

**A Project Report
On
“Face Recognition Based Attendance
Management System”
Submitted to the
Department of MCA
In partial fulfillment of the
MASTER OF COMPUTER APPLICATIONS**

**Under the guidance of
Dr. Lince Rachel Varghese**

**Project Done By
JOHN GEORGE
Reg No:213242210229**



**DEPARTMENT OF MCA
UNION CHRISTIAN COLLEGE
ALUVA, KERALA
August 2023**

**UNION CHRISTIAN COLLEGE
ALUVA, KERALA**



BONAFIDE CERTIFICATE

**Certified that the Project Work entitled
“Face Recognition Based Attendance
Management System”
is a bonafide work**

done by

John George

Reg No:213242210229

In partial fulfillment of the requirement for the Award of

MASTER OF COMPUTER APPLICATIONS

Degree From

Mahatma Gandhi University, Kottayam

(2021-2023)

Mrs. Sherna Mohan

Head of the Department

Dr Lince Rachel Varghese

Project Guide

Submitted for the Viva-Voce Examination held on.....

External Examiner1

(Name & Signature)

External Examiner2

(Name & Signature)

**UNION CHRISTIAN COLLEGE
ALUVA, KERALA**



CERTIFICATE

This is to certify that the project entitled “Face Recognition Based Attendance Management System” has been successfully carried out by John George (Reg. No: 213242210229) in partial fulfilment of the Course Master of Computer Applications.

Mrs. Sherna Mohan

HEAD OF THE DEPARTMENT

Dr Lince Rachel Varghese

INTERNAL GUIDE

Date:

**UNION CHRISTIAN COLLEGE
ALUVA, KERALA**



CERTIFICATE

This is to certify that the project entitled “Face Recognition Based Attendance Management System” has been successfully carried out by John George (Reg no: 213242210229) in partial fulfilment of the course Master of Computer Applications under my guidance .

Date:

Dr Lince Rachel Varghese

INTERNAL GUIDE

**UNION CHRISTIAN COLLEGE
ALUVA, KERALA**



DECLARATION

I, John George , hereby declare that the project work entitled “Face Recognition Based Attendance Management System” is an authenticated work carried out by me at Signature Resources Hub under the guidance of Mr Ajin Mohan for the partial fulfilment of the course Master of Computer Applications. This work has not been submitted for similar purpose anywhere else except to Union Christian College Aluva.

I understand that detection of any such copying is liable to be punished in any way the school deems fit.

Date:

Place:

JOHN GEORGE

Signature

ACKNOWLEDGEMENT

First and foremost, of all I express our gratitude to God almighty for giving me an opportunity to excel in my efforts to complete this main project on time.

I wish to express my deep sense of gratitude to **Dr M.I PUNNOOSE, Principal of Union Christian College, Aluva** for the support provided to me. We would like to express my gratitude and thanks to **Dr A V Alex, Director of MCA and Mrs. Sherna Mohan, Head of the Department** for providing me the best facilities and atmosphere for completion of my main project.

My sincere thanks to **Project Coordinators, Mrs. Veena Jose ,Dr Lince Rachel Varghese, Mrs. Shikha kb and our Internal Guide Dr Lince Rachel Varghese**, for their valuable guidance that helped me to complete my main project work in a determined way with in stipulated time.

Finally, I would like to express my sincere thanks to my family and friends, for always being a source of inspiration, their undying support and encouragement without which this main project wouldn't have been a success.

