.
* **Data Privacy Concerns:** Although security measures were taken, storing facial data still presents ethical and legal considerations in real-world deployment.

Future improvements can address these issues through hardware calibration, enhanced preprocessing, and more robust privacy policies.

Chapter 7

Conclusion & Future Work

## 7.1 Summary of contributions.

This project introduced a hybrid facial recognition-based attendance system that operates both online and offline. The key contributions of the work include:

* Designing and implementing a complete web-based attendance system using Angular and ASP.NET Core.
* Developing a desktop application that works offline and synchronizes data with the server when reconnected.
* Integrating a pre-trained **EfficientNet-B2** model for facial recognition, achieving high accuracy and performance.
* Building a secure and scalable backend with proper role-based access control, data storage, and API communication.
* Ensuring user-friendly interfaces for both students and administrators, reducing reliance on manual systems and minimizing attendance fraud

## 7.2 Possible improvements or extensions for future work.

Although the system meets its intended objectives, several enhancements could be explored in future iterations:

* **Mobile Application Integration:**  
  A native mobile application can be developed using **Flutter** to allow students and instructors to interact with the system on-the-go. The mobile app would utilize the same backend and authentication system, ensuring seamless integration with the current infrastructure.
* **Improved Face Detection under Challenging Conditions:**  
  Incorporate additional preprocessing techniques or switch to more advanced models for better performance in low-light or side-profile scenarios.
* **Liveness Detection:**  
  To enhance security and avoid spoofing (e.g., using photos), implement a liveness detection module that verifies the user is a real person during attendance.
* **Cloud Deployment and Multi-Institution Support:**  
  Scale the system to serve multiple educational institutions with isolated databases and user groups using cloud-based architecture.
* **Automated Reporting & Notifications:**  
  Add automated report generation, email alerts for absentees, and real-time attendance dashboards for better administration.

References

[1] M. Tan and Q. V. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” *arXiv preprint arXiv:1905.11946*, 2019. [Online]. Available: <https://arxiv.org/abs/1905.11946>

[2] OpenCV Library. [Online]. Available: <https://opencv.org/>

[3] Face Recognition using Dlib and face\_recognition library. [Online]. Available: <https://github.com/ageitgey/face_recognition>

[4] Microsoft Docs, “ASP.NET Core Documentation.” [Online]. Available: <https://learn.microsoft.com/en-us/aspnet/core/>

[5] Microsoft Docs, “Entity Framework Core Documentation.” [Online]. Available: <https://learn.microsoft.com/en-us/ef/core/>

[6] SQL Server Documentation. [Online]. Available: <https://learn.microsoft.com/en-us/sql/sql-server/>

[7] Angular Documentation. [Online]. Available: https://angular.io/docs

[8] Python Tkinter Documentation. [Online]. Available: <https://docs.python.org/3/library/tk.html>

[9] Postman API Platform. [Online]. Available: <https://www.postman.com/>

[10] GitHub, “Visual Studio Code.” [Online]. Available: <https://code.visualstudio.com/>

[11] API Documentation – Attendance System by Face Recognition. [Online]. Available: [Swagger UI](http://aps.tryasp.net/swagger/index.html)

Appendices

- Additional diagrams, code snippets, user manuals, or datasets.

**API Docs**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

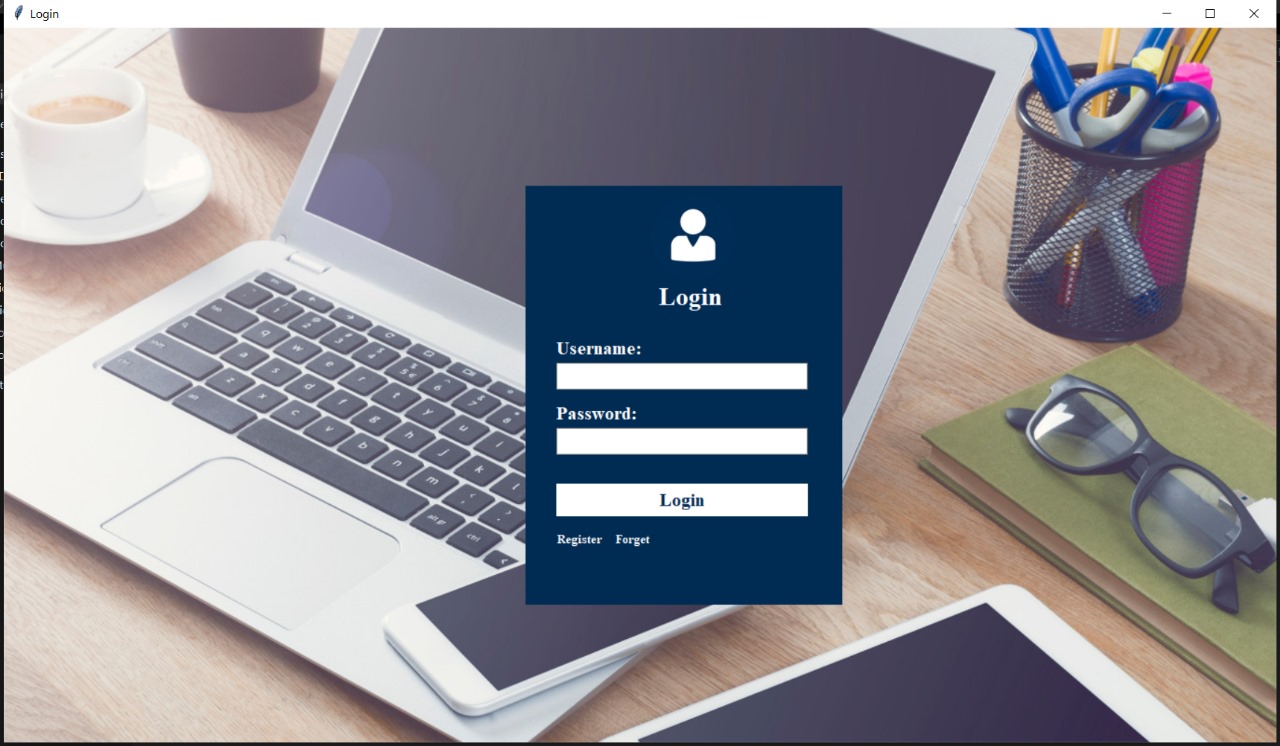
AI-generated content may be incorrect.

