International Journal on Science and Technology (IJSAT)
E-ISSN: 2229-7677 ● Website: www.ijsat.org ● Email: editor@ijsat.org

FARS - Facial Attendance Recognition System

Vadlapally Yashwanth Reddy [1], Kadiyala Tejendra Sai [2], K.Shirisha [3], G.Naga Sujini [4], Dr.K.Rajitha [5], R.Mohan Krishna Ayyappa [6]

[1, 2] Student, Department of Computer Science and Engineering, Mahatma Gandhi Institute of Technology, Gandipet, India.
[3, 4, 5, 6] Assistant Professor, Department of Computer Science and Engineering, Mahatma Gandhi Institute of Technology, Gandipet, India.

Abstract
The Facial Attendance Recognition System using Raspberry Pi is a cutting-edge, AI-powered solution designed to automate attendance tracking through facial recognition technology. Traditional attendance methods, such as manual registers, RFID cards, and biometric fingerprint scanners, often suffer from inefficiencies, security risks, and the potential for proxy attendance. This project aims to address these challenges by providing a contactless, real-time facial recognition system that enhances accuracy, security, and efficiency. The system is developed using a Raspberry Pi 5 (4GB) as the core processing unit, integrated with OpenCV, Dlib, and the Face Recognition API to perform real-time face detection and recognition. A Pi Camera Module or USB camera captures images, while machine learning algorithms match detected faces with pre-registered data. The system automatically logs attendance records into a MySQL database, ensuring secure and reliable data storage. A Flask/Django-based web dashboard allows administrators to manage and review attendance records conveniently. This system offers numerous advantages over traditional methods, including eliminating physical interaction, reducing administrative workload, preventing fraudulent attendance marking, and improving overall efficiency. It is highly scalable and can be deployed across educational institutions, corporate offices, healthcare facilities, and secured access areas. The portability and low-cost nature of the Raspberry Pi make it an ideal choice for widespread adoption.

Keywords: Raspberry Pi, openCV, camera, Facial Recognition, smart campus, Autonomous

1. Introduction

The **Facial Attendance Recognition System** is an automated solution designed to mark attendance using facial recognition technology. This system eliminates the need for traditional attendance methods like manual entry or RFID cards, making the process more efficient, secure, and contactless. It is built using a **Raspberry Pi 5 (4GB)**, integrated with OpenCV and machine learning algorithms for real-time face detection and recognition.

Facial recognition technology has gained significant attention in recent years due to its accuracy, speed, and security benefits. The ability to automate attendance tracking not only reduces administrative workload but also minimizes human errors. By leveraging computer vision and AI, this project aims to create a seamless and reliable solution suitable for various applications, including educational institutions, corporate offices, and secure access systems. The Raspberry Pi, being a low-cost and efficient hardware platform, makes this implementation highly scalable and portable

1.1 Problem Definition

Traditional campus management in educational institutions is often fragmented, relying on separate systems or manual processes for attendance. This fragmentation leads to inefficiencies, increased administrative workload, and a poor user experience for students and staff. Students must manage attendance, increasing in false attendance. These operations are hampered by physical note books and attendance registers. This lacks real-time updates, causing uncertainty for parents and students. Attendance systems are vulnerable to proxy attendance and require manual verification.

1.2 Existing Applications

Traditional attendance systems include manual registers, RFID-based systems, and biometric fingerprint scanners. These systems require physical interaction, which can lead to inefficiencies, human errors, and security concerns. Manual registers are time-consuming and prone to inaccuracies, while RFID and fingerprint-based systems require additional hardware and may face hygiene concerns, especially in a post-pandemic world.

1.3 Proposed Application

The Automated Facial Recognition-based Attendance System addresses traditional attendance challenges by offering a contactless, AI-powered solution focused on accuracy, security, and efficiency. Built using Python with OpenCV for real-time face detection and recognition, and integrated with a secure backend database, the system is designed for scalability, ease of use, and reliability. It captures faces through a live video feed, matches them against pre-registered profiles, and automatically marks attendance without manual intervention. The system ensures secure data handling, prevents duplicate entries, and maintains detailed attendance logs with timestamps. Its modular design allows for easy integration with existing institutional management platforms, offering real-time updates and centralized monitoring. The solution is suitable for educational institutions, offices, and other organizations seeking a modern, efficient alternative to traditional attendance methods.

1.4 Requirements Specification

Requirement Specifications describe the arti-craft of Software Requirements and Hardware Requirements used in this project.

Software Requirements

1. **Operating System:** Raspberry Pi OS
2. **Database:** Firebase Cloud Firestore
3. **Programming Language:** Python 3.8 or later
4. **Backend:** Python (with Flask optional for API, but not required in your current implementation)
5. **Frontend:** Tkinter/ttk (Python GUI); no React.js frontend in your current project
6. **Libraries/Frameworks:**
   a. firebase-admin (for Firebase integration)
   b. Pillow (image processing)
   c. OpenCV (for facial recognition)
   d. datetime, re, threading, smtplib (for various utilities)
7. **Communication Protocols:** HTTPS (for secure Firebase communication)
8. **Security:** Raspberry Pi Camera, facial recognition, and PIN/password for admin access

9. **AI/ML Integration:** face_recognition, dlib (implicitly used via face_recognition)

Hardware Requirements

1. **Processor:** Raspberry Pi 5 (Primary Computational Device)
2. **Memory:** 4GB RAM (minimum), 8GB recommended for smooth operation
3. **Storage:** 30GB free disk space (for application, dependencies, and local data caching)
4. **Camera: USB** or Pi connected Camera of any resolution
5. **Connectivity:** Wi-Fi or Ethernet (for internet access to Firebase)

Power: Standard USB C power supply

2. Literature Survey

"Real-Time Face Recognition on Raspberry Pi Using OpenCV" by T. S. Kamaraj and R. Ramesh
This study explores the implementation of real-time facial recognition on a Raspberry Pi using OpenCV, with a focus on low-cost, efficient deployment. The authors demonstrate how the Raspberry Pi, coupled with a USB camera, can be utilized for face detection and recognition in various applications, including security and attendance systems. The research highlights the feasibility of using Raspberry Pi for real-time face recognition, making it an ideal platform for attendance tracking in resource-constrained environments. For our project, this study emphasizes the effectiveness of Raspberry Pi and OpenCV for facial recognition tasks.

