

**EFFICIENT ATTENDANCE TRACKING
USING ON_SPOT FACIAL RECOGNITION SYSTEM**

A PROJECT REPORT

Submitted by

**NANDHAPRABHU R
(Reg.No: 24MCR072)**

**SELVA SAHAYAM JENSITON S
(Reg.No: 24MCR096)**

**THANUSRI M V
(Reg.No: 24MCR115)**

in partial fulfillment of the requirements for the

award of the degree

of

**MASTER OF COMPUTER APPLICATIONS
DEPARTMENT OF COMPUTER APPLICATIONS**



KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI, ERODE – 638 060

DECEMBER 2024

DEPARTMENT OF COMPUTER APPLICATIONS**KONGU ENGINEERING COLLEGE****(Autonomous)****PERUNDURAI, ERODE – 638 060****DECEMBER 2024****BONAFIDE CERTIFICATE**

This is to certify that the project report entitled “**EFFICIENT ATTENDANCE TRACKING USING ON_SPOT FACIAL RECOGNITION SYSTEM**” is the bonafide record of project work done by, **NANDHAPRABHU R (24MCR072)**, **SELVA SAHAYAM JENSITON S (24MCR096)** and **THANUSRI M V (24MCR115)** in partial fulfilment of the requirements for the award of the Degree of Master of Computer Applications of Anna University, Chennai during the year 2024-2025.

SUPERVISOR**HEAD OF THE DEPARTMENT****(Signature with seal)****Date:**

Submitted for the End Semester Viva Voce Examination held on _____

INTERNAL EXAMINER**EXTERNAL EXAMINER**

DECLARATION

We affirm that the project title **EFFICIENT ATTENDANCE TRACKING USING ON_SPOT FACIAL RECOGNITION SYSTEM** being submitted in partial fulfilment for the award of **Master of Computer Applications** is the original work carried out by us. It has not formed the part of any other project submitted for award of any degree, either in this or any other University.

Date:

NANDHAPRABHU R
(REG.NO:24MCR072)

SELVA SAHAYAM JENSITON S
(REG.NO:24MCR115)

THANUSRI M V
(REG.NO:24MCR96)

I certify that the declaration made by the above candidates is true to the best of my knowledge.

Date:

Name and Signature of the Supervisor
(Ms.S.HEMALATHA)

ABSTRACT

Traditional attendance management systems, such as manual roll calls and card-based methods, are inefficient and fraught with challenges. Manual roll calls consume valuable instructional time, especially in large classrooms, and are prone to human error. Card-based systems, while slightly more automated, are susceptible to misuse through proxy attendance or lost/stolen cards. These methods lack integration with modern technologies, limiting real-time reporting and secure data management. Furthermore, the administrative burden of managing attendance records manually often leads to delays and inaccuracies in maintaining institutional records, making these systems unsuitable for modern educational needs.

The proposed system, "Efficient Attendance Tracking Using On-Spot Facial Recognition," addresses these issues by introducing a mobile-based platform powered by advanced facial recognition technology. Teachers capture real-time images of students during class, and the system automatically processes these images to identify and authenticate students by matching them with pre-stored profiles in a secure database. This process eliminates the need for physical interaction or manual effort. The system operates in both online and offline modes, ensuring reliability even in areas with poor or intermittent connectivity. Offline attendance data is seamlessly synced with the central database when connectivity is available. The platform also incorporates features like instant notifications, real-time verification, and secure cloud storage for attendance data, providing a comprehensive and modern solution for attendance tracking.

This solution significantly improves efficiency, accuracy, and security in attendance tracking by leveraging advanced image processing and deep learning technologies. It eliminates manual errors, mitigates proxy attendance risks, and provides real-time updates and secure data storage. The system's user-friendly interface and scalability make it a robust tool for educational institutions, ensuring seamless attendance management in diverse operational scenarios. The system works both online and offline, ensuring reliability even with intermittent connectivity. Additional features include real-time verification, instant notifications, and secure cloud storage for data management.

ACKNOWLEDGEMENT

We express our sincere thanks to our beloved Correspondent **Thiru.A.K. ILANGO B.Com., M.B.A., LLB.**, and other philanthropic trust members of Kongu Vellalar Institute of Technology Trust for having provided with necessary resources to complete this project. We are always grateful to our beloved visionary Principal **Dr.V.BALUSAMY BE(Hons)., MTech., PhD** and thank him for his motivation and moral support.

We convey our gratitude and heartfelt thanks to our Head of the Department **Dr.A.TAMILARASI M.Sc., M.Phil., Ph.D., M.Tech.**, Department of Computer Applications, Kongu Engineering College for her perfect guidance and support that made this work to be completed successfully.

We also like to express our gratitude and sincere thanks to our Project Coordinator and Supervisor **Ms.S.HEMALATHA MCA**, Assistant Professor(Sr.G), Department of Computer Applications, Kongu Engineering college who have motivated us in all aspects for completing the project in scheduled time.

We owe a great deal of gratitude to our parents for helping us to overwhelm in all proceedings. We bow our heart and head with heartfelt thanks to all those who thought us their warm services to succeed and achieve our work.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No.
	ABSTRACT	iv
	LIST OF FIGURES	ix
	LIST OF ABBREVIATIONS	xi
1.	INTRODUCTION	1
	1.1 PROBLEM DEFINITION	1
	1.2 OBJECTIVE OF THE PROJECT	1
2.	SYSTEM ANALYSIS	3
	2.1 EXISTING SYSTEM	3
	2.1.1 Drawbacks of Existing System	3
	2.2 PROPOSED SYSTEM	4
	2.2.1 Advantages of Proposed System	4
	2.3 FEASIBILITY STUDY	4
	2.3.1 Operational Feasibility	5
	2.3.2 Technical Feasibility	6
	2.3.3 Economical Feasibility	6
3.	SYSTEM SPECIFICATION	8
	3.1 SOFTWARE REQUIREMENT	8
	3.2 HARDWARE REQUIREMENT	8

3.3 SOFTWARE DESCRIPTION	9
3.3.1 Front End	9
3.3.2 Back End	9
3.3.3 Database	11
4. SYSTEM DESIGN	12
4.1 MODULE DESCRIPTION	12
4.2 Registration Module	12
4.3 Camera Module	13
4.4 Attendance Status Module	14
4.5 Profile Module	16
4.2 DATAFLOW DIAGRAM	17
4.3 DATABASE DESIGN	19
4.4 INPUT DESIGN	20
4.5 OUTPUT DESIGN	22
5. SYSTEM TESTING	24
5.1 UNIT TESTING	25
5.2 VALIDATION TESTING	27
5.3 INTEGRATION TESTING	29
6. SYSTEM IMPLEMENTATION	32
6.1 ERROR HANDLING	33

6.2 TESTING	33
6.3 INSTALLATION	33
6.4 DOCUMENTATION	34
6.5 TRAINING	34
6.6 SUPPORT	34
7. CONCLUSION AND FUTURE ENHANCEMENT	35
7.1 CONCLUSION	35
7.2 FUTURE ENHANCEMENTS	35
APPENDIX-A SAMPLE CODING	36
APPENDIX-B SCREENSHOTS	48
REFERENCES	52

LIST OF FIGURES

FIGURE No.	TITLE	PAGE No.
4.1	Registration Module	13
4.2	Camera Module	14
4.3	Attendance Status Module	15
4.4	Profile Module	16
4.5	Data flow diagram level 0	17
4.6	Data flow diagram level 1	18
4.7	Database design	19
4.8	Registration Page	20
4.9	Sign Up Page	21
4.10	Attendance Status Page	22
B.1	Registration Page For User	48
B.2	Registration Page For Students & Teachers	49
B.3	Camera Page	49
B.4	Attendance Report	50

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
RFID	Radio Frequency Identification.
DFD	Data Flow Diagram.
API	Application Programming Interface

CHAPTER 1

INTRODUCTION

1. PROBLEM DEFINITION

Traditional attendance systems in educational institutions face several challenges in effectively managing student attendance. One major issue is the reliance on manual roll calls or basic electronic methods such as RFID cards or biometric devices, which are time-consuming and prone to inaccuracies. These methods also suffer from the risk of proxy attendance, leading to compromised data reliability.

Another challenge is the administrative burden on teachers, especially in large classrooms, where taking and maintaining accurate attendance records becomes cumbersome. Additionally, existing systems often lack flexibility and depend on continuous internet connectivity, limiting their usability in offline environments.

In short, the current systems are inefficient, error-prone, and fail to provide seamless integration with modern technology. A more advanced solution is required to ensure accuracy, save time, and streamline the attendance process, addressing the needs of both educators and students effectively.

2. OBJECTIVE OF THE PROJECT

The primary objective of the Efficient Attendance Tracking Using On-Spot Facial Recognition System is to create a robust and automated platform that simplifies and enhances the process of student attendance management in educational institutions. This project aims to eliminate traditional inefficiencies such as manual roll calls, proxy attendance, and cumbersome record-keeping, ensuring accurate and reliable attendance data. By leveraging advanced facial recognition technology and real-time image verification, the system seeks to reduce manual effort, improve accuracy, and provide a seamless experience for educators and students. Additionally, the platform is designed to operate both online and offline ensuring uninterrupted functionality and scalability for diverse educational environment

CHAPTER 2

SYSTEM ANALYSIS

1. EXISTING SYSTEM

There are various student attendance systems in use today, ranging from manual roll-call methods to automated solutions such as RFID cards, barcodes, and biometric systems. Manual methods require teachers to call out names or pass around attendance sheets, which are time-consuming and prone to errors or proxy attendance. Automated systems, while more efficient, have their limitations. RFID cards and barcodes can be misplaced or exchanged, and biometric systems like fingerprint scanners are costly to set up and maintain. Online attendance systems for remote learning often rely on login tracking, which does not guarantee active participation. These existing systems lack the efficiency, accuracy, and flexibility needed for seamless attendance management.