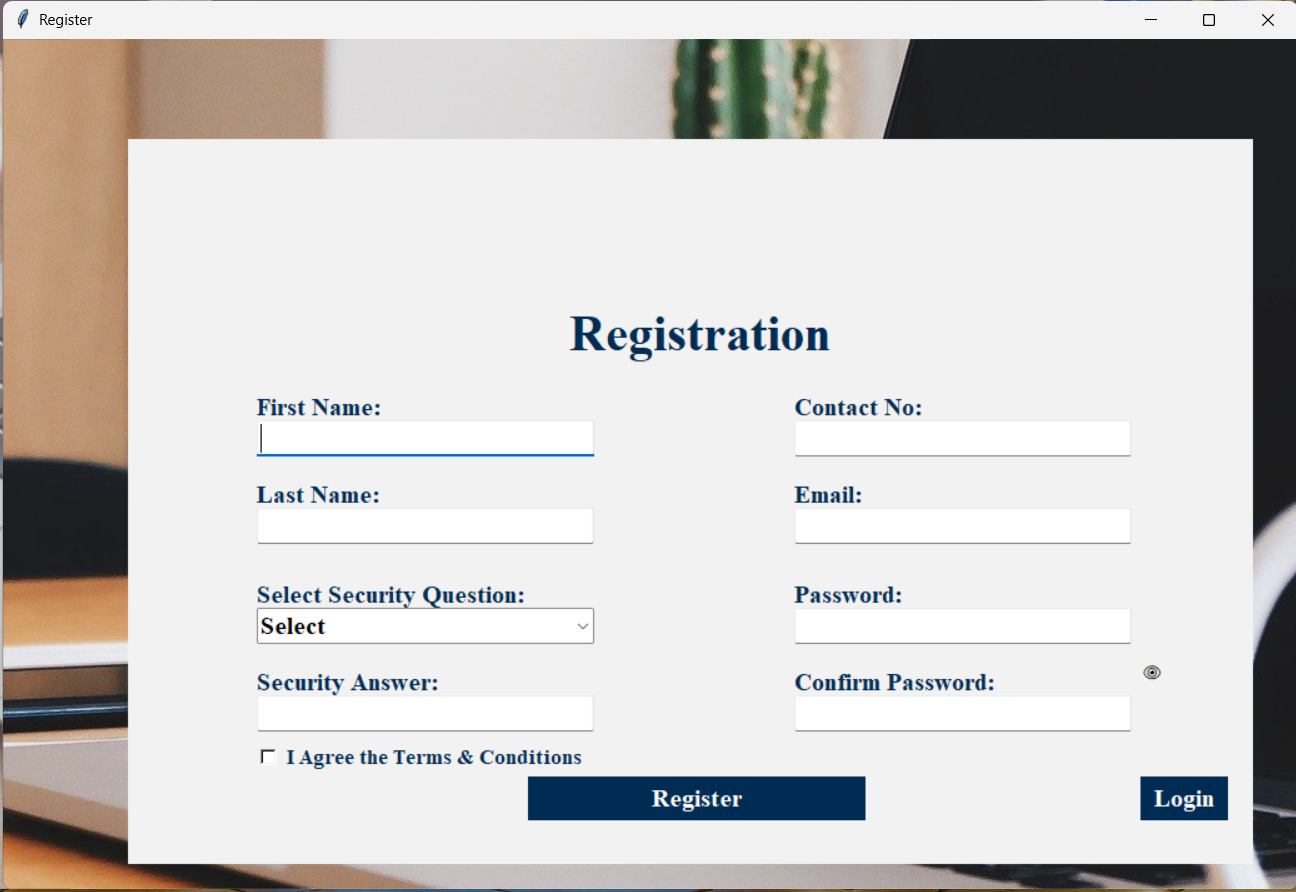
## Time Plan

A diagram of a project

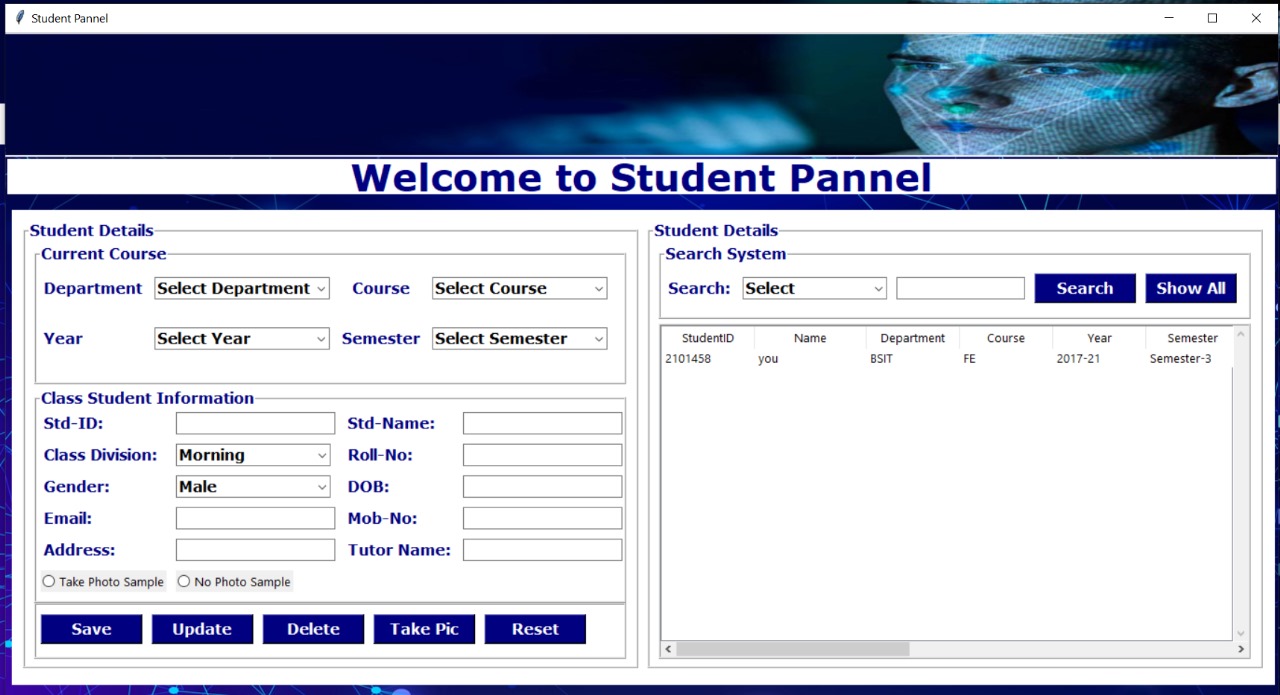
AI-generated content may be incorrect.