"Face Recognition System for Attendance Monitoring Using Raspberry Pi" by Prashant Choudhary, et al.
This paper proposes an automatic attendance system based on facial recognition using Raspberry Pi and Python libraries, such as OpenCV and dlib. The system captures students' images, processes them for feature extraction, and matches them with stored data to mark attendance automatically. The paper highlights the advantages of a facial recognition system in terms of efficiency, accuracy, and prevention of proxy attendance. This research serves as a relevant model for integrating face recognition with attendance systems, which is central to our project.

"**Smart Attendance System Based on Face Recognition Using Raspberry Pi"** by M. J. Dsouza and S. K. Prasad
The authors present a smart attendance system that integrates face recognition and the Raspberry Pi platform. The system uses a camera module to capture student images, which are processed using the face_recognition library. The system compares the captured face to stored data, and attendance is marked automatically. The study emphasizes the real-time processing capabilities of Raspberry Pi and the potential for remote monitoring via a web interface. For our project, this paper is significant as it demonstrates how facial recognition can be integrated into an attendance system with remote access.

"Face Recognition Based Attendance System: A Review" by Abhishek Tiwari and Pankaj Kumar
This review discusses various face recognition techniques and their application in automated attendance systems. The authors compare different algorithms like Eigenfaces, Fisherfaces, and CNN-based models in terms of accuracy, processing time, and scalability. The study highlights the importance of choosing the right algorithm based on the system's requirements, such as the computational limitations of a

Raspberry Pi. This literature helps inform our selection of the appropriate algorithm for facial recognition in our project.

"Automatic Attendance System Based on Facial Recognition and IoT Using Raspberry Pi" by N. S. V. Prasad and B. R. Anand
This research proposes an IoT-based attendance system that uses facial recognition for marking attendance in classrooms. The system employs a Raspberry Pi to capture facial images and compares them with stored data for verification. It also integrates cloud storage for data backup and remote access via a web interface. The paper demonstrates how IoT and Raspberry Pi can work together to create a seamless attendance tracking system. For our project, this highlights the potential for integrating cloud-based storage and remote access into our facial recognition attendance system.

"The Role of Internet of Things in Smart Education" by Valentina Terzieva, Svetozar Ilchev, Katia Todorova.
This study explores how IoT can transform education by creating smart learning environments. It discusses IoT's role in optimizing educational processes through sensor-based data collection and intelligent communication protocols. The research presents two IoT prototypes designed to enhance smart schools by automating data gathering and information dissemination. For our project, this highlights the integration of IoT for efficient student tracking, automated attendance, and real-time updates in an educational setting.

"A Survey of Face Recognition Algorithms and Their Applications in Biometric Systems" by H. Y. Zhang et al
This survey presents an overview of the most common face recognition algorithms, including traditional methods like PCA and LDA, and deep learning approaches such as CNNs. It also explores various applications of face recognition, with a focus on biometric systems like attendance management. The study addresses challenges such as lighting variations, occlusions, and real-time processing, which are relevant to implementing a facial recognition system on a Raspberry Pi. For our project, this research provides insight into the challenges and possible solutions when using Raspberry Pi for facial recognition.

"Cloud-Based Face Recognition System Using Raspberry Pi" by A. P. Shah and S. J. Bhandari
This paper introduces a cloud-based facial recognition system that utilizes Raspberry Pi to capture and process face data, while sending the data to a cloud server for storage and analysis. The system is designed to automatically mark attendance and send real-time notifications to students and teachers. The study demonstrates the benefits of cloud computing for facial recognition, including data scalability, remote access, and system reliability. For our project, this study emphasizes the importance of cloud integration for future scalability and data management.

3. Methodology

This section outlines the approach taken to design, develop, and implement the FAR system, ensuring a robust, scalable, and user-friendly campus management solution.

3.1 Technologies Used

This section outlines the approach taken to design, develop, and implement the FAR system, ensuring a robust, scalable, and user-friendly campus management solution.

a) **Programming Language:** Python 3.8 or later Python was chosen for its readability, extensive library support, and ease of integration with both hardware (webcam) and cloud services.
b) **GUI Framework:** Tkinter/ttk Tkinter provides a lightweight, native interface for building desktop applications, ensuring cross-platform compatibility and ease of use for both students and administrators.
c) **Database:** Firebase Cloud Firestore Firestore offers a scalable, real-time, cloud-based NoSQL database, enabling secure storage and instant synchronization of student, transaction, library, attendance, and bus activity data.
d) **Image Processing & Face Recognition:** OpenCV, Pillow OpenCV is used for facial recognition in attendance verification, while Pillow handles image processing tasks such as resizing and format conversion.
e) Other Libraries:
   - firebase-admin for Firebase integration
   - datetime , re , threading for utility functions and background processing

3.2 Development Process:

The development of RADIT followed an iterative and modular approach, ensuring flexibility and continuous improvement throughout the project lifecycle:

a) **Requirement Analysis**: Stakeholder needs were gathered and analyzed to define the system's core modules: Facial Recognition, attendance and attendance logs.
b) **System Design:** The architecture was planned with a focus on modularity, allowing each service to function independently yet integrate seamlessly via a shared database and authentication system.
c) Module Implementation:
   - **Facial Recognition Attendance:**
     The Facial Recognition Attendance module is the core component of the system, utilizing the Raspberry Pi's camera and OpenCV to capture student faces and compare them against a pre-existing database.
   - **Database Management:**
     The **Database Management** module is implemented using **Firebase Cloud Firestore**, which stores student profiles, attendance records, and logs of the facial recognition events.
   - **Local Attendance Storage:** Attendance being stored in a local csv file as a redundancy to the database.
d) **Database Integration:** Firebase Firestore collections were structured attendance, ensuring efficient data retrieval and security.
e) **Testing:** Each module underwent unit and integration testing. User acceptance testing was conducted with sample data to validate workflows and user experience.
- **Deployment & Feedback:** The system was deployed in a controlled environment, and feedback was collected from users (students, mentors and project guides) for further refinement.