1. Drawbacks of Existing System

Manual attendance management methods are inherently prone to errors, inefficiencies, and inconsistencies, which often lead to inaccuracies in maintaining attendance records. These traditional systems lack real-time verification capabilities, making them highly susceptible to proxy attendance and other forms of manipulation. Biometric solutions, although more secure, present their own challenges as they are expensive to install, operate, and maintain, making them inaccessible for many educational institutions with budget constraints. Similarly, systems that rely on tools like RFID cards are prone to reliability issues due to lost, damaged, or exchanged cards, further undermining their effectiveness. Moreover, many existing attendance systems are highly dependent on stable internet connectivity, limiting their functionality in regions with poor or intermittent network access. This lack of analytical capabilities hampers informed decision-making and limits opportunities for process improvements, further highlighting the inadequacies of these conventional methods.

2. PROPOSED SYSTEM

The Efficient Attendance Tracking System Using On-Spot Facial Recognition addresses limitations of traditional attendance methods by introducing advanced features for improved accuracy and efficiency. Utilizing real-time facial recognition, it eliminates manual roll-call inefficiencies and proxy attendance risks.

Designed for reliability, the system supports offline functionality, syncing data automatically when connectivity is restored. With a user-friendly interface, minimal training is required for teachers and administrators. Features like encrypted data storage, real-time notifications, automated reports, and leave management enhance security and transparency.

Optimized for mobile use, the system enables teachers to mark attendance via smartphones with live photo capture. Machine learning ensures continuous improvement in recognition accuracy, making the solution scalable and adaptable for institutions of all sizes. This comprehensive system modernizes attendance management for educational environments.

1. Advantages of Proposed System

The proposed Efficient Attendance Tracking System Using On-Spot Facial Recognition automates attendance tracking, improving accuracy and eliminating manual roll calls and proxy attendance, thus saving time and reducing errors. The mobile platform offers a user-friendly interface for teachers to mark attendance using live photo capture, ensuring a seamless and efficient experience. With both online and offline functionality, it ensures uninterrupted operation even in areas with limited connectivity, while automated reports and real-time data synchronization simplify administrative tasks and record-keeping.

Scalable to institutions of various sizes, the system adapts easily to diverse needs while safeguarding sensitive student information with advanced security measures. By leveraging facial recognition technology, the solution modernizes attendance tracking, offering a secure, efficient, and reliable system that enhances teacher productivity and institutional accountability.

3. FEASIBILITY STUDY

A feasibility study for the Efficient Attendance Tracking System Using On-Spot Facial Recognition assesses its practicality and viability by examining technical, operational, and economic aspects. This ensures that the system is implementable, efficient, and cost-effective for educational institutions. Here are the key components that would be part of a feasibility study for an Efficient Attendance Tracking System Using On-Spot Facial Recognition:

- Operational Feasibility
- Technical Feasibility
- Economic Feasibility

1. Operational Feasibility

The operational feasibility assesses how well the system integrates into the existing educational environment and addresses current challenges. The system offers a simple, user-friendly interface for teachers to mark attendance using live photo capture, reducing their workload and saving class time.

The online and offline functionality ensures operational reliability even in areas with limited internet connectivity, allowing data to sync automatically once the connection is restored. Institutions can easily adapt the system without significant changes to existing processes, making it practical for day-to-day use.

The proposed system meets key operational requirements such as reliability, accuracy, and security. Features like real-time notifications and automated reports ensure administrators and teachers have easy access to critical attendance data.

The system's technical feasibility ensures smooth integration with existing infrastructure, supporting both cloud and local storage options. Its scalable architecture and real-time data synchronization improve efficiency while reducing human error.

Financially, the system offers a cost-effective solution with low maintenance and minimal upfront investment and its modular design ensures institutions.

2.3.2 Technical Feasibility

6

The proposed system relies on readily available technologies such as facial recognition, image processing, and mobile platforms, making it technically feasible. The system leverages smartphone cameras for image capture, open-source libraries for facial recognition, and cloud-based or local databases for attendance records.

Key components such as data synchronization (online/offline modes) and real-time processing are supported by modern mobile devices, ensuring smooth implementation. The hardware requirements are minimal, as the system can run on standard smartphones and laptops, reducing dependency on expensive equipment.

The software design incorporates modular and scalable architecture, allowing future upgrades. Open-source tools and frameworks, such as Python and TensorFlow for facial recognition, further enhance feasibility while minimizing costs.

2.3.3 Economical Feasibility

The system is economically viable as it significantly reduces the costs associated with traditional attendance methods, such as paper registers, manual labor, and the maintenance of hardware like RFID scanners or biometric devices.

By utilizing existing devices like smartphones and open-source software, the implementation cost is minimized. Maintenance expenses are also low due to the simplicity of the system's design and its reliance on cost-effective tools.

The long-term economic benefits include improved efficiency, reduced administrative workload, and accurate data that enhances decision-making processes. The system's scalability allows institutions to adopt it incrementally, ensuring affordability and flexibility for various budgets.

SUMMARY

The project introduces an automated attendance system using facial recognition to eliminate errors and proxy attendance. It works both online and offline, offering real-time notifications and secure data storage. Scalable and cost-effective, it utilizes smartphone technology and open-source tools for institutions.

CHAPTER 3

SYSTEM SPECIFICATION

3.1 SOFTWARE REQUIREMENT

Operating System	: Windows 11 and above
Tools	: Android Studio
Frontend	: Flutter
Backend	: Python, Fastapi
Database	: Firebase

3.2 HARDWARE REQUIREMENT

Processor	: Quad-core CPU (ARM based)
RAM	: 4 GB RAM
Hard disk	: 1 TB
Keyboard	: ACCUTYPE keyboard
Mouse	: Optical Mouse

3. SOFTWARE DESCRIPTION

1. Front End

Flutter

Flutter is an open-source UI software development toolkit created by Google, designed for building natively compiled applications for mobile, web, and desktop from a single codebase. It utilizes the Dart programming language, which allows developers to create highly interactive and visually appealing user interfaces with ease. One of the key advantages of Flutter is its rich set of pre-designed widgets that follow specific design languages, such as Material Design for Android and Cupertino for iOS, enabling developers to create applications that look and feel native on both platforms. Flutter's hot reload feature significantly enhances the development experience by allowing developers to see changes in real-time without restarting the application, thus speeding up the iteration process. With its growing community and extensive documentation, Flutter has become a popular choice for developers looking to create cross-platform applications efficiently and effectively. Overall, Flutter empowers developers to build high-quality, responsive applications that deliver a seamless user experience across multiple platforms.

2. Backend

Python

Python is a powerful and versatile language ideal for backend development due to its simplicity, readability, and rich ecosystem of frameworks like Django and Flask. These frameworks streamline building robust web applications by handling features like routing, authentication, and database management. Python supports both SQL and NoSQL databases, enabling seamless data handling. Its strong community provides extensive resources and third-party packages, enhancing productivity. With compatibility for APIs, integration capabilities, and performance optimization via tools like FastAPI, Python is top choice for modern backend usages. Python is an ideal backend language, providing simplicity, robust frameworks like Django and Flask, and strong support for databases and API integration.

FastAPI

FastAPI is a modern, high-performance web framework for building APIs with Python, designed to create robust and efficient backend applications. One of its standout features is its ability to handle asynchronous programming, which allows developers to write non-blocking code that can manage multiple requests simultaneously, resulting in improved performance and responsiveness. FastAPI is built on top of Starlette for the web parts and Pydantic for data validation, making it both fast and easy to use. It automatically generates interactive API documentation using Swagger UI and ReDoc, which enhances the developer experience by providing clear insights into the API endpoints and their expected inputs and outputs.

The framework emphasizes type hints and data validation, allowing developers to define data models using Python's type annotations. This not only improves code readability but also helps catch errors early in the development process, leading to more reliable applications. FastAPI is also highly compatible with various databases and can easily integrate with ORMs like SQLAlchemy or Tortoise-ORM, making it a flexible choice for data management. Additionally, its support for dependency injection simplifies the management of application components, promoting cleaner and more maintainable code.

FastAPI is particularly well-suited for building microservices and serverless applications due to its lightweight nature and speed. Its performance benchmarks show that it is one of the fastest Python frameworks available, making it an excellent choice for applications that require high throughput and low latency. Overall, FastAPI provides a powerful and efficient solution for backend development, enabling developers to create scalable and high-performance APIs with minimal effort.

FastAPI is a modern, fast (high-performance) web framework for building APIs with Python. It is based on standard Python type hints, which allows for automatic validation, serialization, and documentation of API endpoints. FastAPI is designed to be fast, reliable, and easy to use, making it ideal for building RESTful APIs quickly while maintaining high performance. It supports asynchronous programming and is built on top of Starlette and Pydantic, offering excellent scalability and security.

3.3.3 Database

Firestore

Firestore is a comprehensive platform developed by Google that provides a suite of tools and services for building and managing applications, with a strong emphasis on real-time data synchronization and cloud-based storage. One of its core offerings is Firestore Realtime Database, a NoSQL cloud database that allows developers to store and sync data in real-time across all connected clients. This feature is particularly beneficial for applications that require instant updates, such as chat applications, collaborative tools, and live dashboards. Firestore also offers Cloud Firestore, a more advanced NoSQL database that provides additional features such as richer querying capabilities, hierarchical data structures, and better scalability. Firestore allows developers to store data in documents organized into collections, making it easier to manage complex data relationships. Both databases support offline capabilities, enabling applications to function seamlessly even when the user is not connected to the internet. Changes made while offline are automatically synchronized when the connection is restored.