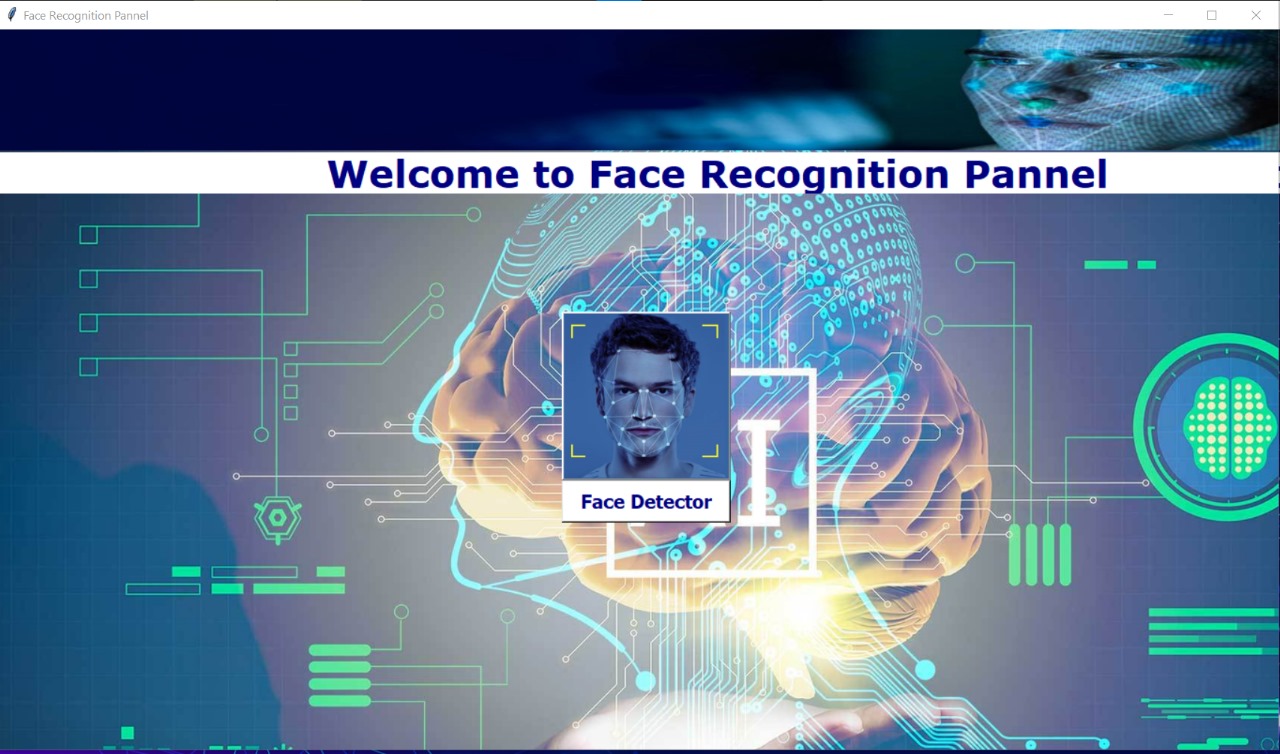
**Desktop Application**

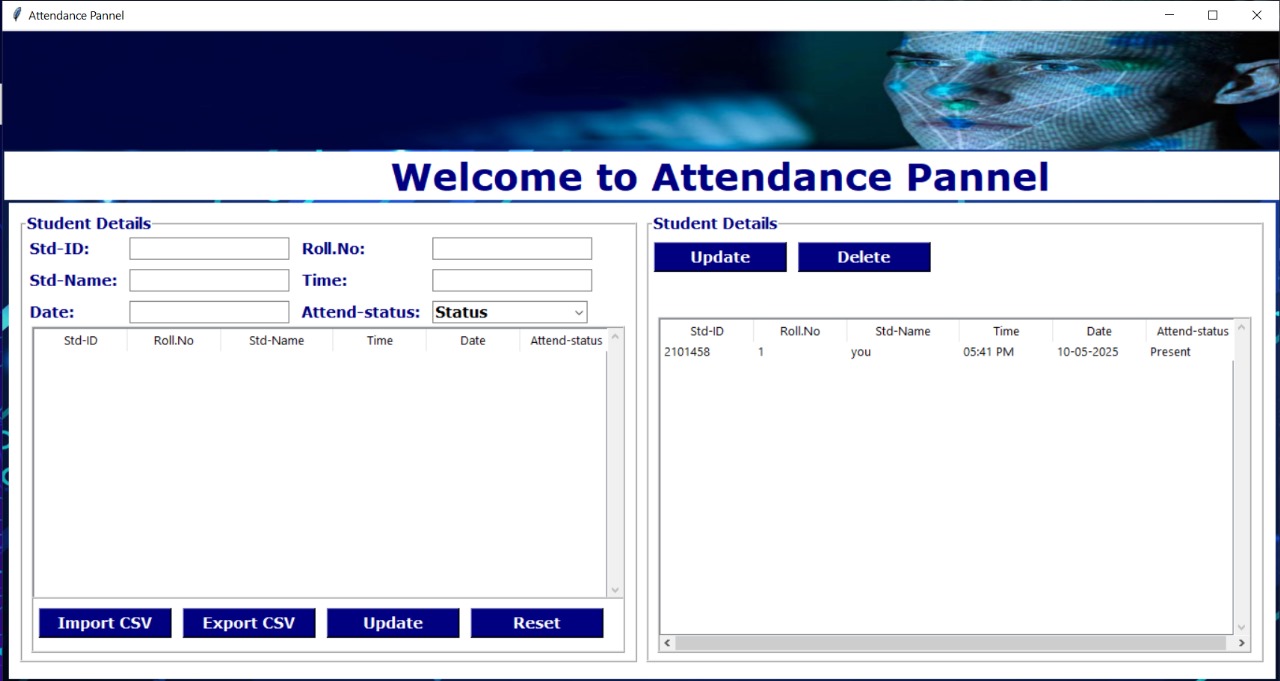


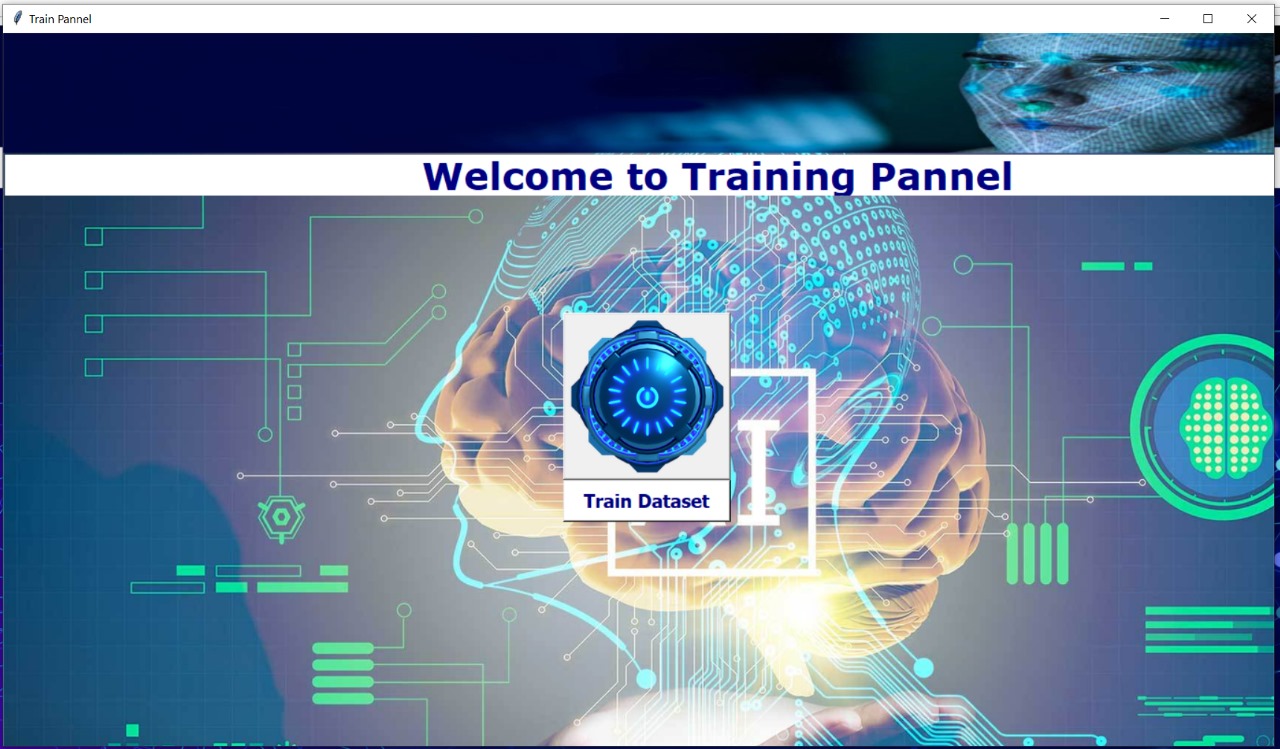