4. Design of FAR System

The **FAR** system is designed as a **modular, role-based** platform for student authentication, attendance tracking. It follows a **scalable and maintainable** architecture with key components including **database design, modular application structure, user interface, and role-based access control**.

4.1. System Architecture

The system consists of multiple components interacting through **Facial Recognition and a database-driven backend**. It is shown in Figure 4.1.

Key Components:

- **Raspberry Pi:** Computes Everything
- **Database:** Stores student records, and attendance logs.
- **Graphical User Interface (GUI):** Role-based dashboards for different users.

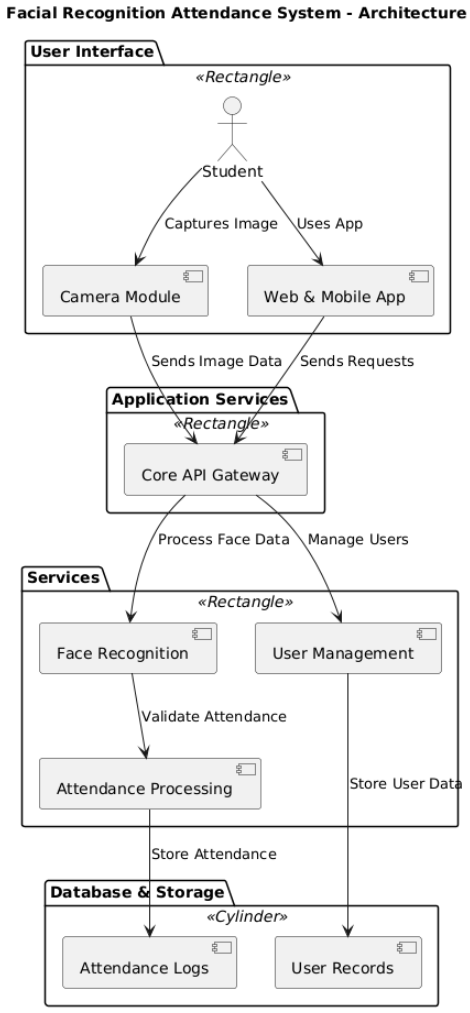


Figure 4.1: System architecture diagram

4.2. Database Design

The system uses an **database** (attendance.db) with the following tables as shown in Figure 4.2:

- **storeAttendance()**→ Stores attendance, time of entry, etc.
- **fetchAttendance()** → Fetches the stored Attendance.

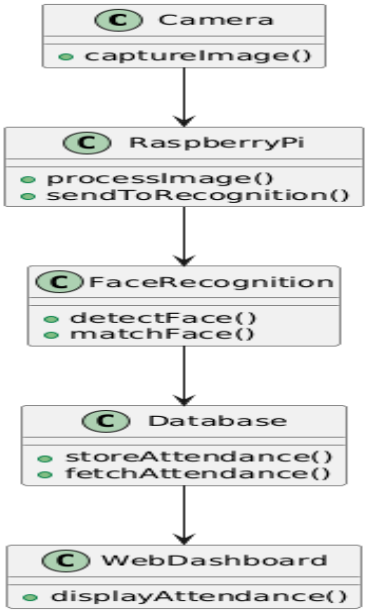


Figure 4.2: ER diagram

4.3. Control Flow showing User Activity

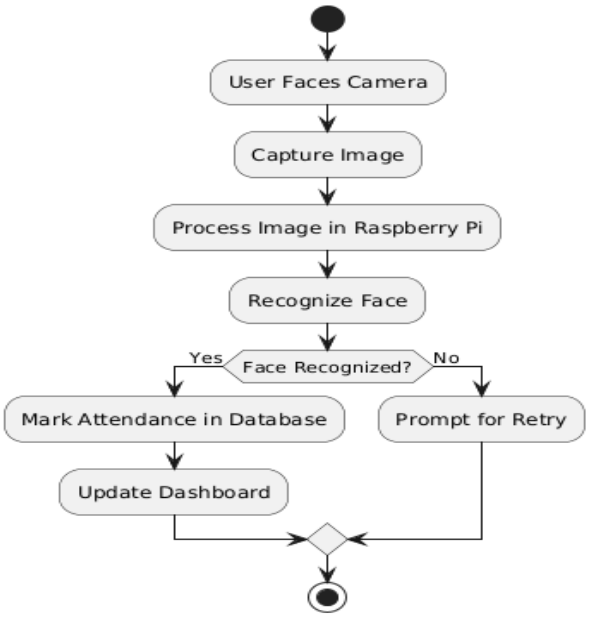


Figure 4.3: Control Flow

The **Control Flow Diagram** shown in Figure 4.3 illustrates the step-by-step execution of system processes such as face recognition and **attendance marking.**

4.4. Modular Architecture & Object-Oriented Structure

The **Class Diagram** below outlines the core components of the **Facial Attendance System**. Each class represents a functional module in the system, capturing key operations that enable facial recognition and attendance logging.

Key Classes & Responsibilities:

- **Camera**: Captures live images from the environment.
  - captureImage(): Captures a snapshot of the user's face.
- **RaspberryPi**: Handles local computation and device integration.
  - processImage(): Preprocesses the image (e.g., resizing, grayscale).
  - sendToRecognition(): Forwards image data to the recognition module.
- **FaceRecognition**: Performs facial analysis and identity verification.
  - detectFace(): Detects the presence of a face in the image.
  - matchFace(): Matches detected face with pre-stored profiles.
- **Database**: Stores and retrieves attendance records.
  - storeAttendance(): Saves attendance log into the system.
  - fetchAttendance(): Retrieves attendance data for display.
- **WebDashboard**: User interface for administrators and teachers.
  - displayAttendance(): Shows attendance data via a web interface.