SUMMARY

The Efficient Attendance Tracking System requires Windows 11 or higher, Android Studio, and Flutter for the frontend, with Python and FastAPI for the backend. Firestore is used for cloud data storage. Hardware needs include a Quad-core CPU, 4 GB RAM, 1 TB hard disk, ACCUTYPE keyboard, and an optical mouse.

Flutter enables cross-platform development with fast cycles, while FastAPI ensures high performance. Firestore provides real-time data syncing and offline functionality, offering a scalable and efficient attendance tracking solution.

The use of Flutter ensures a consistent user experience across both Android and web platforms. Overall, the combination of these technologies results in a robust, user-friendly, and scalable attendance management solution. This approach also allows for faster development and easier maintenance, reducing the complexity of managing multiple platforms.

CHAPTER 4

SYSTEM DESIGN

1. MODULE DESCRIPTION

A module description provides detailed information about the module and its supported components, which is accessible in different manners.

The project contains the following modules:

- Registration Module
- Camera Module.
- Attendance Status Module.
- Profile Module.

1. Registration Module

The Registration Module is the gateway to the entire system, offering secure access to both teachers and students. This module is designed to authenticate users and grant them access to their respective functionalities. It includes two primary login options, Teacher Login and Student Login. Teacher Login, Teachers enter their credentials (username and password) to access their personal dashboard, where they can manage classes, track student attendance, view reports, and access other administrative tools. After logging in, teachers can perform tasks such as taking attendance using facial recognition, generating attendance reports, managing leave requests, and more. Student Login, Students log in to view their attendance history, check their leave status, request time off, and monitor their performance. Students can also access their profile information, including personal details, and have the ability to track how many sessions they've missed, and request leave approvals through the system.

The Registration Module is built with robust security features such as encryption and secure data handling to ensure that unauthorized users cannot gain access. It uses a secure authentication mechanism to verify the identity of both teachers and students, ensuring the integrity of the system and the privacy of users.

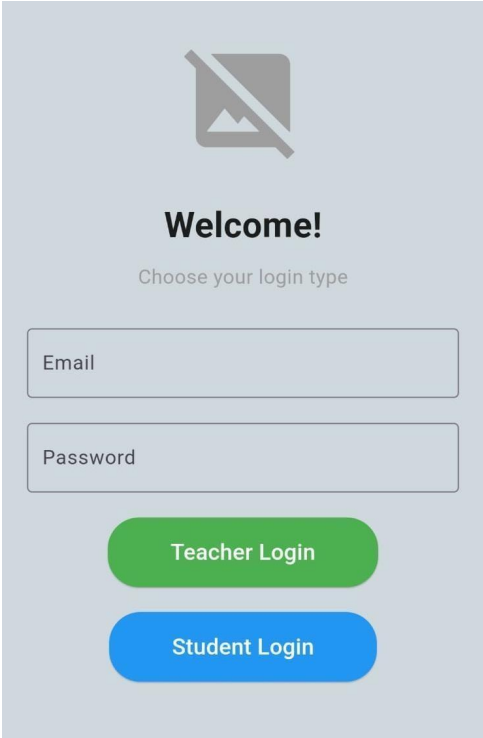

 The image shows a mobile application interface for a registration module. At the top, there is a logo consisting of a square with a diagonal line and a mountain-like shape inside. Below the logo, the text "Welcome!" is displayed in a bold, black font. Underneath "Welcome!", the text "Choose your login type" is shown in a smaller, gray font. There are two input fields: one labeled "Email" and another labeled "Password". Below these fields are two buttons: a green button labeled "Teacher Login" and a blue button labeled "Student Login".

Figure 4.1 Registration Module

4.1.2 Camera Module

The Camera Module is the heart of the facial recognition attendance system, enabling teachers to capture real-time images of students. When a teacher starts a class, the Camera Module uses a camera (integrated into the teacher's mobile) to take pictures of the students present in the class.

This module is responsible for the real-time image capture and synchronization with the facial recognition system. The images taken are immediately processed and compared with the students' pre-stored profiles in the database. The system uses advanced algorithms to identify and authenticate each student by matching the captured image with the stored facial data. Once verified, the system automatically marks the student as present for that session.

The Camera Module efficiently handles large classrooms by processing multiple faces in one shot. It adjusts to various lighting conditions and angles for accurate image capture. In case of a mismatch, the system alerts the teacher for manual verification.

The Camera Module supports multiple cameras, allowing the teacher to scan large groups of students simultaneously. It also ensures high accuracy in facial recognition, even in varying lighting conditions. The system works offline, so the captured data is stored locally and synced with the central server once an internet connection is available. The Camera Module eliminates the need for manual attendance-taking, speeding up the process and ensuring greater accuracy.

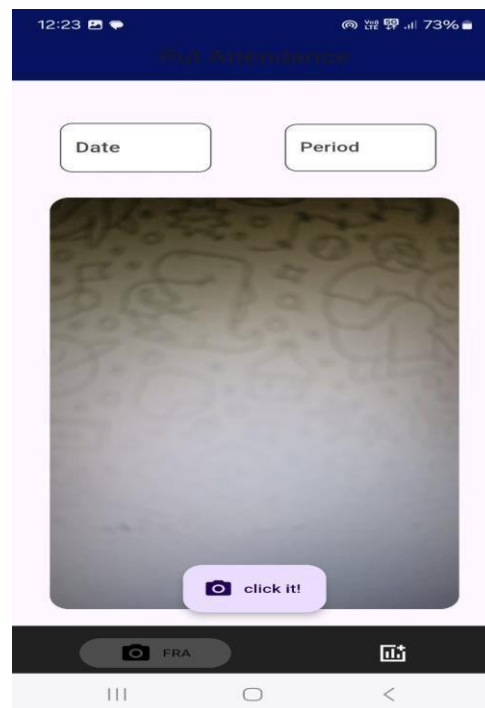


Figure 4.2 Camera Module

4.1.3 Attendance Status Module

The Attendance Status Module tracks and displays the real-time attendance of each student. Once a student's identity is verified by the Camera Module, their attendance status is automatically updated within the system.

For Teachers, This module provides an overview of the entire class's attendance, showing which students are present, absent, or marked for leave. It helps teachers quickly check the attendance status, update or modify records if necessary, and generate reports on attendance trends.

The Attendance Status Module provides real-time updates on student attendance. It allows teachers to quickly view, update, or modify attendance records.

For Students, The Attendance Status Module allows students to view their personal attendance data, see how many classes they have attended or missed, and check their overall attendance percentage. The module may also include features for students to request leave, check if their leave request has been approved or rejected, and view leave history.

The Attendance Status Module is designed for real-time updates, ensuring that attendance information is always accurate and accessible. It also provides a history of attendance records, allowing both teachers and students to track attendance patterns over time.

The module supports customization and ensures that attendance data is automatically saved and securely stored. The Attendance Status Module provides real-time updates on student attendance, allowing teachers to quickly monitor presence. It allows easy modification of attendance records and supports generating detailed attendance reports. This helps in tracking attendance trends and ensuring accurate record-keeping

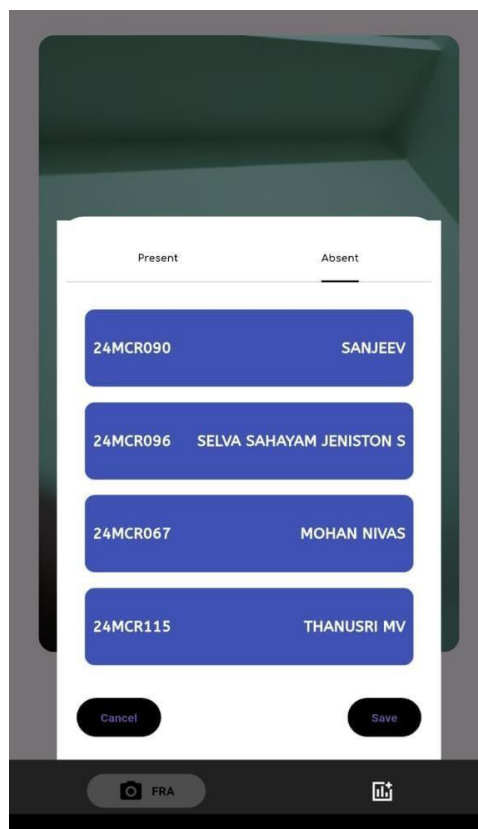


Figure 4.3 Attendance Status Module

4.1.4 Profile Module

The Profile Module allows both teachers and students to manage their personal information, track their attendance history, and maintain records. The module provides a centralized location for users to view and update their details. For Teachers, Teachers can manage their own profiles, including their personal information, login credentials, and their assigned classes. This module also allows teachers to review their teaching schedules, the attendance records of students, and any leave or request approvals for the classes they manage. Teachers can edit their profile information, update their passwords, and check reports of attendance trends in their classes. For Students, Students can access their profile information, including their personal details, attendance history, grades, and leave requests. Students can also track their performance by viewing attendance statistics, including total attended classes, missed sessions, and leave records. The Profile Module ensures that students can update their contact information and request leave directly from their profiles.

The Profile Module is linked to the centralized database, where all user data is securely stored and maintained. The module ensures that personal information is up-to-date and private, while offering both teachers and students easy access to critical records and data. It provides transparency, allowing students to monitor their attendance and performance, and enabling teachers to manage their classes efficiently.