Table of Contents

ABSTRACT.....	4
Chapter 1 Introduction.....	5
1.1 Introduction.....	5
1.2 Problem Statement.....	5
1.3 Scope and Relevance of the Project.....	6
1.4 Objectives	6
Chapter 2 System Analysis	9
2.1 Introduction.....	9
2.2 Existing System	9
2.3 Proposed System.....	10
2.4 Feasibility Study	12
2.4.1 Technical Feasibility	12
2.4.2 Economical Feasibility.....	13
2.4.3 Operational Feasibility.....	14
2.5 Software Engineering Paradigm Applied.....	14
Chapter 3 System Design.....	16
3.1 Introduction.....	16
3.2 Database Design.....	17
3.2.1 Table Name: Student.....	17
3.2.2 Table Name: Attendance	17
3.2.3 Table Name: Issue Reporting.....	17
3.3 Object Oriented Design-UML Diagrams	17
3.3.1 Use Case Diagram.....	17
3.3.2 Activity Diagram.....	18
3.3.3 Class Diagram.....	18
3.3.4 Sequence Diagram	18
3.4 Modular Design	18
3.5 Design	19
3.5.1 Login.....	19
3.5.2 User Registration.....	19
3.5.3 Training Image	20
3.5.4 Marking Attendance.....	20
3.4.5 Admin Home Page	20

3.5.6 User Home Page.....	20
3.5.7 Tracking Attendance by admin	20
3.5.8 Attendance Track by User	20
3.5.9 Issue Reporting	20
3.5.10 User Profile	20
Chapter 4 System Environment.....	21
4.1 Software Requirements Specification.....	21
4.2 Tools, Platforms	22
4.2.1 Front End Tool:Tkinter	22
4.2.2 Back End Tool.....	23
4.2.3 Operating System.....	24
Chapter 5 System Implementation	26
5.1 Coding.....	26
Chapter 6 System Testing	27
6.1 Introduction.....	27
6.2 Unit Testing.....	27
6.3 Integration Testing	33
6.4 System Testing	36
Chapter 7 System Maintenance	41
7.1 Introduction.....	41
7.2 Maintenance	41
Chapter 8 System Security Measures.....	43
8.1 Introduction.....	43
8.2 Operating System-Level Security:	43
8.3 Database Level Security:	44
Chapter 9 System Planning and Scheduling.....	46
9.1 Introduction.....	46
9.2 GANNT Chart.....	47
Chapter 10 System Cost Estimation	48
10.1 Introduction.....	48
10.2 LOC Based Estimation / Function Point based Estimation	49
Chapter 11 Conclusion.....	51
Chapter 12 Future Enhancement and Scope of further Development.....	52
__12.1 Introduction.....	52

12.2 Merits of the System	52
12.3 Limitations of the System	54
12.4 Future Enhancement of the System	55
Chapter 13 Annexure.....	58
13.1 Organization profile	58
13.2 Annexure 1	59
1.1 Table Name: Student	59
1.2 Table Name: Attendance	60
1.3 Table Name: Issue Reporting	60
_13.3 Annexure 2	61
2.1 Use Case Diagram.....	61
2.2 Activity Diagram.....	62
2.3 Class Diagram.....	69
2.4 Sequence Diagram	70
_13.4 Annexure 3	72
3.1 Login	72
3.2 User Registration	72
3.3 Training Image	73
3.4 Marking Attendance	73
3.5 Admin Home Page	74
3.6 User Home Page	74
3.7 Tracking Attendance by admin	75
3.8 Attendance Track By User	75
3.9 Issue Reporting	76
3.10 User Profile	76
13.5 Annexure 4	77
13.6 References.....	103

ABSTRACT

Attendance marking in a organisation is not only an difficult task but also a time consuming one at that. Due to an unusually high number of user there will always be a probability of proxy attendance. Attendance marking with conventional methods has been an area of challenge. The growing need of efficient and automatic techniques of marking attendance is a growing challenge in the area of face recognition. In recent years, the problem of automatic attendance marking has been widely addressed through the use of standard biometrics like fingerprint and Radio frequency Identification tags etc., However, these techniques lack the element of reliability. In this proposed project an automated attendance marking and management system is proposed by making use of face detection and recognition algorithms. Instead of using the conventional methods, this proposed system aims to develop an automated desktop application that records the employee's attendance by using facial recognition technology. The main objective of this work is to make the attendance marking and management system efficient, time saving, simple and easy. Here faces will be recognized using the face recognition algorithm LBPH(Local Binary Pattern Histogram). The processed image will then be compared against the existing stored record and then attendance is marked in the database accordingly. Compared to existing traditional attendance marking system, this system reduces the workload of people. This proposed system will be implemented with 4 phases such as Image Capturing, Training, Face Detection for Attendance, and Updating of Attendance in database and issue reporting.