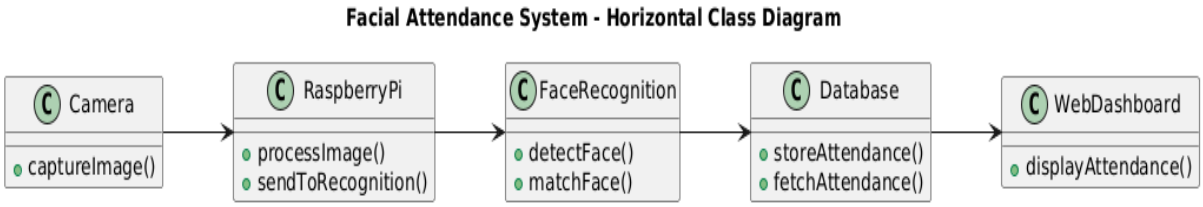


Figure 4.4: Class diagram

4.5. Role-Based Authentication & Access Control

The **Use Case Diagram** defines user interactions within the Facial Attendance System. It illustrates the permissions and activities available to different types of users.

Actors & Use Cases:

- **Admin**
  - **View Attendance**: Reviews attendance records for all users.
  - **Manage Database**: Adds/removes student records and resets logs.
- **Student**
  - **Mark Attendance**: Initiates the attendance process via the face recognition interface.

- **Face Recognition System**: Automatically triggered to identify the student using camera input.

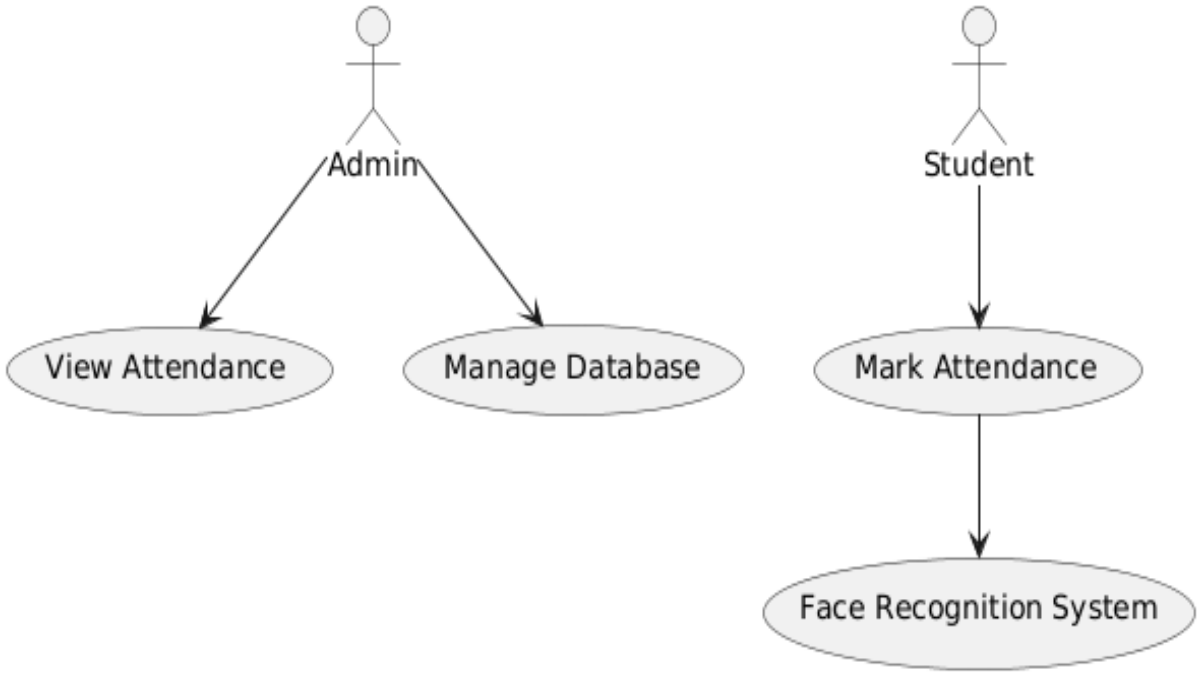


Figure 4.5: Use case diagram

4.6. Student-Based Process Workflow

The **Student Sequence Diagram**, shown in Figure 4.6, illustrates a typical student interaction (e.g., attendance marking, transaction processing).

Example - Attendance Process:

1. Student registers their face.
2. System looks for a face match.
3. System retrieves student details if there is a match.
4. Attendance is logged in the database.