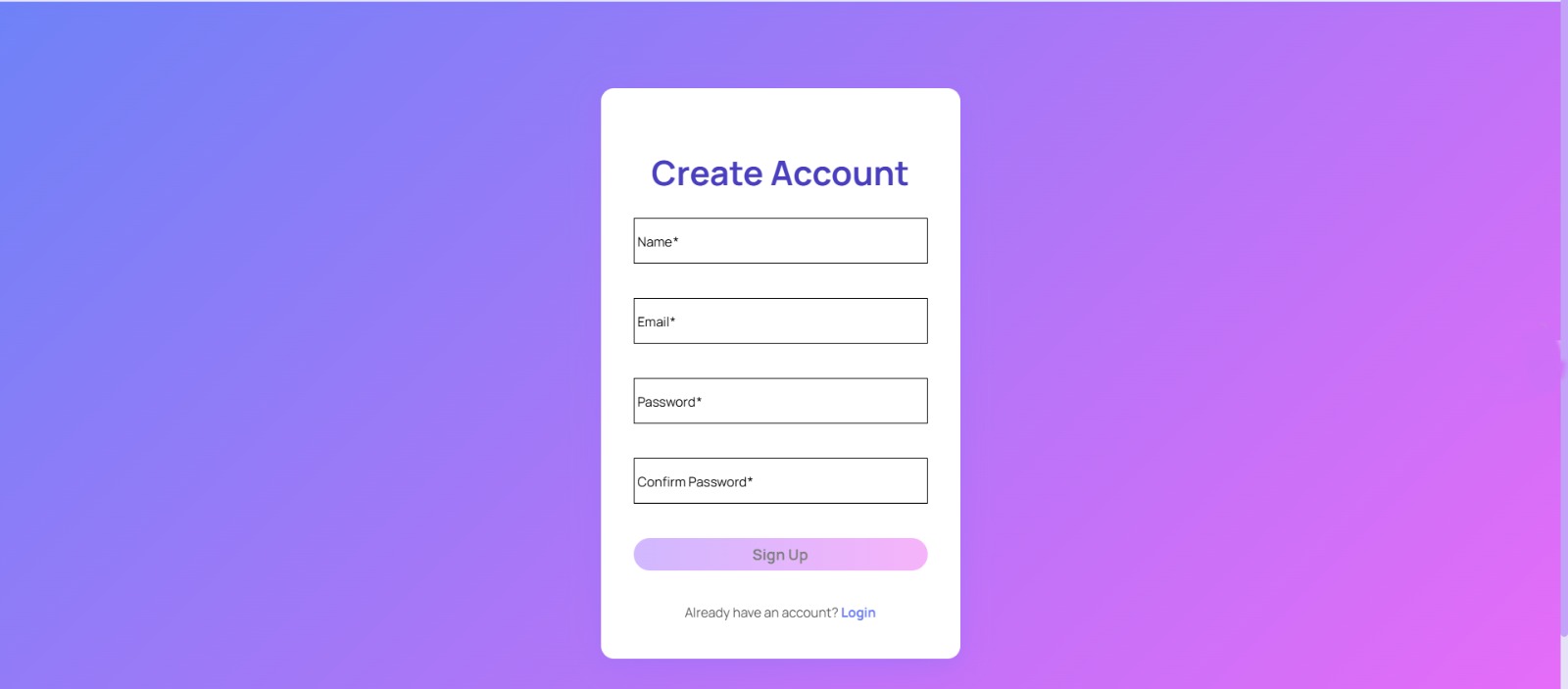


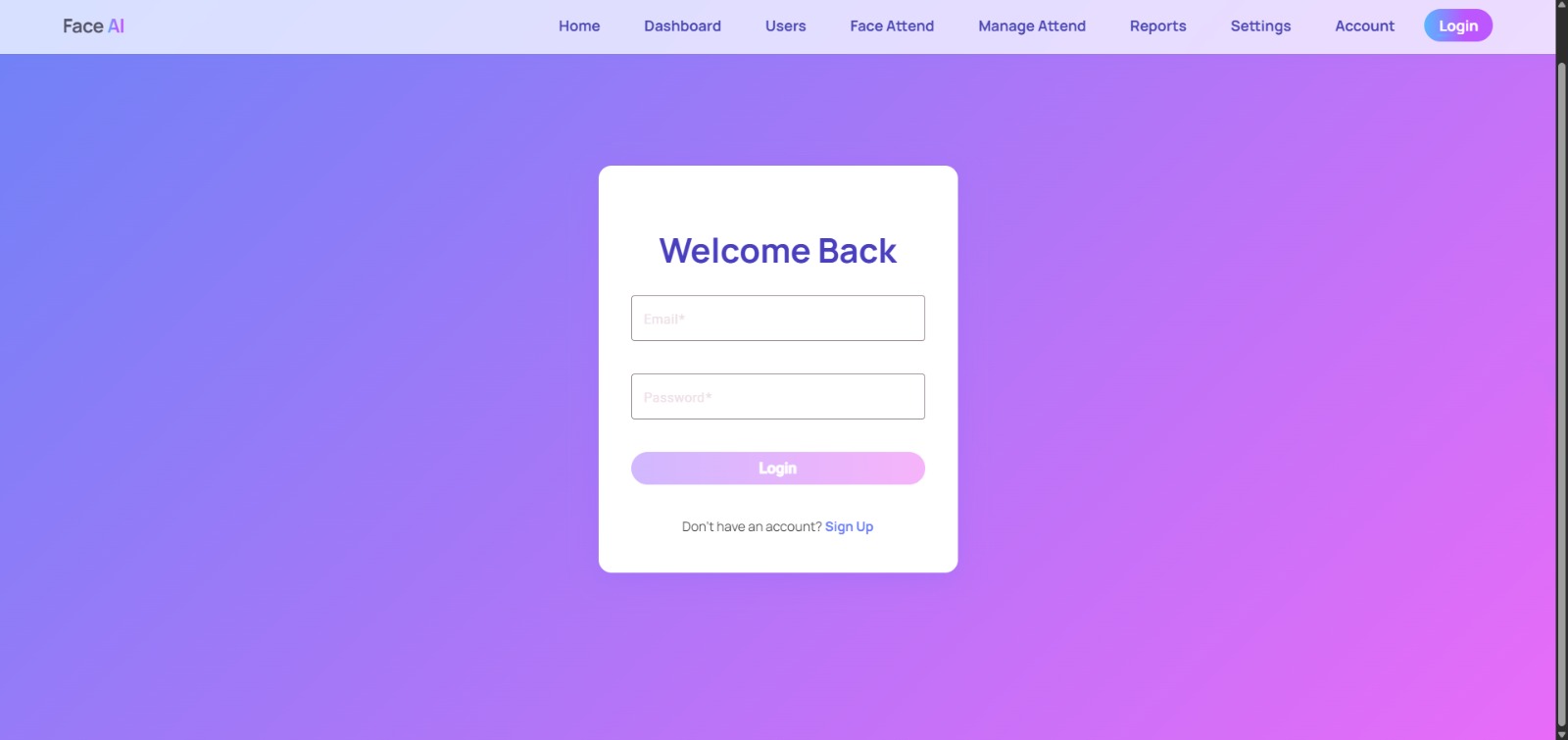


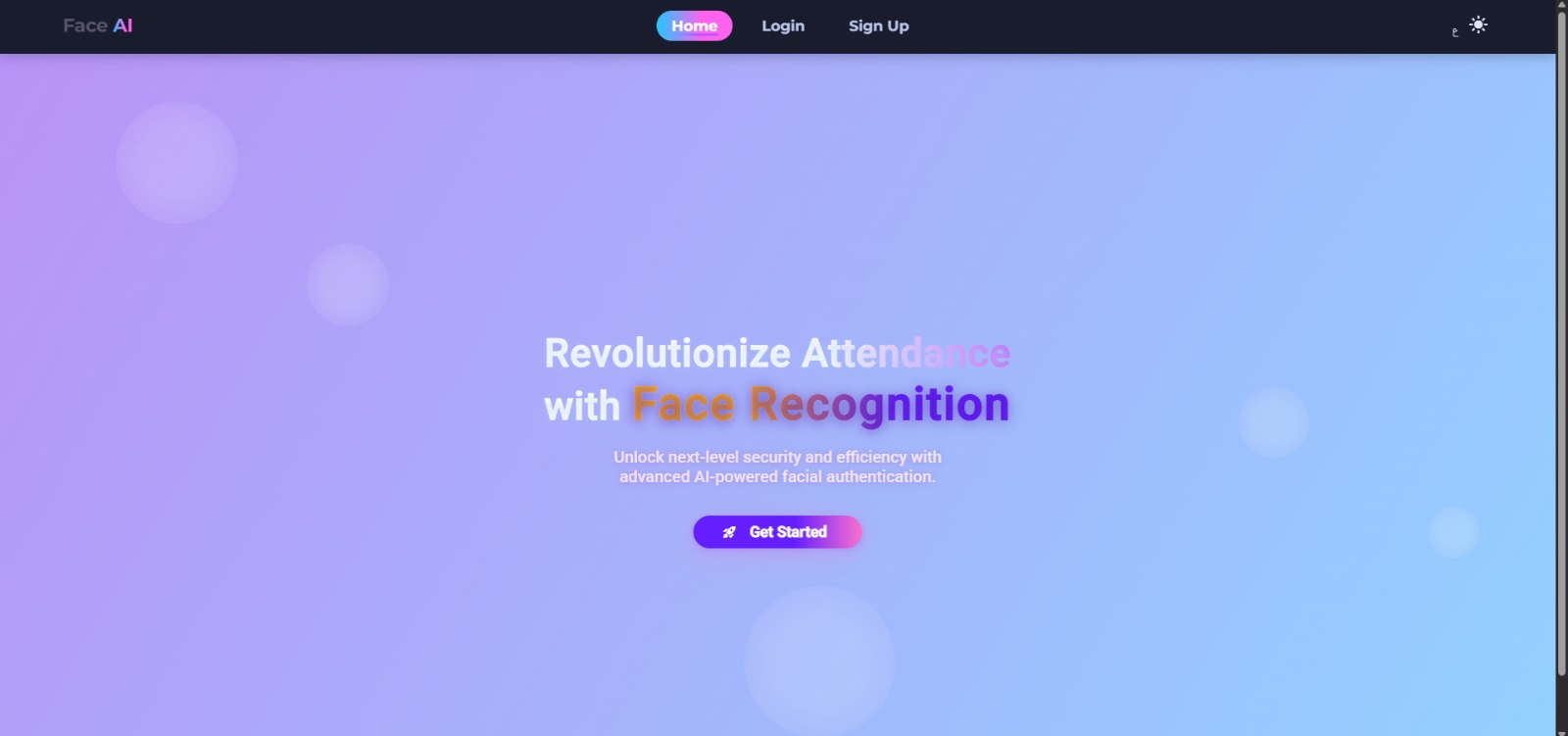


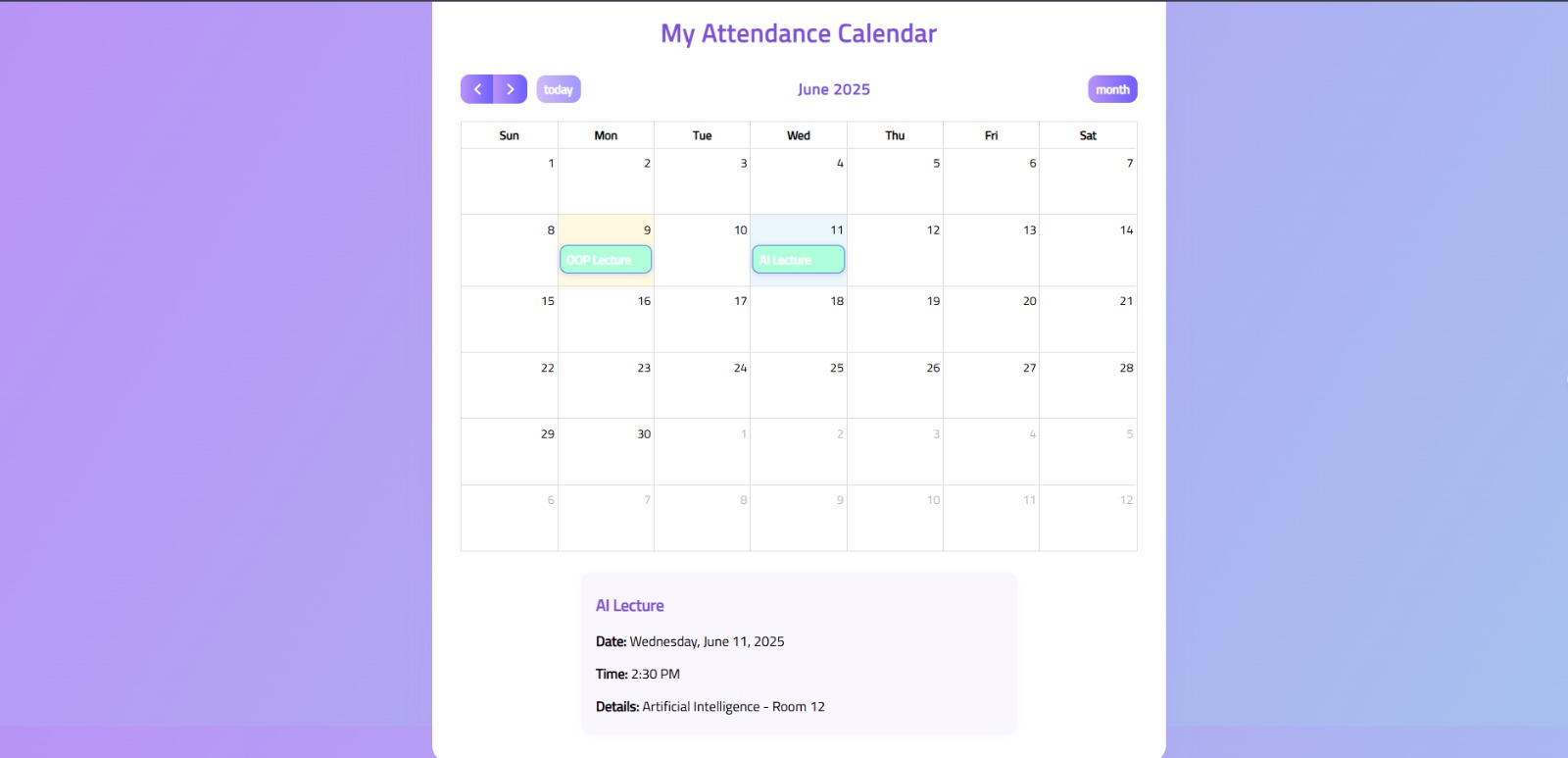