Chapter 1 Introduction

1.1 Introduction

The Face Recognition-Based Attendance Management System (FRAMS) introduces a revolutionary solution for attendance management, replacing traditional methods with cutting-edge face recognition technology. This system incorporates a range of functionalities to enhance the entire attendance process. To begin, users can easily register by providing their information and capturing a facial image, which is then used for training the face recognition model. Once registered, marking attendance becomes effortless - users simply stand in front of the face recognition device, and the system records their attendance with a timestamp. Administrators benefit from real-time tracking of attendance, generating comprehensive reports, and managing user profiles. The advantages of this project are numerous, including heightened accuracy, time efficiency, improved security, and reduced costs. As organizations embrace this user-friendly and innovative system, they are propelled into a more streamlined and technologically advanced future of attendance management.

1.2 Problem Statement

The proposed project aims to tackle the limitations and inefficiencies of traditional attendance management methods by developing a Face Recognition-Based Attendance Management System (FRAMS). The existing manual processes, such as paper-based registers and calling out names, often lead to inaccuracies, proxy attendance, and increased administrative burden. To address these issues, this project will leverage cutting-edge face recognition technology to provide a more accurate, secure, and user-friendly approach to attendance tracking. The key objectives include developing a robust face recognition model for precise identification, automating attendance marking to enhance efficiency, ensuring stringent data protection measures for user privacy, and creating an intuitive interface for easy user registration and attendance marking. Real-time monitoring and comprehensive reporting features will offer administrators valuable insights into attendance patterns, allowing timely interventions when needed. By integrating seamlessly with existing systems and adapting to varying environmental conditions, this system promises to revolutionize attendance management, delivering enhanced accuracy, efficiency, and security for organizations and educational institutions alike.

1.3 Scope and Relevance of the Project

The scope of the Face Recognition-Based Attendance Management System (FRAMS) encompasses the development of a sophisticated and automated system that utilizes state-of-the-art face recognition technology to revolutionize attendance management. The project involves designing a robust face recognition algorithm capable of accurately identifying individuals from their facial features, implementing a user-friendly interface for user registration, and creating a secure database to store facial data, relevant user information, and issue details. The system will offer real-time attendance tracking, generating comprehensive reports for administrators, and ensuring seamless integration with existing attendance systems. The scope also includes addressing data security and privacy concerns, adhering to ethical and legal standards to safeguard user information.

The relevance of this project is significant in the current technological landscape, where organizations and institutions strive for efficiency and accuracy in managing their workforce or student attendance. By automating attendance marking through face recognition, the system eliminates manual errors, reduces administrative burden, and mitigates the possibility of fraudulent attendance practices like proxy attendance. In the wake of the COVID-19 pandemic, the contactless nature of this project aligns perfectly with the need for hygienic practices, ensuring a safer attendance management process. The project's emphasis on user-friendliness makes it accessible to a wide range of users, easing the adoption process and increasing user satisfaction. With real-time monitoring and reporting capabilities, administrators can make data-driven decisions promptly, improving overall operational efficiency. This project also holds broader applications beyond traditional organizations, including educational institutions, event management, and security systems. Its relevance lies in offering a modern, secure, and efficient solution to attendance management, benefiting organizations, institutions, and users in various industries and sectors.

1.4 Objectives

The objective of the Face Recognition-Based Attendance Management System project is to develop and implement a cutting-edge solution that automates attendance tracking using facial recognition technology. The system aims to replace traditional attendance management methods with a more accurate, efficient, and secure solution.