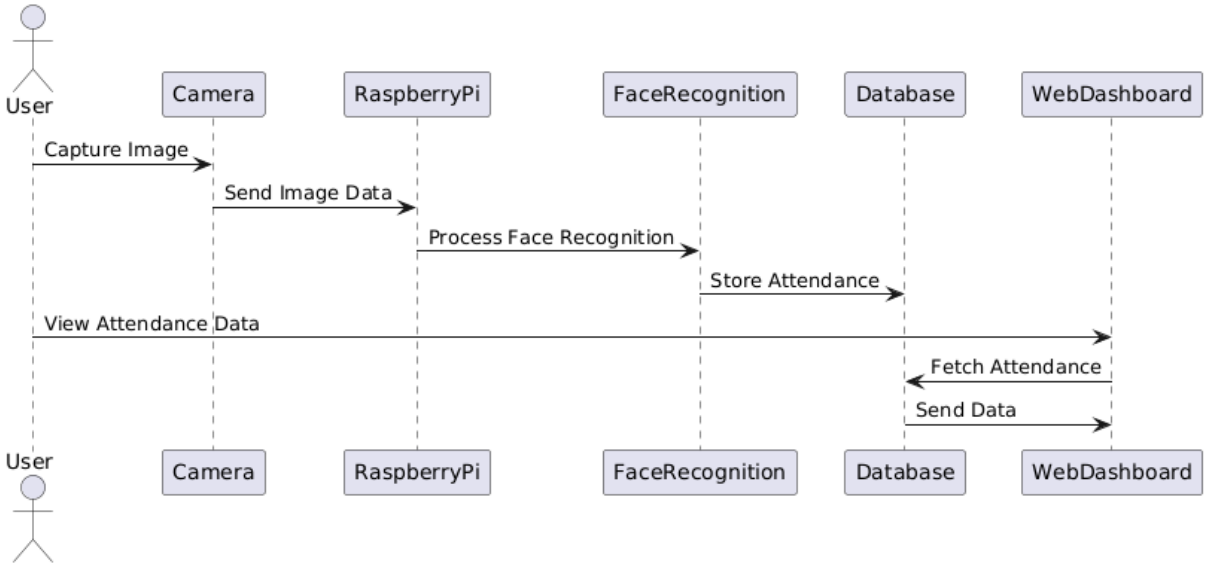


Figure 4.6: Sequence diagram for a student user

5. Implementation

5.1 System Architecture

The **Facial Recognition Attendance System** is architected as a modular Python application running on a Raspberry Pi 5, utilizing the **OpenCV** library for facial recognition and **Firebase Firestore** for cloud-based data storage. The application follows a modular approach, with distinct components managing tasks such as image capture, face recognition, attendance recording, and database management.

The architecture is organized into separate modules:

- **Face Recognition Module** for detecting and verifying student faces.
- **Attendance Module** for logging attendance data.
- **Database Integration Module** for managing student profiles and attendance records.
- **Notification Module** for sending alerts to administrators or parents in case of discrepancies or successful attendance logging.

These modules interact through an event-driven design, where GUI actions or hardware events trigger backend logic (such as facial recognition or database updates). This structure ensures real-time synchronization of attendance data and seamless integration between hardware and cloud-based systems.

5.2 Database Integration

The system uses **Firebase Firestore** as the cloud database, enabling secure and scalable storage for all attendance and student-related data. The **firebase-admin SDK** is used to authenticate and interact with Firestore, ensuring real-time synchronization between the local application and cloud database.

The database structure includes collections such as:

- **students**: Stores student profiles (name, student ID, face encodings).
- **attendance**: Records attendance entries with timestamps, including the student ID and status (present/absent).

- **logs**: Captures all facial recognition events and errors.

CRUD operations are abstracted into utility functions within the utils.py file to maintain consistency across the application. For example, when a student's face is recognized, the attendance collection is updated in real-time. Firebase Firestore's security rules ensure that data is protected and accessible only to authorized users (e.g., admins).

5.3 Camera Integration

The **Face Recognition Module** is implemented using **OpenCV** and the **face_recognition** library. The system uses a webcam to capture live images, which are then processed to extract facial features and generate face encodings. These encodings are stored in the Firebase database during the student registration process.

At the time of attendance, a live image is captured, and its encoding is compared to the stored encodings in the database. If a match is found, the system marks the student as present and updates the attendance record. This ensures accurate attendance logging and prevents proxy attendance. Facial recognition acts as a second layer of authentication for attendance marking, making the process both secure and efficient.

5.4 User Interface

The **User Interface (UI)** for the facial recognition attendance system is built using **Tkinter**, providing a simple yet effective platform for administrators to manage and monitor attendance data.

- **Login and Authentication**: The login screen supports PIN/password entry for administrators. Students do not need to log in manually, as their faces are automatically recognized during attendance marking.
- **Dashboard**: After authentication, administrators are presented with a dashboard displaying real-time attendance data, student profiles, and logs of facial recognition events.
- **Attendance Module**: Administrators can view, edit, and download attendance reports. The system provides clear indicators (e.g., color-coded attendance statuses) to highlight discrepancies.
- **Role-Based Access**: The UI is designed to cater to different user roles, with admins having full control over the system and viewing capabilities, while students only interact with the attendance process (via face recognition).

The UI ensures intuitive navigation and quick access to essential functionalities, improving usability and user experience.

5.6 Security

Security is a key aspect of the system, and several measures are implemented:

- **Face Recognition**: Ensures that only authorized students are marked present. Facial data is stored securely in the cloud and used for matching during attendance.
- **PIN/Password for Admin Access**: Admin access to the system is protected with a PIN or password, ensuring that only authorized personnel can manage sensitive data.

- **Data Encryption**: All communications with Firebase are secured using HTTPS, and sensitive data (such as face encodings) is encrypted before being stored in the database.
- **Role-Based Access Control**: Admins and students have different access levels, ensuring that only admins can modify attendance records or view detailed logs.

5.7 Error Handling and Reliability

- **Hardware Failures**: If the camera or any connected hardware fails, the system logs the error and provides user-friendly messages for troubleshooting.
- **Network Connectivity**: In case of network issues, the system logs attendance locally and synchronizes data with Firebase once the connection is restored.
- **Invalid User Actions**: If the system fails to recognize a student's face, it prompts the admin to manually mark attendance or address the issue.

The modular design of the system ensures that each component (face recognition, database integration, etc.) can be independently developed and debugged, increasing overall system reliability.

6. Testing and Results

As shown in Figure 6.1, the main interface displays all FAR modules in a centralized dashboard, providing access to Face registration and attendance log.