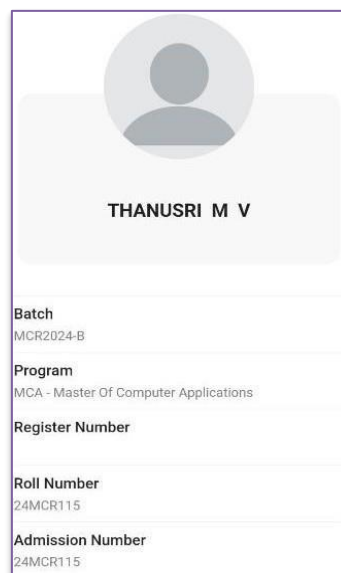


Figure 4.4 Profile Module

4.2 DATAFLOW DIAGRAM

A data flow diagram shows the way information flows through a process or system. It includes data inputs and outputs, data stores, and the various subprocesses the data moves through.

LEVEL 0

The Level 0 diagram shows the overall interaction between the main components of the attendance system. In this diagram, the Teacher captures photos of students and sends them to the System for processing. The system uses Facial Recognition to analyze the photos and mark attendance automatically. The attendance data is then stored in the Database for record-keeping. The system generates an Attendance Report, which is sent back to the teacher for review. This process ensures accurate and efficient attendance management while streamlining the workflow for teachers. The diagram also highlights the seamless flow of data between the camera, recognition system, and database, ensuring minimal delays. It showcases the system's automated nature, reducing the need for manual attendance marking. Additionally, the report generation feature provides teachers with easy access to attendance data for further analysis and decision-making.

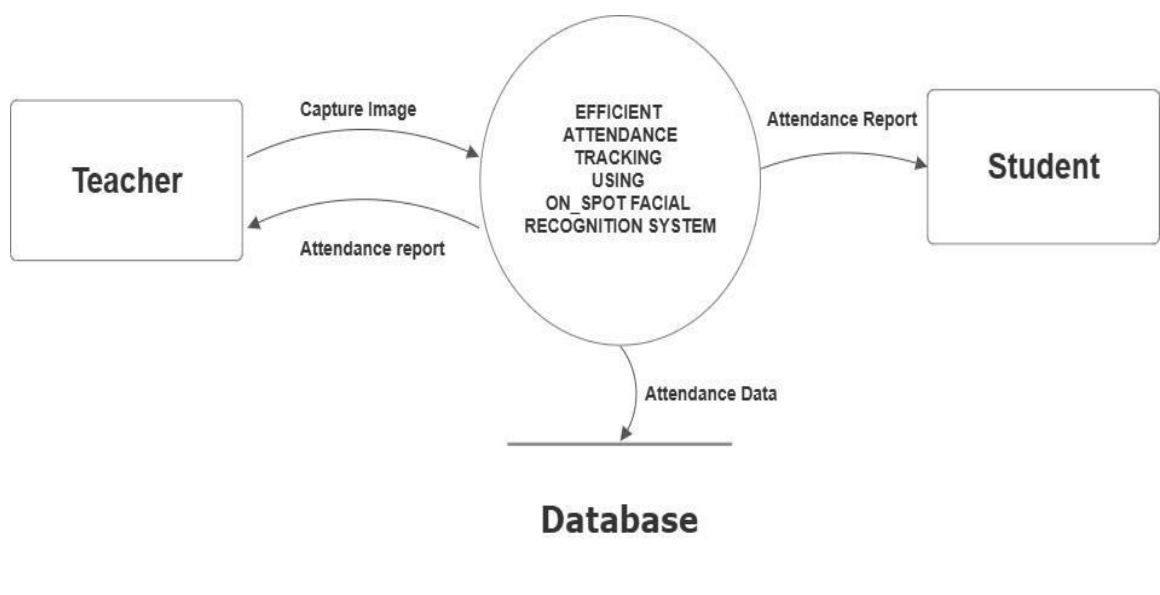


Figure 4.5 Dataflow Diagram Level 0

LEVEL 1

This Level 1 Data Flow Diagram (DFD) provides a detailed view of the Attendance Management System, showing interactions between components. The process starts with the Teacher, who captures photos of students and sends them to the Facial Recognition system for processing. The Facial Recognition component analyzes the photos to identify students and generate attendance data. This Processed Data is sent to the Attendance Manager, which updates the Database with Attendance Records. The Attendance Manager also generates Reports for the teacher, completing the interaction. This diagram highlights the system's flow of data, ensuring attendance is automated, accurate, and well-documented.

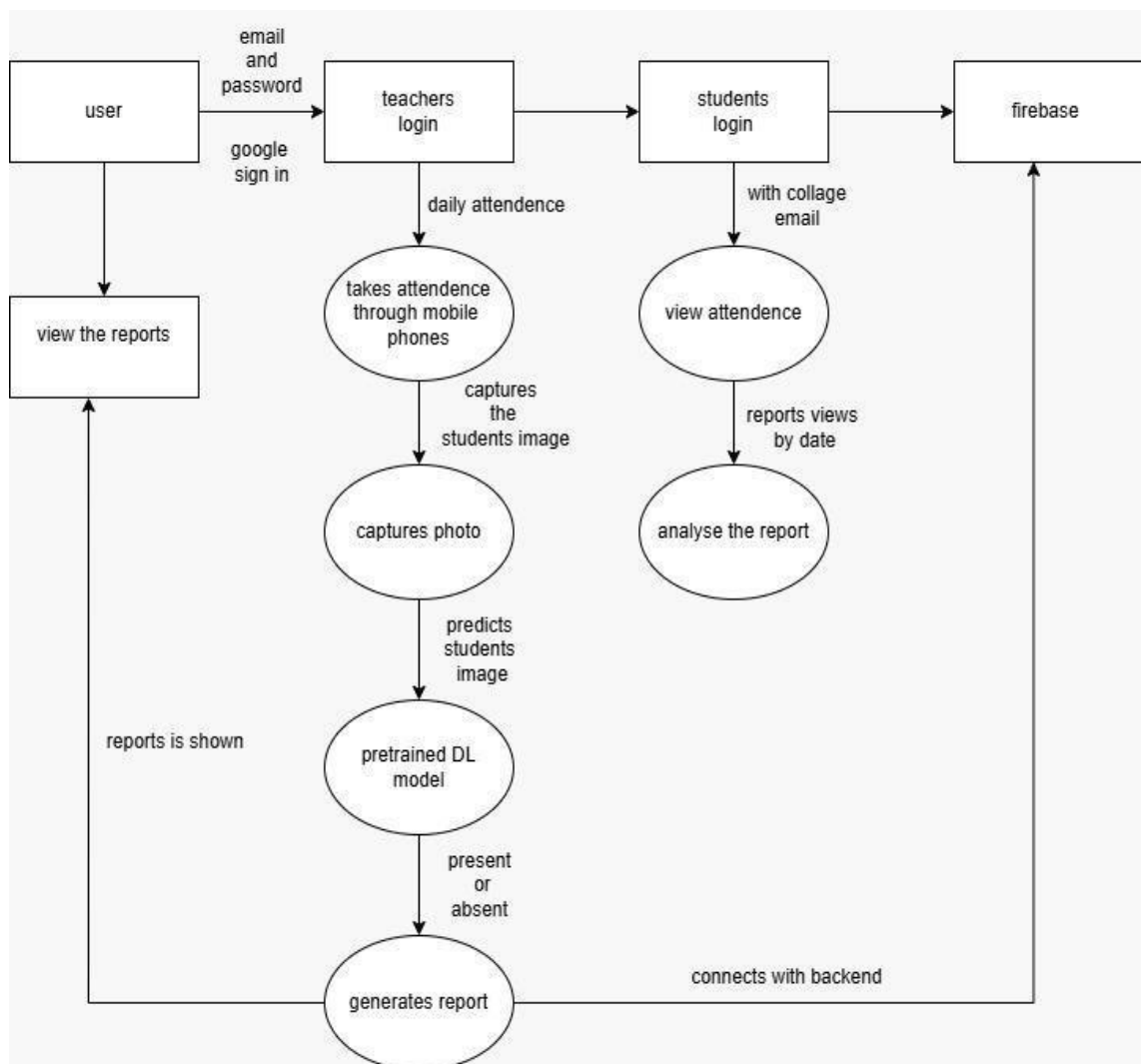


Figure 4.6 Dataflow Diagram Level 1

4.3 DATABASE DESIGN

Database design is the organization of data according to a database model. The designer determines what data must be stored and how the data elements interrelate. With this information, they can begin to fit the data to the database model. Database management system manages the data accordingly. The term database design can be used to describe many different part of the design of an overall database system Principally, and most correctly, it can be thought of as the logical design of the base data structure used to store the data. In an object database the entities and relationships map directly to object classes and named relationships. The term database design could also be used to apply to the overall process of designing, not just the base data structure, but also the forms and queries used as part of the overall database application within the database management system.