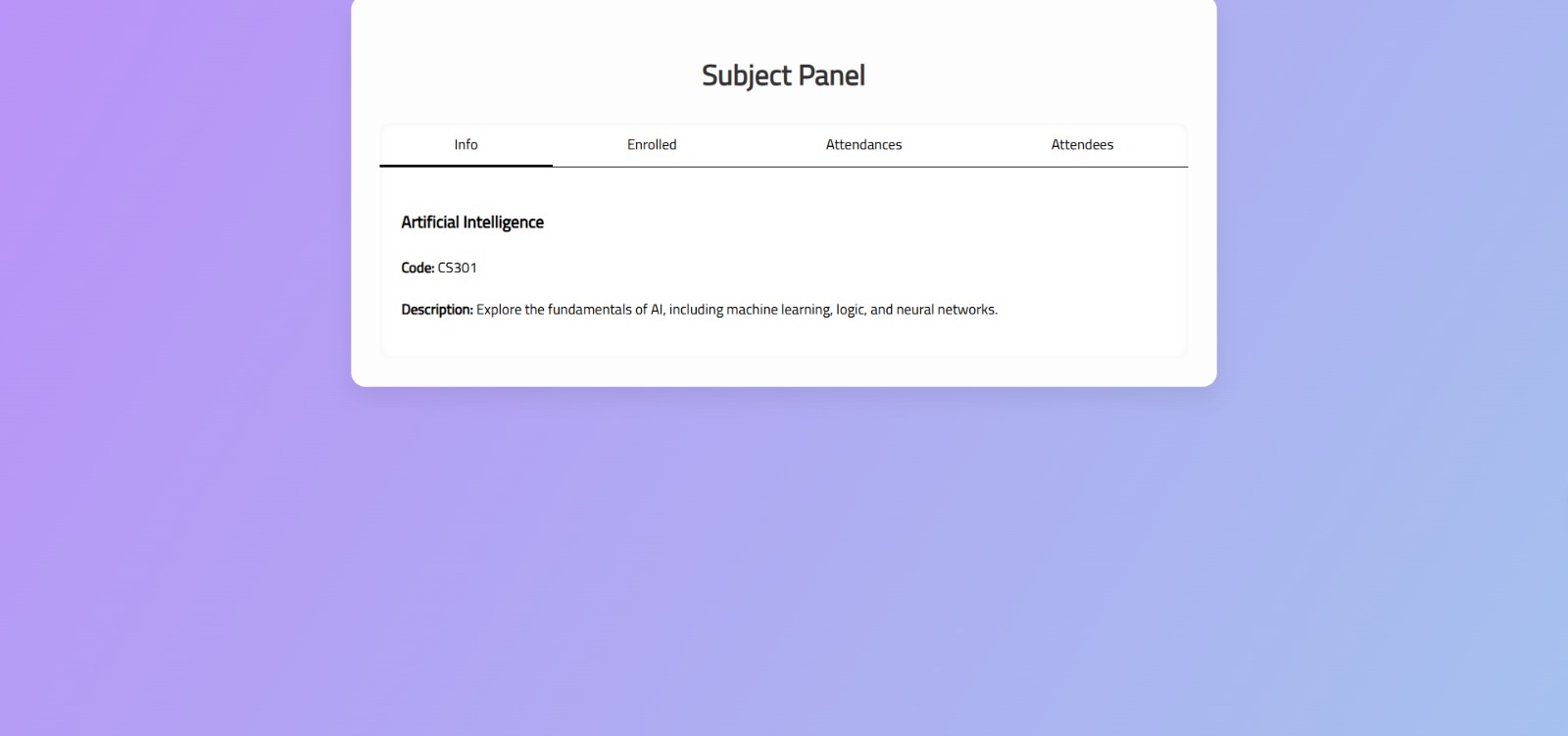
Website Application

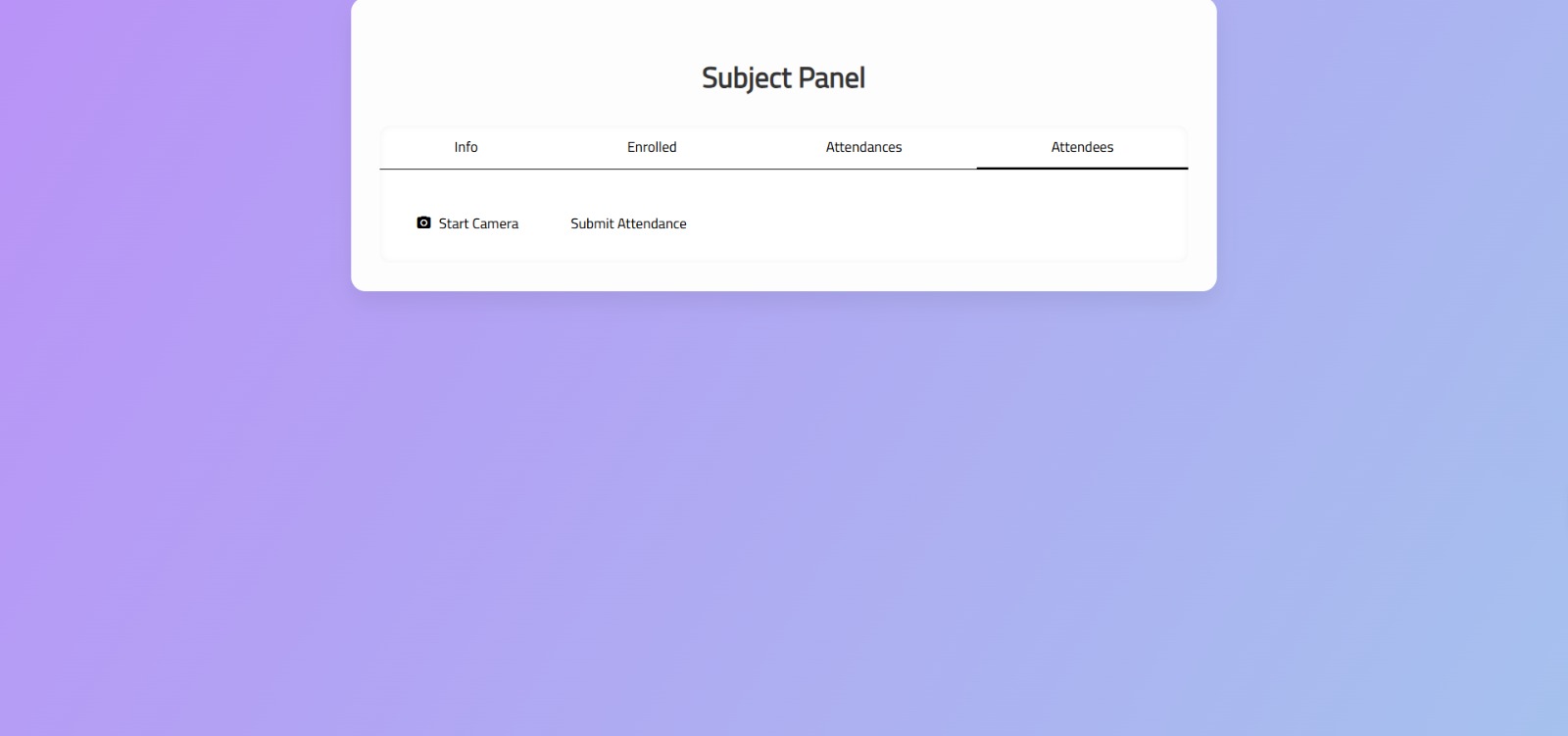


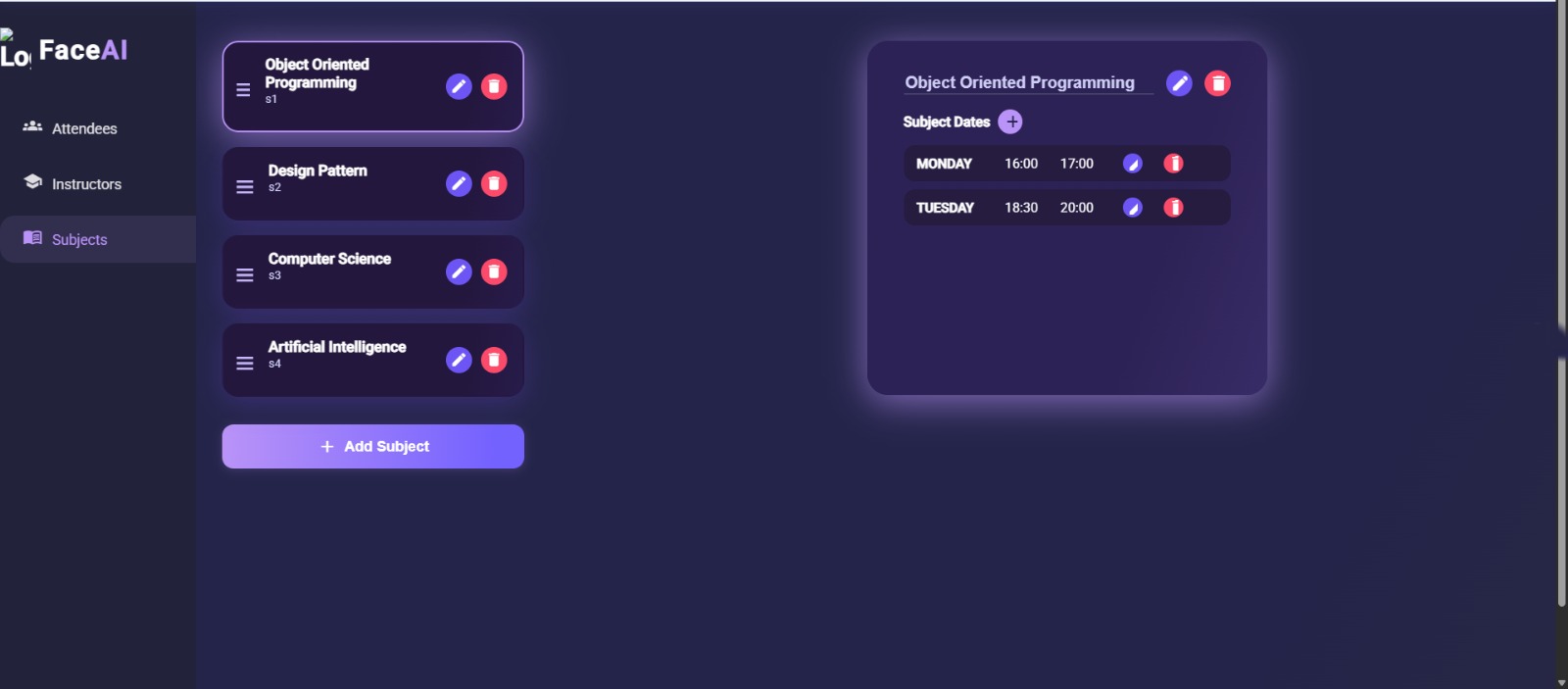