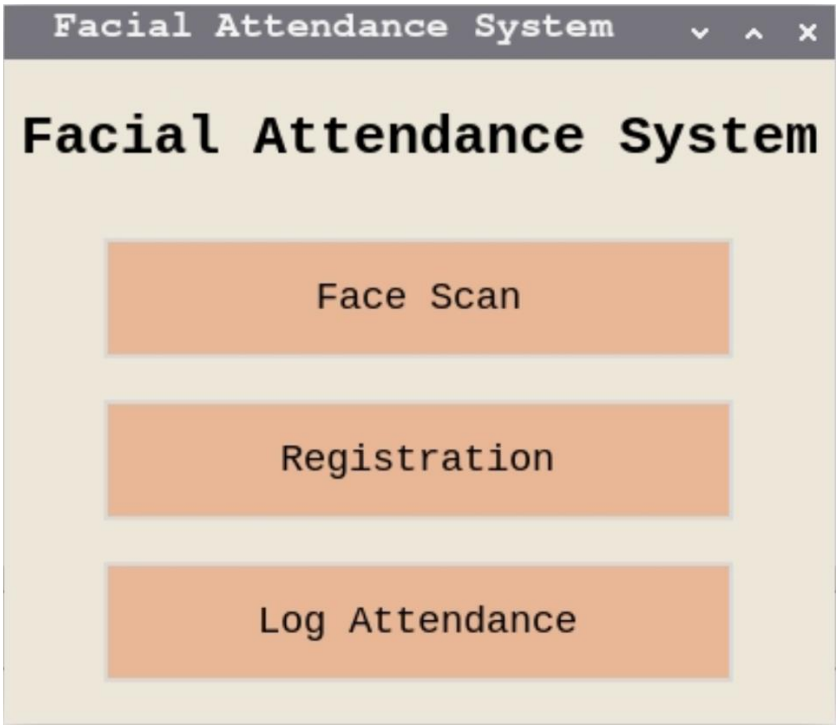
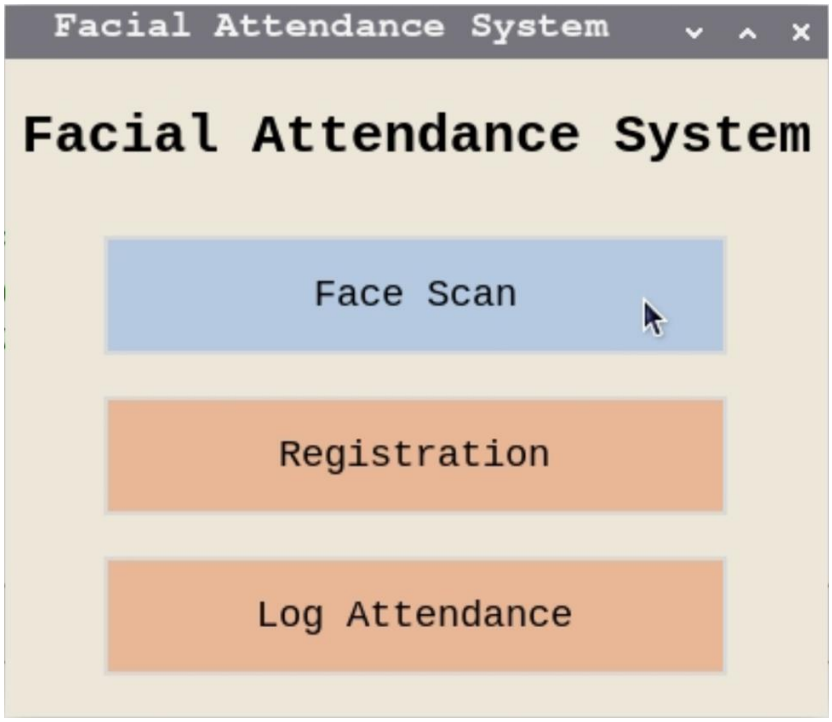


Figure 6.1: The main menu of the application

Figure 6.2 shows the panel that opens when you choose the face scan option from the home menu showing the camera feed used to capture the images.



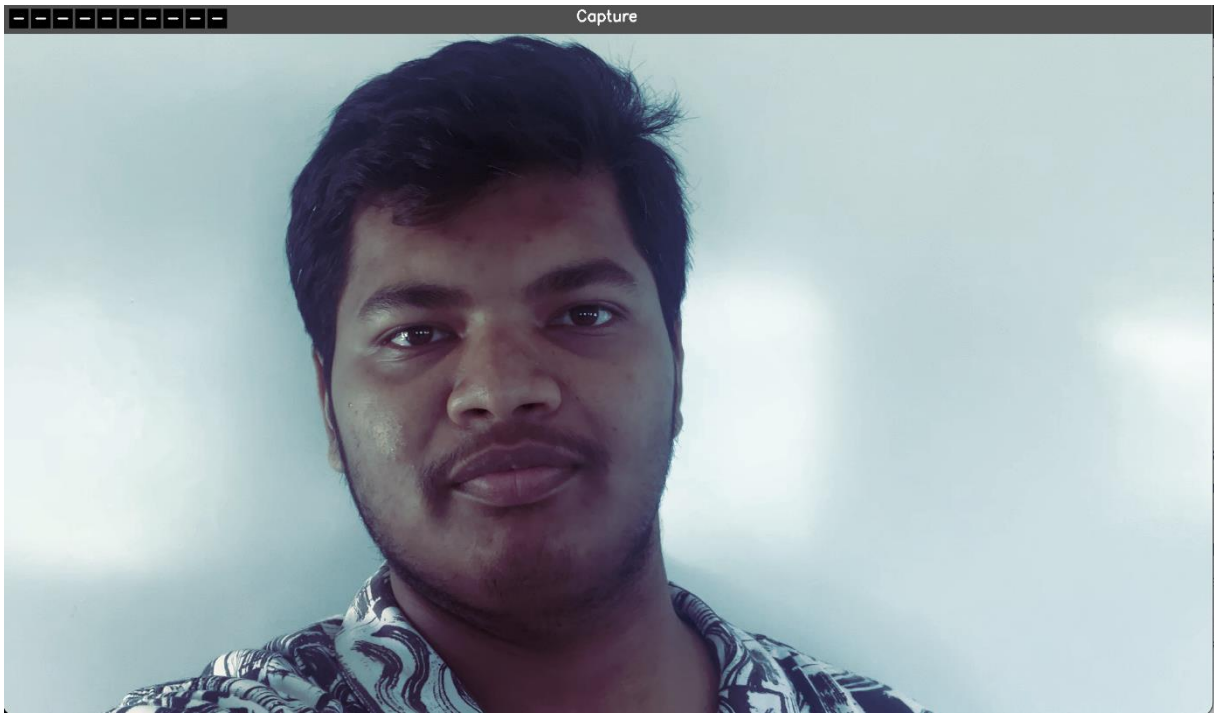
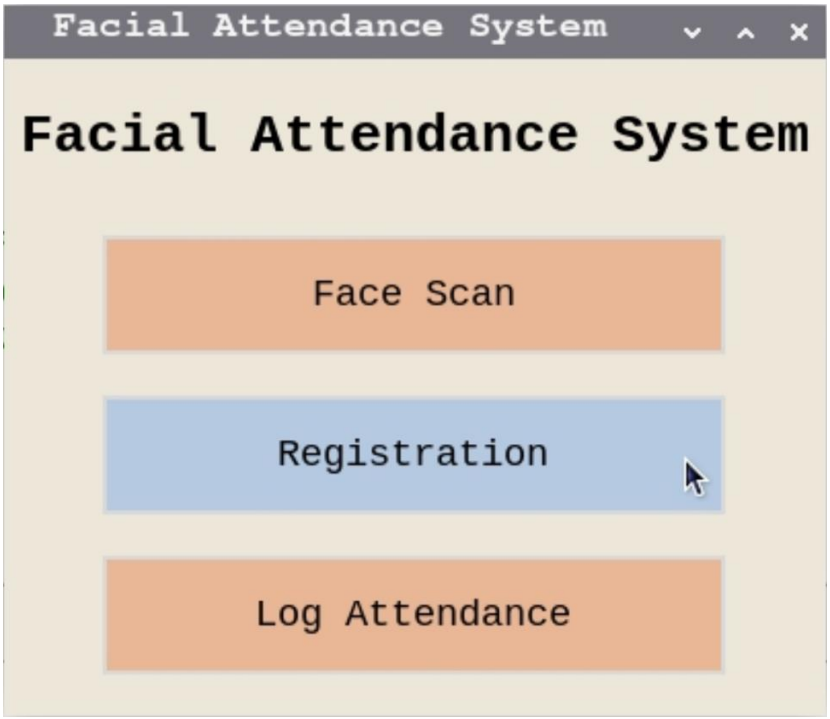