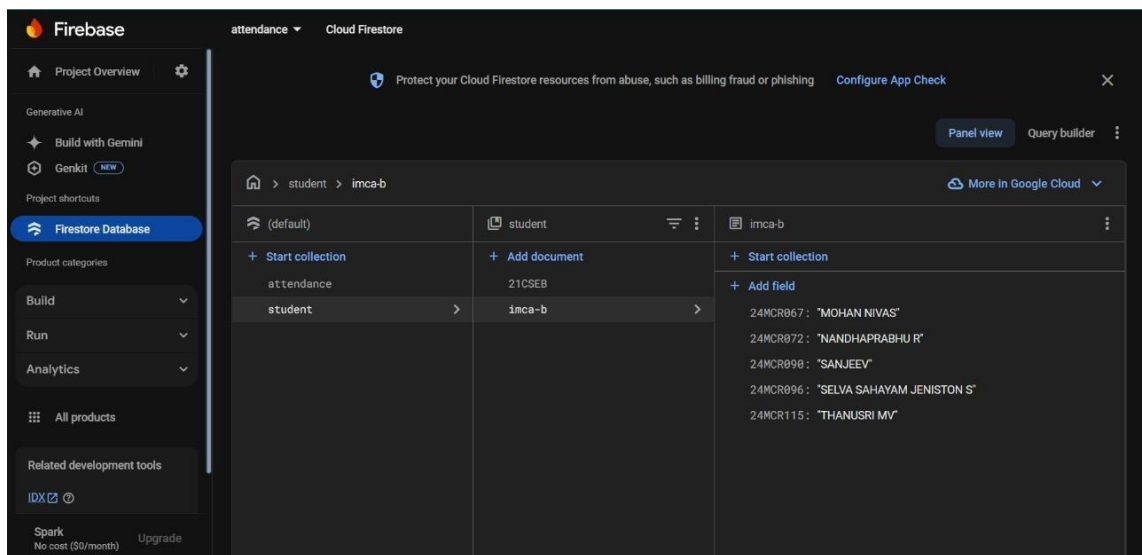


Figure 4.7 Database Design

In Figure 4.7 ,The Firestore database organizes student data using a top-level student collection, where each document (e.g., imca-b) represents a class and contains key-value pairs of unique student IDs and their names. This structure is scalable, flexible, and allows for real-time data access and future enhancements like adding attendance as sub-collections. This structure ensures efficient data management and allows for easy expansion, such as integrating attendance.

4.4 INPUT DESIGN

Input design is the process of converting user-originated inputs to a computer understandable format. Input design is one of the most expensive phase of the operation of computerized system and is often the major problem of a system. A large number of problem with a system can usually be tracked back to fault input design and method. Every moment of input design should be analysed and designed with utmost care. The decisions made during the input design are the project gives the low time consumption to make sensitive application made simple. Thus, the developed system is well within the budget. This was achieved because most of the technologies used are freely available. Only the customized product had to be purchased. In the project, the forms are designed with easy- to-use option. The coding is being done such that proper validation are made to get the prefect input. No error inputs are accepted.

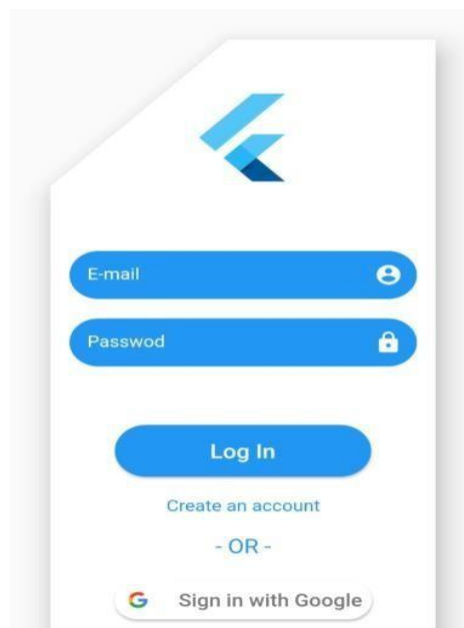
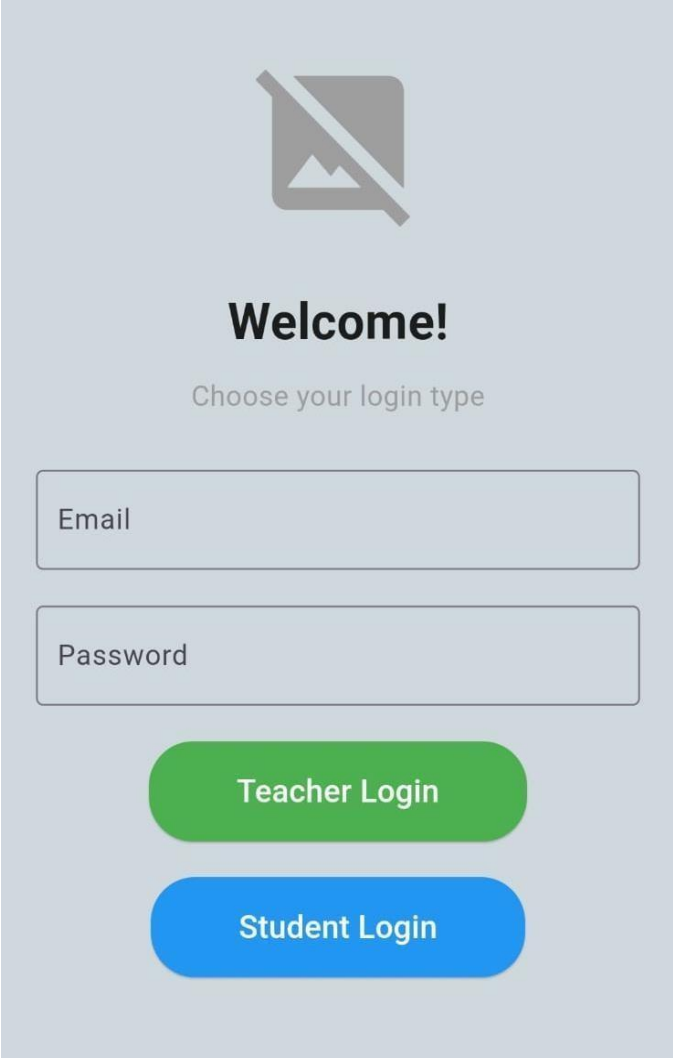



Figure 4.8 Registration Page

Figure 4.8 represents a login UI featuring fields for E-mail and Password, a "Log In" button, and an option to create an account. Additionally, it includes Google Sign- In for a seamless authentication experience. The UI is designed to provide a user-friendly login experience with multiple authentication options. The inclusion of Google Sign-In simplifies access, allowing users to log in quickly.





Welcome!

Choose your login type

Figure 4.9 Sign Up Page

Figure 4.9 depicts a minimalist login interface featuring fields for email and password input. Two brightly colored buttons, "Teacher Login" in green and "Student Login" in blue, allow users to select their role for access. The input fields are centrally aligned for better usability and visibility. Users can quickly switch roles by selecting either the Teacher Login or Student Login button. A compact layout ensures the interface remains intuitive and clutter-free. Thus, the developed system is well within the budget. This was achieved because most of the technologies used are freely available. Only the customized product had to be purchased. In the project, the forms are designed with easy- to-use option. The coding is being done such that proper validation are made to get the prefect input. No error inputs are accepted.

4.5.OUTPUT DESIGN

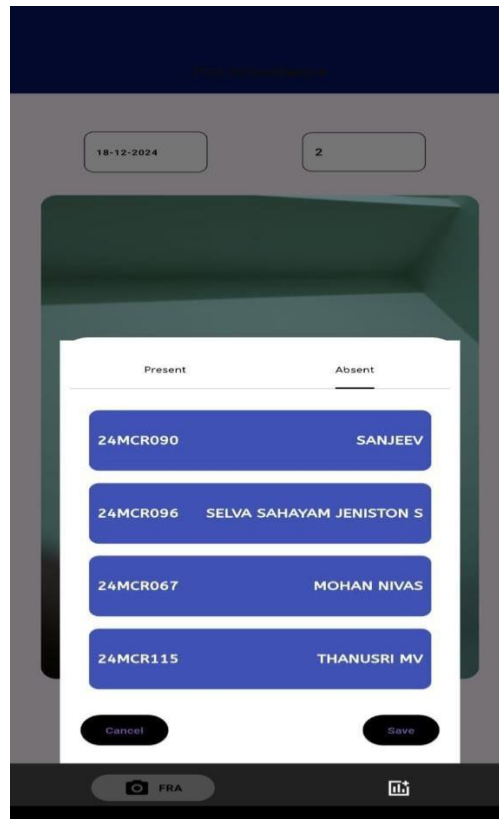


Figure 4.10 Attendance Status Page

Figure 4.10 shows an attendance interface with tabs for marking students as "Present" or "Absent." It includes a list of student IDs and names, date and period selection fields, and buttons to "Cancel" or "Save" the attendance data.

SUMMARY

Chapter 4 details the system design of the attendance tracking platform, covering key modules: Login, Camera, Attendance Status, and Profile. The Login Module allows secure access for teachers and students, while the Camera Module uses facial recognition for automatic attendance tracking. The Attendance Status Module provides real-time attendance updates, and the Profile Module lets users manage their personal and attendance data. Data flow diagrams illustrate the process from image capture to report generation. The database design utilizes Firestore for real-time, scalable data access. Input and output designs ensure efficient user interactions and accurate attendance reporting.

CHAPTER 5

SYSTEM TESTING

System testing is a crucial stage in the software development lifecycle where the entire, fully integrated software system is tested to ensure it meets the specified requirements and is ready for delivery to users. It comes after integration testing, where individual modules are combined and tested together, and before acceptance testing, which validates the system from an end-user perspective. This testing is carried out by a team independent of the developers to ensure an unbiased evaluation of the software.

The primary goal of system testing is to verify that the software works as intended and complies with both functional and non-functional requirements. Functional testing ensures that the system performs its intended tasks as specified in the software requirements, while non-functional testing evaluates critical aspects such as performance, scalability, security, reliability, and usability. Once the development team completes the coding and prepares the necessary documentation, the testing team systematically runs a series of tests designed to uncover flaws and verify functionality. During this process, any issues identified are reported to the developers for resolution, and the system is re-tested to confirm the fixes.

System testing is particularly important because, while developers test individual components and their integrations, some issues might go unnoticed in earlier stages. If not identified and resolved during system testing, these issues could surface later, causing problems for end-users. By rigorously testing the system as a whole, this phase ensures that the software is reliable, meets user expectations, and is ready for deployment, ultimately contributing to the delivery of a high-quality product.