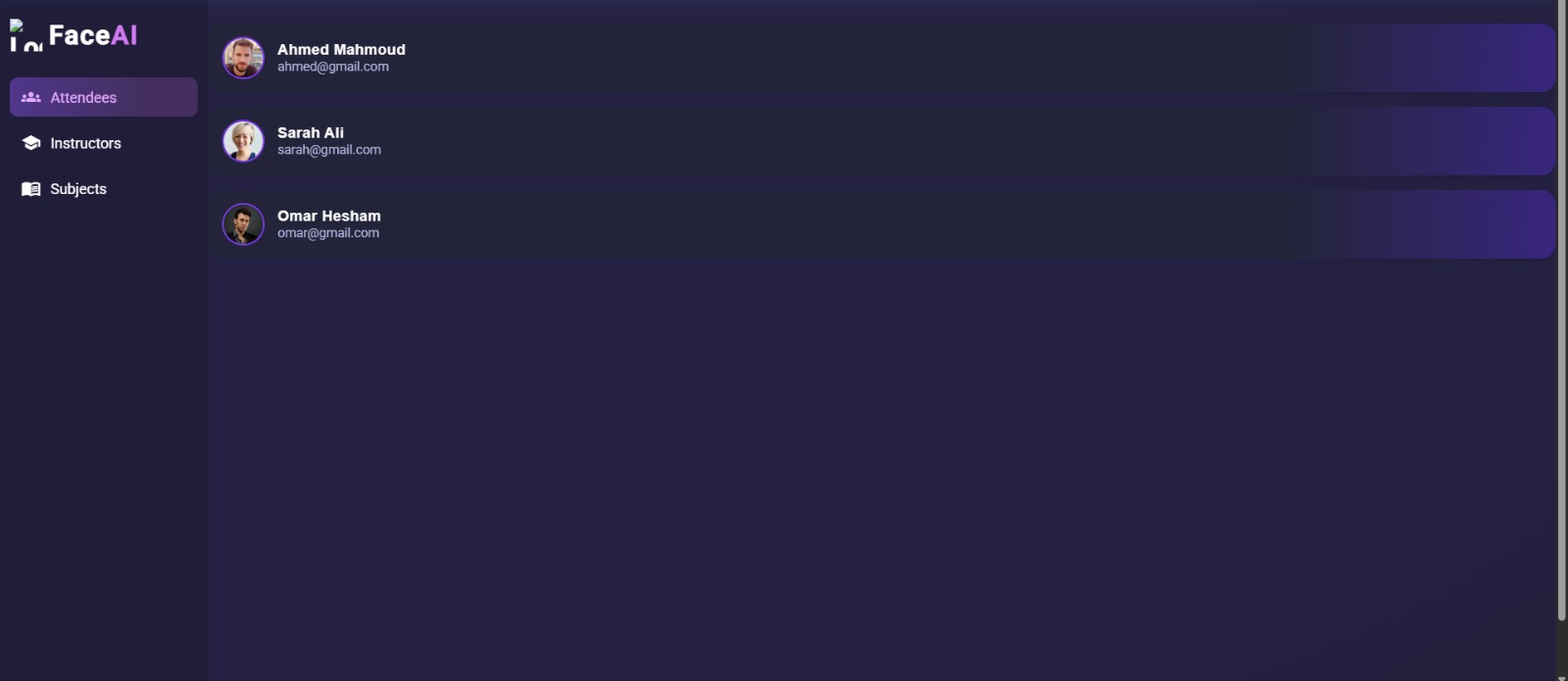


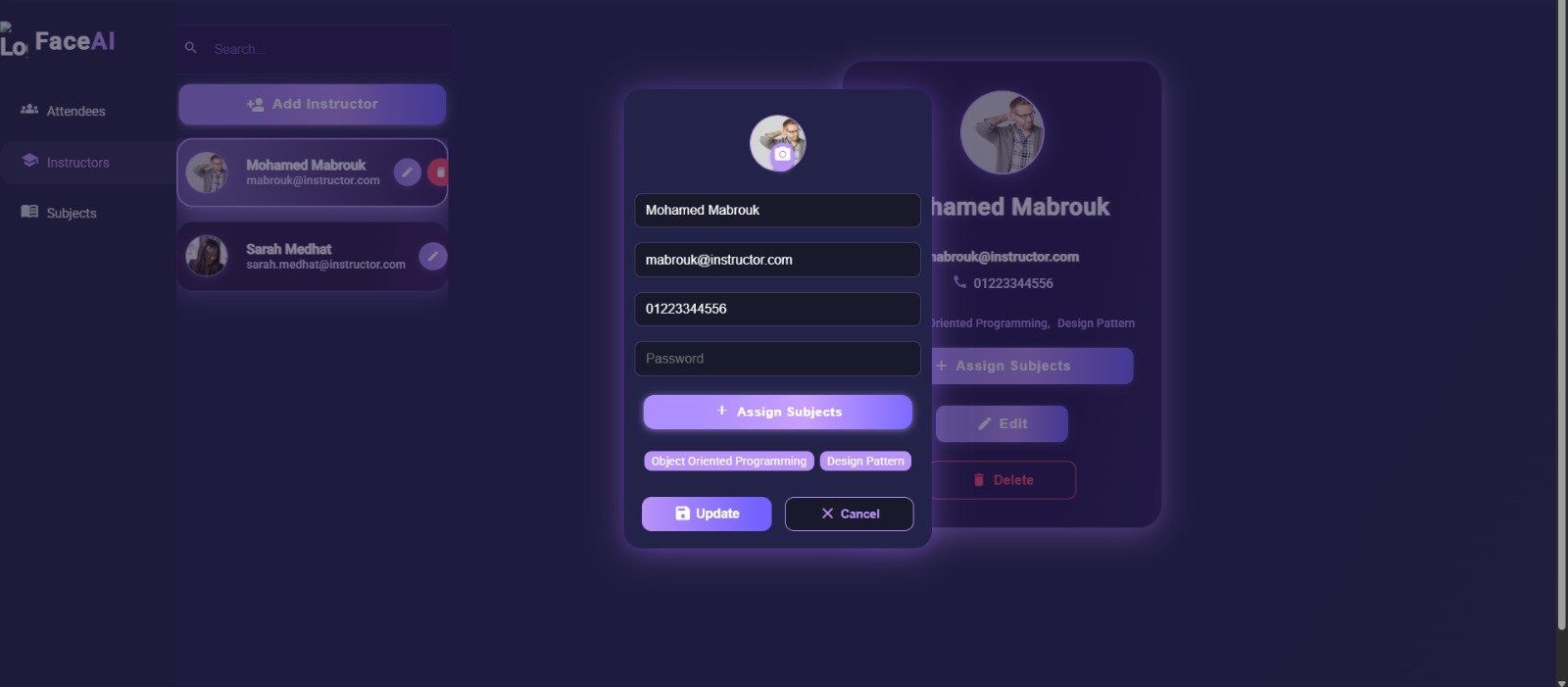








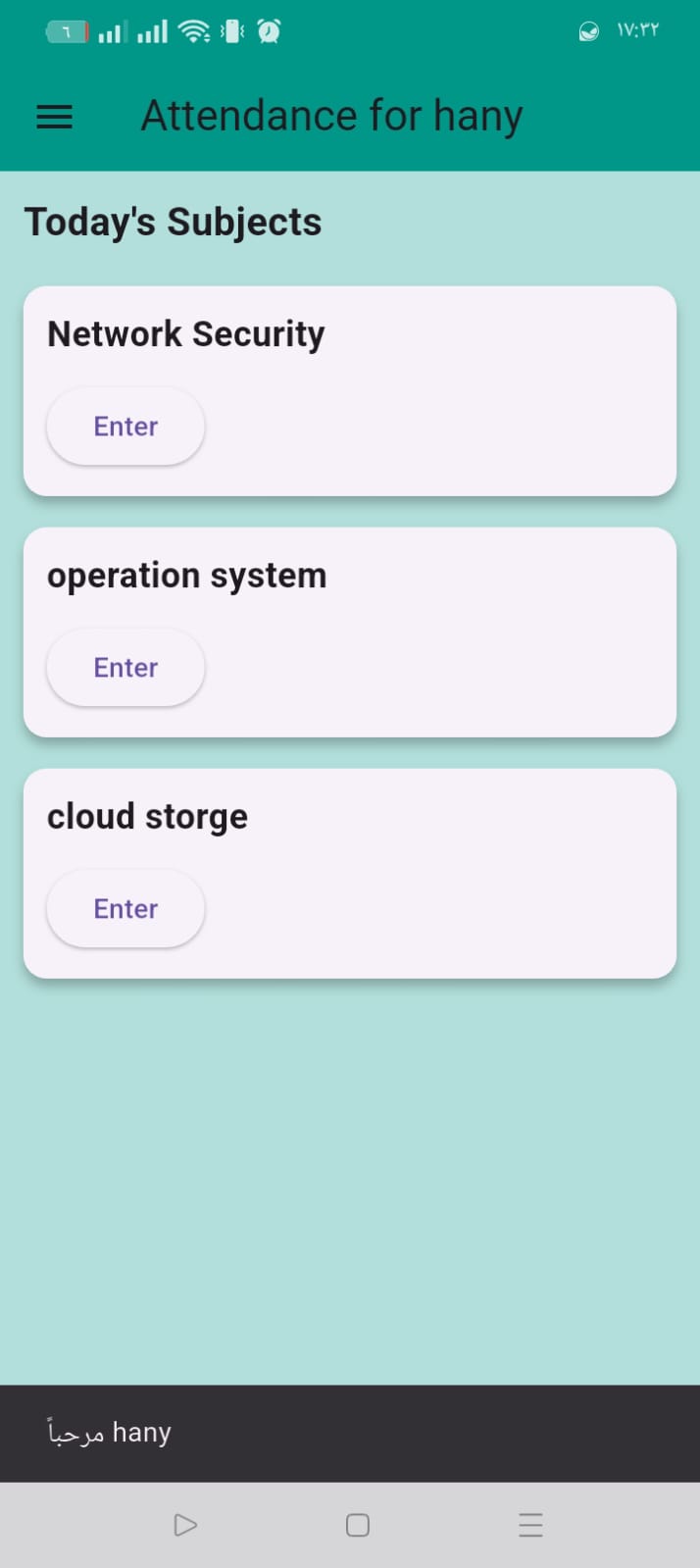




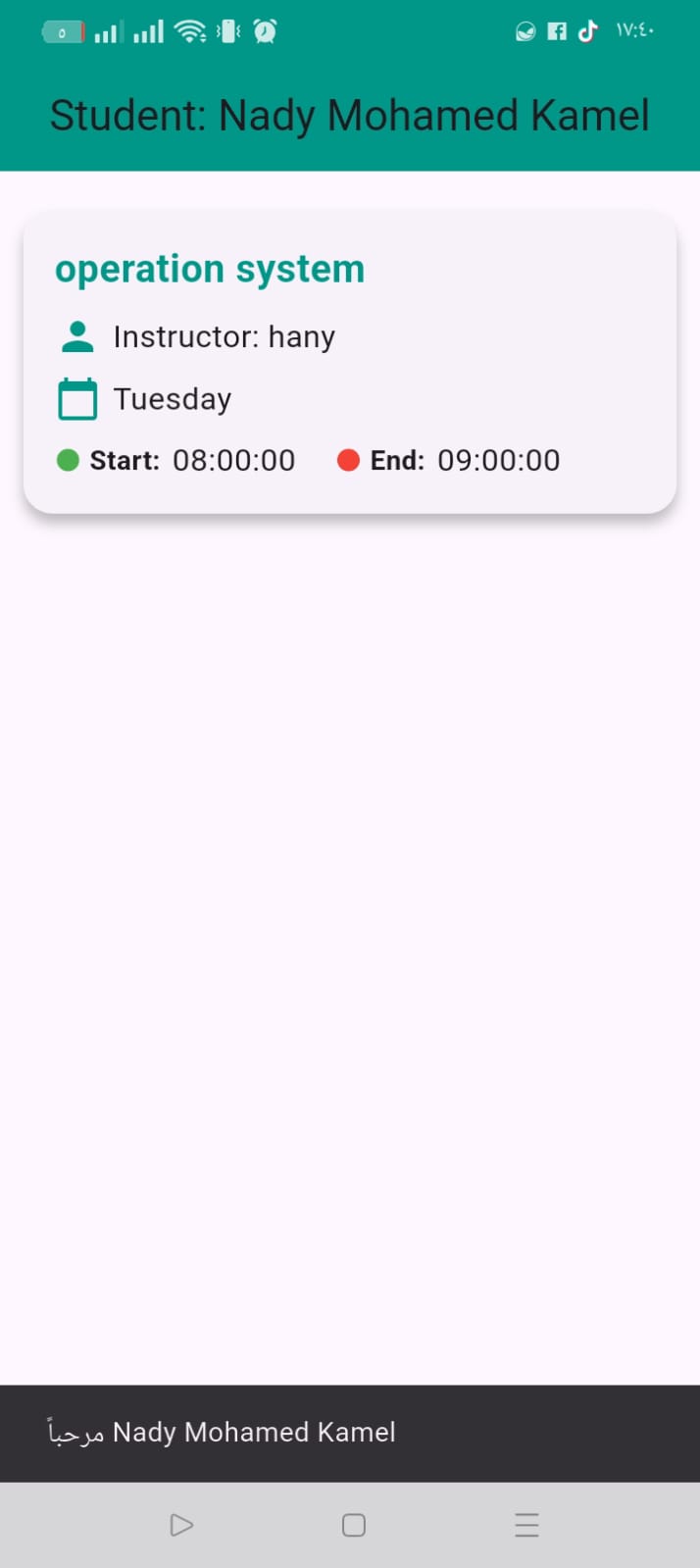
Future Work Design (Moblie Application by flutter)

A screenshot of a login screen

AI-generated content may be incorrect.

A screenshot of a phone

AI-generated content may be incorrect.A screenshot of a phone

AI-generated content may be incorrect.A screenshot of a phone

AI-generated content may be incorrect.