Figure 6.2: The Face Scan option

As shown in Figure 6.3, we can see the option of REGISTRATION which trains the model with the images taken using the camera previously.




```
[INFO] start processing faces...
[INFO] processing image 1/79
[INFO] processing image 2/79
[INFO] processing image 3/79
```

```
[INFO] processing image 71/79
[INFO] processing image 72/79
[INFO] processing image 73/79
[INFO] processing image 74/79
[INFO] processing image 75/79
[INFO] processing image 76/79
[INFO] processing image 77/79
[INFO] processing image 78/79
[INFO] processing image 79/79
[INFO] serializing encodings...
[INFO] Training complete. Encodings saved to 'encodings.pickle'
```

Figure 6.3: The Registration option.

Figure 6.4 Demonstrates the Log attendance option which we can use to mark the attendance which therefore logs the attendance based on the facial recognition in a database.

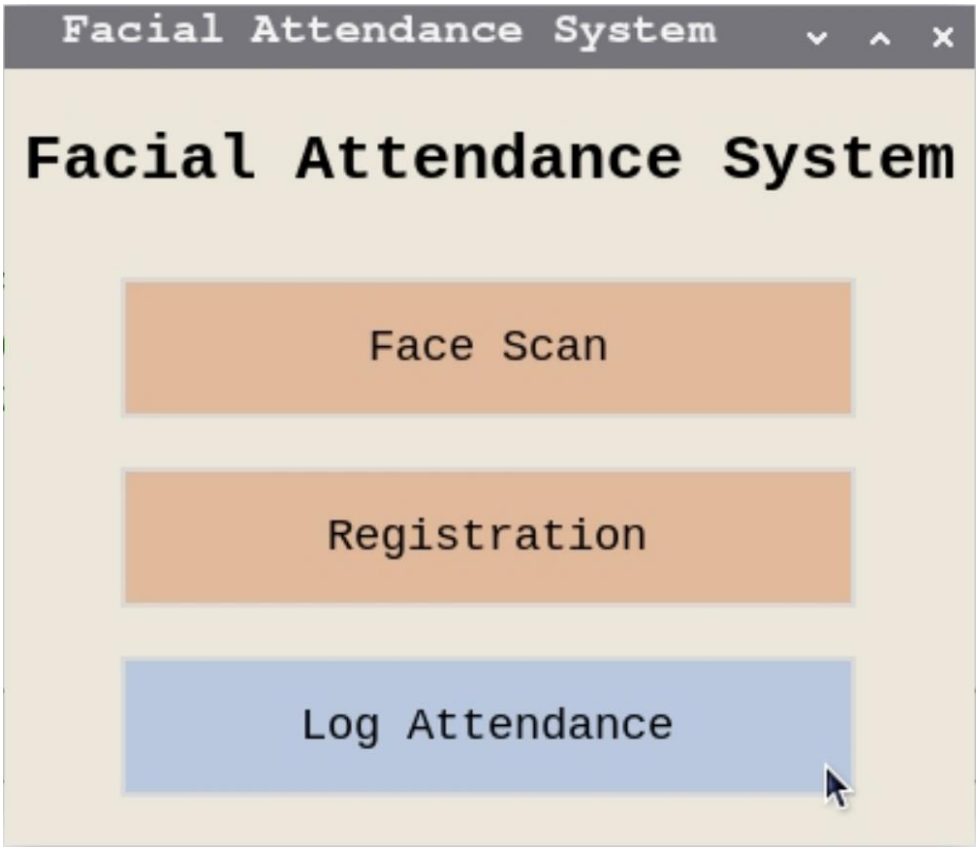


Figure 6.4: The Log attendance system

Figure 6.5 Demonstrates Students face being recognized by the FAR System and Logs the Attendance into the Database.