This testing involves both functional and non-functional assessments, verifying that the software performs its intended tasks while also meeting critical standards for performance, security, reliability, and usability. Performed by an independent testing team, system testing identifies flaws that might have been overlooked in earlier stages, providing developers with actionable feedback for improvement.

5.1 UNIT TESTING

Unit testing is the process of testing the smallest parts of a software system, called units, to make sure they work correctly. A unit could be a single function, method, or piece of code that can be tested on its own. The main goal of unit testing is to verify that each part of the code works as expected before combining it with other parts of the system. Developers write test cases that describe how a unit should behave and then run these tests to check if the unit gives the expected results. If there are mistakes or issues, they can be fixed right away.

Unit testing is usually done early in the development process, which helps to find and fix problems quickly. This is important because fixing bugs early makes it easier and cheaper than finding them later when the system is more complex. By testing small parts of the code, unit testing also helps developers understand the code better, making it easier to modify or improve it without introducing new errors.

Sometimes, unit testing involves using tools like mocking or stubbing to control external parts of the system that the unit interacts with. This helps isolate the unit being tested so that it's not affected by other parts of the system during testing. While unit testing is an essential part of ensuring good code quality, it is only one part of the overall software testing process. After unit testing, other types of testing like integration testing and system testing are done to ensure that all the pieces of the software work together smoothly.

Test Case 1

Module	: User Authentication
Test Type	: Unit Testing
Input	: Username and Password
Sample Test	: The system successfully authenticates the user.
Expected Output	: The user is successfully authenticated and granted access to the system.

Analysis : This test ensures that the user authentication module works as expected. The system should correctly authenticate users with valid credentials and deny access for invalid credentials. If the system allows access with incorrect login details or denies access with valid ones, it means there is an issue with the authentication logic that needs to be fixed.

Test Case 2

Module : Facial Recognition

Test Type : Unit Testing

Input : A facial image of a student.

Expected Output : The system should correctly recognize the student's face and return a match with the registered profile.

Sample Test

Output : Student facial image recognized successfully and matched with the registered profile.

Analysis : This test verifies that the facial recognition algorithm processes the input correctly and identifies the student by matching the captured image with the stored profiles. If the recognition process fails to identify the student or matches the wrong profile, it indicates an issue with the facial recognition logic that needs to be addressed

Test Case 3

Module : Attendance Marking

Test Type : Unit Testing

Input : Facial image captured for a student and the corresponding attendance status (Present/Absent).

Expected Output : The system should mark the attendance as "Present" for the student if their face matches the stored profile.

Sample Test

Output : Attendance marked as Present for the student after facial recognition.

Analysis : This test checks if the system correctly marks attendance based on facial recognition. The attendance status should be updated as "Present" only when a valid facial match is found. If the system incorrectly marks the attendance (either for the wrong student or fails to mark attendance), it indicates an issue with the attendance marking logic or facial recognition integration.

5.2 INTEGRATION TESTING

Integration Testing

Integration testing is a process where different parts (or modules) of software are combined and tested together to check if they work well with each other. It helps to find problems that might happen when the modules are connected, such as data not transferring correctly or modules not working together as expected.

Integration testing can involve testing just two modules, several modules, or even the entire system. Each level of testing gets more complex and requires careful planning.

This testing usually happens after unit testing and before system testing. To make it successful, good planning, documentation, and tracking are needed. Integration testing ensures that all parts of the system work together properly and helps deliver a high-quality final product.

Test Case 1

Module : Authentication & Facial Recognition

Test Type : Integration Testing

Input : Admin or teacher login credentials and a student's facial image captured during attendance marking.

Expected Output : The system should authenticate the admin or teacher, and upon successful login, it should allow access to the attendance management features. When a facial image is captured for attendance, the system should seamlessly integrate the recognition process and authenticate the student.

Sample Test

Output : Admin successfully logged in and accessed the attendance dashboard. Student's facial image recognized, and attendance marked successfully

Analysis : This test ensures that the authentication system works properly with the facial recognition system. The integration between login functionality (admin/teacher) and facial recognition for attendance marking must be flawless. Any failure to authenticate or mark attendance correctly indicates an issue with the integration between these modules..

Test Case 2

Module : Facial Recognition & Attendance Database

Test Type : Integration Testing

Input : Capture a student's facial image for attendance marking

Expected Output : The system should successfully match the student's facial image to a profile in the attendance database and mark the attendance accordingly.

Sample Test

Output : Student recognized successfully. Attendance marked for the student.

Analysis : This test checks if the facial recognition module integrates properly with the attendance database. The captured image should match a registered student's profile, and the attendance data should be updated in the database. If the image does not match or the attendance is not recorded, it indicates a failure in the integration between these two modules.

Test Case 3

Module : Offline Mode & Data Synchronization

Test Type : Integration Testing

Input : Attempt to mark attendance in offline mode, then restore the internet connection and sync data.

Expected Output : The system should allow attendance marking to be done offline and then sync the locally stored data with the central database once the internet is restored.

Sample Test

Output : Attendance recorded offline. Data synchronization with central server successful after connection restored

Analysis : This test ensures that the offline functionality integrates well with the system's synchronization feature. If the data fails to be stored offline or does not sync correctly once online, it suggests an issue with the integration of these modules

5.3 VALIDATION TESTING

Validation testing checks if the software meets the needs and expectations of the users and works as intended. It is done after the software is fully developed to ensure it is ready for release. The process starts with understanding the requirements to know what the software should do. Test cases are then created based on these requirements, often using real-world scenarios to see how users will interact with the system. The testing team runs these tests to check if all features work as expected and are easy to use. If any issues or bugs are found, they are reported to the developers for fixing, and the software is tested again to confirm everything is working correctly. The main purpose of validation testing is to ensure that the software solves the problem it was designed for, is reliable, and provides a good user experience. By catching errors early, it helps deliver a high-quality product that meets user needs and is ready for deployment. Validation testing also helps identify any gaps between user expectations and the actual functionality of the software, ensuring alignment with business goals. By simulating real-world usage, it verifies not only the software's functionality but also its overall usability and effectiveness in practical scenarios.

Test Case 1

Module : Attendance Marking

Test Type : Validation Testing

Input : Attempt to mark attendance for a student when no student is detected by the facial recognition system.

Expected Output : An error message should appear stating that no student was detected.

Sample Test

Output : Error: No student detected. Please ensure you are in the frame for attendance marking

Analysis : This test ensures the system detects a student's presence before marking attendance, validating facial recognition functionality.

Test Case 2

Module : User Registration

Test Type : Validation Testing

Input : Attempt to register a student with a facial image already stored in the system for another student.

Expected Output : An error message should appear stating that the facial image is already associated with another student.

Sample Test

Output : Error: This facial image is already associated with another student. Please use a different image.

Analysis : This test ensures the system prevents duplicate facial images during student registration, highlighting a failure in facial recognition validation.

SUMMARY

Chapter 5 covers system testing to ensure the software meets requirements and is ready for deployment. It includes unit testing for components, integration testing for module interactions, and validation testing for user expectations.

CHAPTER 6

SYSTEM IMPLEMENTATION

Systems implementation is about creating an information system, ensuring it is functional, usable, and meets quality standards. If this phase is not well-planned and managed, it can lead to errors, system issues, or failure to meet requirements. To avoid problems, a coordination committee is often formed to address concerns and ensure smooth implementation. Planning involves selecting the right approach and timeline. Staff need to learn new skills early, so they feel confident using the system after training.

Key tasks during implementation include finalizing designs for documents, screens, and databases, coding and testing programs, and preparing user manuals. Testing ensures the system meets user needs, while proper planning and control are essential for a successful system installation.

The focus throughout the implementation phase needs to be on learning new skills to give staff member's confidence that they can use the system after training. Complete as necessary the design contained in the approved systems design document. For example, the detailed contents of new or revised documents, computer screens, and database must be laid out and created. Write, test, and document the programs and procedures required by the approved systems design document.

Ensure, by completing the preparation of user manuals and other documentation and by training personnel, that the organization's personnel can operate the new system. Determine, by thoroughly testing the system with users, that the system satisfies the users requirements. Ensure a correct conversion by planning, controlling, and conducting an orderly installation of the new system.

Systems implementation ensures the system is functional, usable, and meets quality standards. Key tasks include finalizing designs, coding, testing, and preparing documentation. Proper planning, staff training, and thorough testing are essential to avoid issues. Successful implementation requires careful installation and user requirement validation.

6.1 ERROR HANDLING

Error handling is a critical aspect of software development that ensures the system can gracefully manage unexpected situations or failures. It involves anticipating potential errors, such as invalid input, system crashes, or network issues, and implementing mechanisms to detect, report, and recover from them. Effective error handling improves the user experience by providing clear error messages and guiding users on how to resolve issues. It also helps maintain system stability and security by preventing errors from compromising the software's functionality. By carefully designing and testing error handling routines, developers ensure the software remains robust and reliable in diverse operational environments.

6.2 TESTING

The testing phase commences simultaneously with the coding procedure, allowing for the examination of each software module as it is developed. This parallel progression ensures an iterative and collaborative approach, where coding and testing activities occur concurrently. As each module is coded, it undergoes thorough testing to identify and rectify any potential issues early in the development process. This integrated approach facilitates real-time feedback, fostering a more efficient and streamlined development lifecycle.

6.3 INSTALLATION

Installation refers to the procedure of substituting an existing device with a new one. This involves the conversion of existing data, software, documentation, and painting techniques to formats compatible with the new device. Additionally, installation encompasses the configuration of hardware components and the integration of software necessary for the seamless operation of the new device. It is a critical phase that demands meticulous attention to detail to ensure a smooth transition, minimizing disruptions in workflow. Installation involves migrating data and software, configuring hardware, and ensuring seamless integration for minimal disruption and smooth operation of the new device.

Regular testing during the installation process is imperative to validate the compatibility of the existing data and software with the new device, and any necessary adjustments are made to optimize performance.

6.4 DOCUMENTATION

Documentation comprises individual publications that offer comprehensive information on effectively utilizing the system and navigating its workflow following the setup procedure. These materials serve as valuable resources, providing users with step-by-step instructions, guidelines, and insights into the system's functionalities. They aim to empower users with the knowledge required for seamless interaction with the system, covering aspects such as system features, troubleshooting, and best practices.

6.5 TRAINING

A training plan serves as a structured approach to rapidly educate users on the effective utilization of a new system. It is a systematic method designed to impart the necessary skills and knowledge, enabling consumers to efficiently navigate and leverage the features of the newly implemented system. The training plan outlines the curriculum, instructional methods, and resources required to facilitate a smooth learning process for end-users.

6.6 SUPPORT

The educational approach underwent a likely enhancement before the commencement of the project, signifying a proactive measure to ensure optimal preparation. This improvement might have involved refining instructional strategies, updating learning materials, and incorporating the latest pedagogical techniques. A well-prepared educational foundation is crucial for project success, fostering a conducive environment for effective learning and skill development.

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

1. CONCLUSION

The Efficient Attendance Tracking Using On-Spot Facial Recognition System Efficient Attendance Tracking Using On-Spot Facial Recognition System offers a modern and effective solution to the challenges of traditional attendance systems. By utilizing facial recognition technology and mobile devices, the system ensures accurate, secure, and efficient tracking of student attendance. It eliminates the risk of proxy attendance, reduces manual effort, and guarantees real-time data synchronization, even when offline. This system provides operational efficiency, accuracy, and scalability, making it an ideal choice for educational institutions seeking to improve attendance management. In conclusion, the system modernizes the attendance process, providing a more secure, user-friendly, and reliable alternative to conventional methods, ultimately enhancing the overall educational experience.

2. FUTURE ENHANCEMENT

Future enhancements to the Efficient Attendance Tracking Using On-Spot Facial Recognition System could include,

- **Enhanced Facial Recognition:** Incorporating advanced machine learning algorithms to improve recognition accuracy under varying conditions, such as poor lighting or mask usage.
- **Real-Time Notifications:** Adding real-time alerts to inform students and teachers of attendance status or discrepancies for improved communication and experience.
- **System Integration and Accessibility:** Expanding the system to integrate with grade books, scheduling systems, and multi-platform support (desktop and web applications) for a comprehensive and accessible solution.

APPENDIX-A SAMPLE CODING

Login.dart

```
import 'package:attendancesystem/presistant.dart';
import 'package:flutter/material.dart';

import 'package:firebase_auth/firebase_auth.dart';
//import 'package:mp/screens/teachers_login.dart';
//import 'package:mp/screens/students_login.dart';

class LoginPage extends StatelessWidget {
  final FirebaseAuth _auth = FirebaseAuth.instance;

  Future<void> loginUser(String email, String password, BuildContext context) async {
    try {
      // Sign in user with Firebase Authentication
      UserCredential userCredential = await _auth.signInWithEmailAndPassword(
        email: email,
        password: password,
      );

      if (userCredential.user != null) {
        // Navigate to Teacher or Student page based on email (You can improve this logic)
        if (email.contains("teacher")) {
          // Navigator.pushReplacement(
            // context,
            // MaterialPageRoute(builder: (context) => Teachers_login()),
          // );
        }
      }
    }
  }
}
```

```

    } else {
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => MyHomePage1()),
      );
    }
  }
} catch (e) {
  // Show error message
  ScaffoldMessenger.of(context).showSnackBar(
    SnackBar(content: Text("Login failed: $e")),
  );
}

@override
Widget build(BuildContext context) {
  TextEditingController emailController = TextEditingController();
  TextEditingController passwordController = TextEditingController();

  return Scaffold(
    backgroundColor: Colors.blueGrey.shade100,
    body: Center(
      child: Padding(
        padding: const EdgeInsets.symmetric(horizontal: 20.0, vertical: 30.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [

```

```

// Logo
Image.asset(
  'assets/login_illustration.png',
  height: 250,
  errorBuilder: (context, error, stackTrace) => const Icon(
    Icons.image_not_supported,
    size: 100,
    color: Colors.grey,
  ),
),
const SizedBox(height: 30),

const Text(
  "Welcome!",
  style: TextStyle(
    fontSize: 28,
    fontWeight: FontWeight.bold,
    color: Colors.black87,
  ),
),
const SizedBox(height: 10),
const Text(
  "Choose your login type",
  style: TextStyle(fontSize: 16, color: Colors.grey),
),
const SizedBox(height: 30),

// Input fields

```

```

TextField(
  controller: emailController,
  decoration: const InputDecoration(
    labelText: "Email",

    border: OutlineInputBorder(),
  ),
),
const SizedBox(height: 20),
TextField(
  controller: passwordController,
  obscureText: true,
  decoration: const InputDecoration(
    labelText: "Password",

    border: OutlineInputBorder(),
  ),
),
const SizedBox(height: 20),

// Login Buttons
ElevatedButton(
  onPressed: () {
    // Perform Firebase Login for Teachers
    loginUser(emailController.text, passwordController.text, context);
  },
  style: ElevatedButton.styleFrom(
    backgroundColor: Colors.green,
    padding: const EdgeInsets.symmetric(horizontal: 50, vertical: 15),
    shape: RoundedRectangleBorder(

```

```

borderRadius: BorderRadius.circular(25),

),

),

child: const Text(

"Teacher Login",

style: TextStyle(fontSize: 18, color: Colors.white),

),

),

const SizedBox(height: 20),

ElevatedButton(

onPressed: () {

// Perform Firebase Login for Students

loginUser(emailController.text, passwordController.text, context);

},

style: ElevatedButton.styleFrom(

backgroundColor: Colors.blue,

padding: const EdgeInsets.symmetric(horizontal: 50, vertical: 15),

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(25),

),

),

child: const Text(

"Student Login",

style: TextStyle(fontSize: 18, color: Colors.white),

),

),

],

),

```

```

),
),
);
}
}

```

Main.dart

```

import 'dart:convert';
import 'dart:async';

import 'package:attendancesystem/login.dart';
import 'package:attendancesystem/presistant.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';

import 'package:http/http.dart' as http;
import 'package:firebase_auth/firebase_auth.dart';

import 'firebase_options.dart';
final FirebaseAuth _auth = FirebaseAuth.instance;
void main() async{
WidgetsFlutterBinding.ensureInitialized();

await Firebase.initializeApp(
  options: DefaultFirebaseOptions.currentPlatform,
);
runApp(const MyApp());

}

class MyApp extends StatelessWidget {
  const MyApp({super.key});

```

```
// This widget is the root of your application.

@override
Widget build(BuildContext context) {
  return MaterialApp(
    debugShowCheckedModeBanner: false,
    title: 'Flutter Demo',
    theme: ThemeData(

      primarySwatch: Colors.blue,
    ),
    home: ( _auth.currentUser?.email) != null? MyHomePage1():LoginPage(),
  );
}
}
```

Signup.dart

```
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';

class SignupPage extends StatelessWidget {
  final FirebaseAuth _auth = FirebaseAuth.instance;

  // Function to handle user signup
  Future<void> signupUser(String email, String password, BuildContext context) async
  {
    try {
      // Create user with Firebase Authentication
```



```

UserCredential userCredential = await _auth.createUserWithEmailAndPassword(
  email: email,
  password: password,
);

if (userCredential.user != null) {
  // Show success message
  ScaffoldMessenger.of(context).showSnackBar(
    const SnackBar(content: Text("Signup successful! Please login.")),
  );
  // Navigate back to the login page
  Navigator.pop(context);
}
} catch (e) {
  // Show error message
  ScaffoldMessenger.of(context).showSnackBar(
    SnackBar(content: Text("Signup failed: $e")),
  );
}
}

@override
Widget build(BuildContext context) {
  TextEditingController emailController = TextEditingController();
  TextEditingController passwordController = TextEditingController();
  TextEditingController confirmPasswordController = TextEditingController();

  return Scaffold(

```

```

backgroundColor: Colors.blueGrey.shade100,

appBar: AppBar(
  title: const Text("Signup"),
  backgroundColor: Colors.green,
),
body: Center(
  child: Padding(
    padding: const EdgeInsets.symmetric(horizontal: 20.0, vertical: 30.0),
    child: SingleChildScrollView(

      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          // Signup Illustration
          Image.asset(
            'assets/signup_illustration.png',
            height: 200,
            errorBuilder: (context, error, stackTrace) => const Icon(
              Icons.image_not_supported,
              size: 100,
              color: Colors.grey,
            ),
          ),
          const SizedBox(height: 20),

          const Text(
            "Create an Account",
            style: TextStyle(
              fontSize: 24,