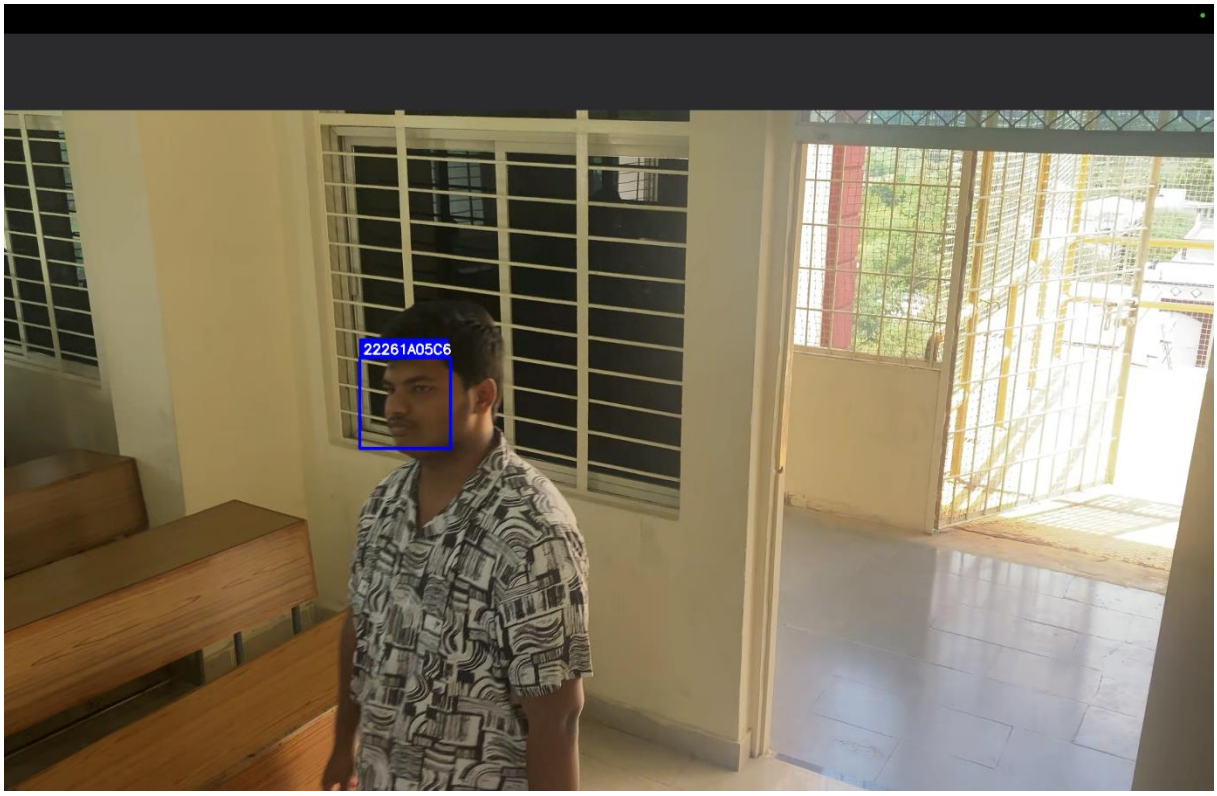


Figure 6.5: Face Recognition in classroom done handsfree

As shown in Figure 6.6, the csv file shows the attendance logs of students entering the class.

Name	Time	
22261A05C6	12:07:58	
22261A0568	12:08:07	
22261A0583	12:16:41	

Figure 6.6: The Database.

7. Conclusion and Future Scope

7.1 Conclusion

The **Facial Attendance Recognition System** powered by **Raspberry Pi** offers an efficient, automated, and secure way to track attendance. By leveraging **AI and computer vision**, it enhances accuracy and convenience, making it a viable solution for various institutions and workplaces. With future enhancements such as cloud integration and mobile accessibility.

Additionally, the system's scalability makes it an ideal solution for organizations of varying sizes. Its capacity for real-time processing ensures that attendance is marked instantly, and with the potential for

cloud integration, this solution can be accessed remotely, providing a flexible and modern approach to tracking attendance.

In conclusion, the Facial Attendance Recognition System has the potential to reshape the future of attendance management, providing a smarter, more reliable solution that could easily replace traditional methods and be adopted by a wide variety of institutions and businesses.

7.2 Future Scope

- **Mobile Application:** Develop native or cross-platform mobile apps for students, parents, and staff to access FAR system services on the go.
- **Web-Based Interface:** Extend the system with a modern web frontend for broader accessibility and easier administration.
- **Offline Functionality:** Implement offline data caching and synchronization to ensure uninterrupted service during network outages.
- **Third-Party Integrations:** Enable integration with other educational platforms, payment gateways, and transportation systems.
- **Accessibility Enhancements:** Improve the user interface for better accessibility, including support for users with disabilities.
- **Automated Notifications:** Expand notification channels (SMS, push notifications) for real-time alerts to students and parents.
- **Customizable Modules:** Allow institutions to enable or disable modules based on their specific needs, making RADIT adaptable to various educational environments.

References

1. **Face Recognition Based Attendance Management System Using Raspberry Pi (October 2019)**
   https://www.researchgate.net/publication/336824776_Face_Recognition_Based_Attendance_Management_System_Using_Raspberry_Pi
2. **Development of a Portable Device for Students' Attendance Marking Based on Facial Recognition**
   (December 2022)
   https://www.researchgate.net/publication/380719388_Development_of_a_Portable_Device_for_Students'_Attendance_Marking_Based_on_Facial_Recognition
3. **Automatic Class Attendance System Using Biometric Facial Recognition Technique Based on Raspberry Pi (September 2023)**
   https://www.researchgate.net/publication/374438873_Automatic_Class_Attendance_System_Using_Biometric_Facial_Recognition_Technique_Based_on_Raspberry_Pi
4. **Raspberry Pi Foundation. (2024).** Official Raspberry Pi Documentation. Retrieved from **https://www.raspberrypi.org/documentation**
5. **OpenCV Library**
   Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools.
   https://opencv.org/
6. **Face Recognition Python Library**

Geitgey, A. (2017). face_recognition: Simple face recognition library built with deep learning. **https://github.com/ageitgey/face_recognition**

7. **Tkinter GUI Toolkit**
   Lundh, F. (2001). An Introduction to Tkinter. PythonWare.
   https://docs.python.org/3/library/tkinter.html