```

```
        fontWeight: FontWeight.bold,  
        color: Colors.black87,  
      ),  
    ),  
    const SizedBox(height: 30),  
  
    // Email Field  
    TextField(  
      controller: emailController,  
      decoration: const InputDecoration(  
        labelText: "Email",  
  
        border: OutlineInputBorder(),  
      ),  
    ),  
    const SizedBox(height: 20),  
  
    // Password Field  
    TextField(  
      controller: passwordController,  
      obscureText: true,  
      decoration: const InputDecoration(  
        labelText: "Password",  
  
        border: OutlineInputBorder(),  
      ),  
    ),  
    const SizedBox(height: 20),  
  
    // Confirm Password Field
```

```

TextField(
  controller: confirmPasswordController,
  obscureText: true,
  decoration: const InputDecoration(
    labelText: "Confirm Password",
    border: OutlineInputBorder(),
  ),
),
const SizedBox(height: 30),

// Signup Button
ElevatedButton(
  onPressed: () {
    // Check if passwords match
    if (passwordController.text == confirmPasswordController.text) {
      signupUser(emailController.text, passwordController.text, context);
    } else {
      ScaffoldMessenger.of(context).showSnackBar(
        const SnackBar(content: Text("Passwords do not match!")),
      );
    }
  },
  style: ElevatedButton.styleFrom(
    backgroundColor: Colors.green,
    padding: const EdgeInsets.symmetric(horizontal: 50, vertical: 15),
    shape: RoundedRectangleBorder(
      borderRadius: BorderRadius.circular(25),
    ),
  ),

```

```

    ),
    child: const Text(
      "Signup",
      style: TextStyle(fontSize: 18, color: Colors.white),
    ),
  ),

  const SizedBox(height: 20),

  // Navigate to Login Page
  TextButton(
    onPressed: () {
      Navigator.pop(context);
    },
    child: const Text(
      "Already have an account? Login here",
      style: TextStyle(color: Colors.blue, fontSize: 16),
    ),
  ),
],
),
),
),
),
),
);
}
}

```

APPENDIX-B SCREENSHOTS

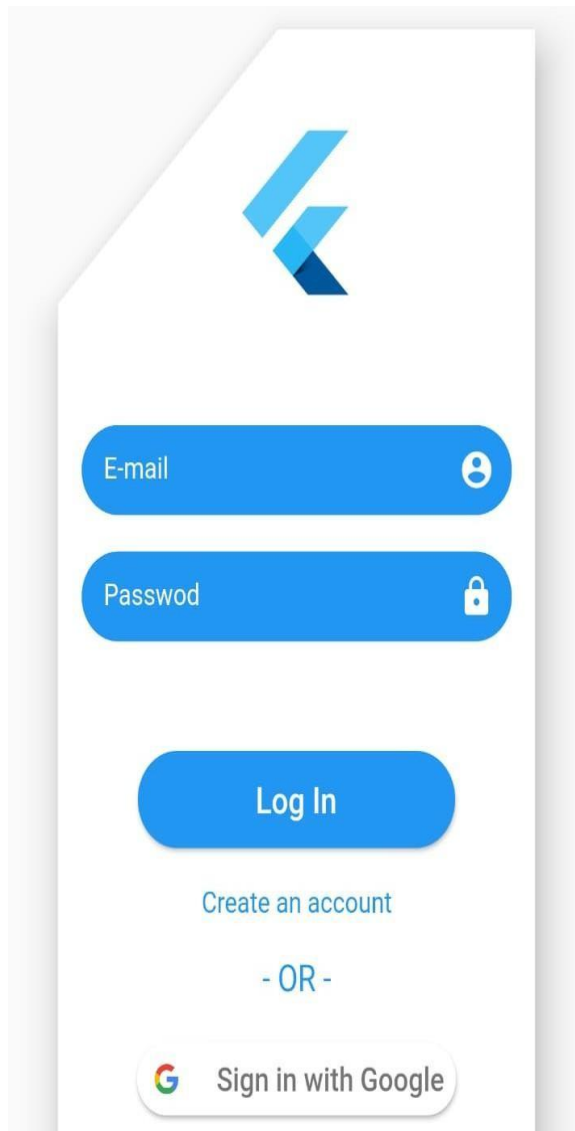


Figure B.1 Registration Page

Figure B.1 represents a login UI featuring fields for E-mail and Password, a "Log In" button, and an option to create an account. Additionally, it includes Google Sign-In for a seamless authentication experience. The login UI is designed to be simple and user-friendly, ensuring quick access for users. The inclusion of a "Log In" button allows easy submission of credentials, while the "Create Account" option facilitates new user registration. Google Sign-In integration provides an alternative, convenient way for users to authenticate without needing to remember additional credentials.

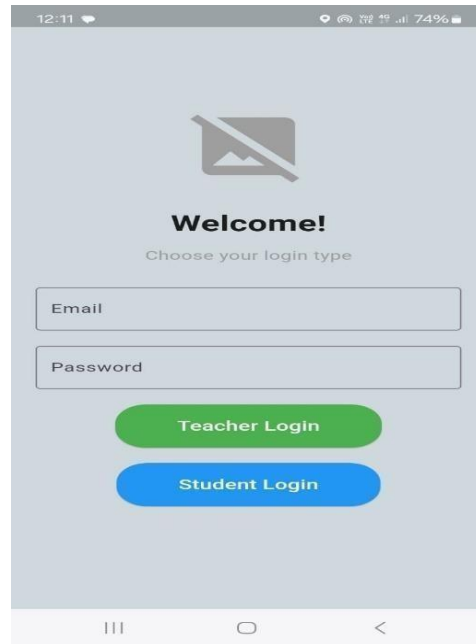


Figure B.2 Login Page

Figure B.2 depicts a minimalist login interface featuring fields for email and password input. Two brightly colored buttons, "Teacher Login" in green and "Student Login" in blue, allow users to select their role for access.

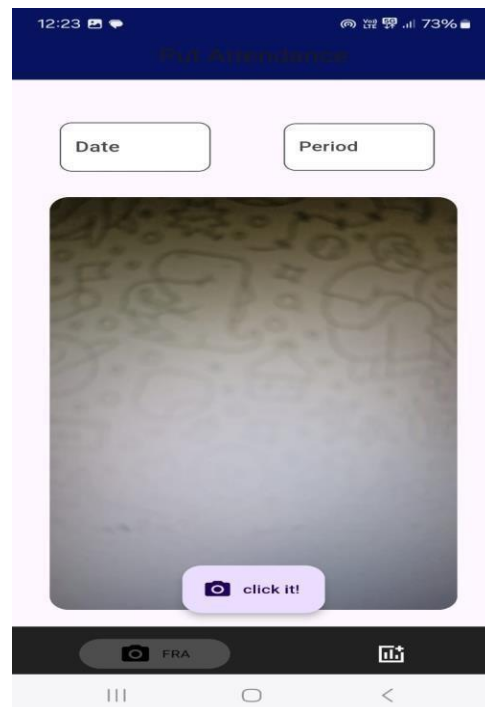


Figure B.3 Camera Page

In Figure B.3, Teacher's can select the Date and period and capture the student's Photo and the system will process and mark the attendance status.

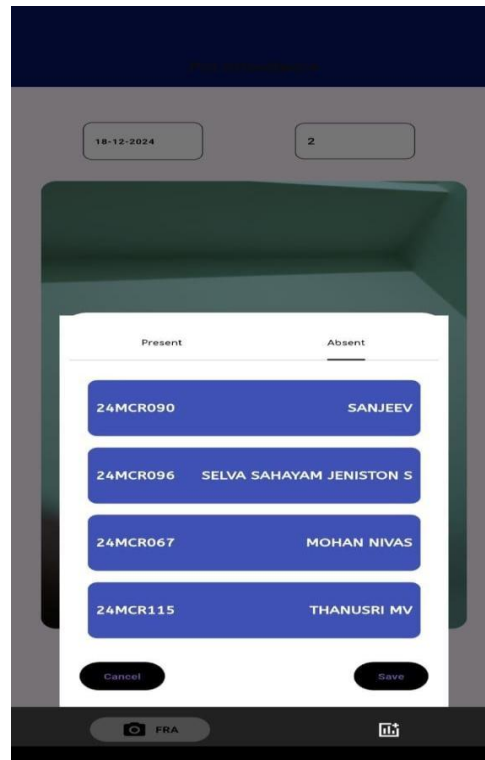


Figure B.4 Attendance Status Page

Figure B.4 displays an attendance tracking interface with options to mark students as "Present" or "Absent." It includes fields for selecting the date and period at the top, a list of student IDs and names categorized under attendance status, and two buttons at the bottom labeled "Cancel" and "Save" for finalizing the input. A camera option and additional icons are visible at the bottom for capturing attendance-related data.

SUMMARY

Chapter 7 summarizes the conclusion and future enhancements of the "Efficient Attendance Tracking Using On-Spot Facial Recognition System." The system provides an effective solution to traditional attendance challenges by using facial recognition and mobile devices for accurate, secure, and efficient attendance tracking. It eliminates proxy attendance, reduces manual effort, and supports real-time data synchronization, even offline, making it a scalable and reliable tool for educational institutions. Future improvements could include enhanced facial recognition for varying conditions, real-time notifications for better communication, and system integration with grade books and scheduling platforms for a more comprehensive solution.

REFERENCES

1. R. M. Pratama, R. P. Surya, and E. P. S. Rudi, "Face Recognition Technology for Attendance System," IEEE International Conference on Electrical Engineering and Informatics, pp. 211-215, 2018.
2. M. Park, H. S. Ryu, and J. Y. Lee, "A Facial Recognition Attendance System Using Deep Learning Algorithms," Journal of Electrical Engineering & Technology, vol. 13, no. 6, pp. 2053-2059, 20
3. <https://towardsdatascience.com/face-recognition-using-deep-learning-with-python-8cfdb3029079> (Face Recognition with Deep Learning)
4. <https://flutter.dev/docs/get-started/install> (Building Mobile Applications with Flutter)
5. <https://www.driftrack.com/offline-first-approach-to-building-mobile-apps/> (Offline Storage Solutions for Mobile App)