Business Benefits Expected from the Project:

1. **Accuracy and Elimination of Errors:** The use of facial recognition technology ensures accurate and reliable attendance tracking, eliminating the potential for manual errors or fraudulent activities like buddy punching. This accuracy leads to more reliable attendance data for payroll calculations and resource planning.
2. **Time Savings:** The automation of attendance tracking eliminates the need for manual processes, such as paper registers or swipe cards. This saves time for both employees and administrative staff, allowing them to focus on more valuable tasks. Additionally, the reduction in administrative efforts reduces costs associated with manual attendance management.
3. **Enhanced Security:** Facial recognition technology provides a secure method of verifying employees' identities, reducing the risk of unauthorized access or attendance manipulation. This helps organizations maintain a secure work environment and prevent time theft or unauthorized entry.
4. **Streamlined Processes and Efficiency:** The Face Recognition-Based Attendance Management System streamlines attendance tracking processes by automating data capture and recording. This leads to improved efficiency in HR operations, enabling better resource management and optimized scheduling based on accurate attendance data.
5. **Real-time Monitoring and Reporting:** The system provides real-time monitoring of attendance, allowing administrators to view and analyze attendance data instantly. This enables timely interventions for any attendance-related issues and facilitates informed decision-making.
6. **Contactless and Hygienic Solution:** The contactless nature of face recognition technology makes it a hygienic solution, particularly in environments where physical contact may lead to the transmission of germs or viruses. This is especially relevant in post-pandemic scenarios.
7. **Scalability and Flexibility:** The system is highly scalable, making it suitable for organizations of all sizes, from small businesses to large enterprises. Additionally, it can be easily customized to cater to specific attendance management needs.
8. **User-Friendly Experience:** The simplicity and ease of use of face recognition technology make it a user-friendly system. Employees or students can quickly check-in or check-out with minimal effort, enhancing overall user satisfaction.

9. **Non-Transferable and Reliable Authentication:** Facial features are unique to each individual, making face recognition a non-transferable form of authentication. This ensures that only the rightful individuals record their attendance, reducing the risk of attendance fraud.
10. **Integration with Access Control Systems:** Face recognition-based attendance systems can be seamlessly integrated with access control systems, allowing employees or students access to specific areas based on their attendance status, enhancing security.
11. **Reduced Paper Usage:** By replacing manual attendance registers or swipe cards with a digital face recognition system, the organization contributes to environmental sustainability by reducing paper usage and waste.
12. **Compliance and Audit Trail:** The system can provide an audit trail of attendance records, ensuring compliance with attendance policies and regulations. This can be valuable during audits or legal inquiries.
13. **Remote Attendance Monitoring:** In scenarios where employees or students may work or study remotely, the system can facilitate remote attendance monitoring, ensuring accurate tracking of attendance regardless of location.
14. **Minimal Infrastructure Requirements:** Face recognition technology often requires minimal infrastructure, with the ability to utilize existing cameras or devices, reducing the need for additional hardware investments.
15. **Seamless Integration with Timekeeping and Payroll Systems:** Face recognition-based attendance systems can be integrated with timekeeping and payroll systems, automating payroll calculations based on accurate attendance data.
16. **Adaptability to Diverse Environments:** The system can be used in various settings, including schools, offices, construction sites, events, and more, showcasing its versatility and applicability across different industries.

Chapter 2 System Analysis

2.1 Introduction

A face recognition based Attendance Management System is Web applications that uses facial recognition technology to manage attendance in organizations. The system automates the process of taking attendance by capturing images of people's faces and analysing them to identify unique facial features. The system is typically composed of a camera, software, and a database. The camera captures images of individuals and sends them to the software, which analyses the images and compares them to the database. The database contains information about each person's facial features, including their name, photograph, and other relevant details. Admin of system can register staff with their details and He can also View complete details of each staff and their attendance Details .Admin can modify the details of staff. Staff will login to the system using their own I'd and password and mark their attendance with their face. Also staff can view their attendance details

2.2 Existing System

The existing system of the Face Recognition-based Attendance Management System primarily relies on traditional attendance methods, such as manual attendance registers, biometric fingerprint scanners, or swipe cards and calling out names. This system poses several challenges and limitations, which have prompted the need for a more advanced and efficient solution.

1. **Manual Processes:** The current system involves manual data entry of attendance, which is time-consuming and prone to errors. Teachers or administrators must physically mark attendance for each student or employee, leading to inefficiencies and inaccuracies in record-keeping.
2. **Biometric Scanners:** Some institutions or organizations use biometric fingerprint scanners for attendance. While this is a step up from manual methods, it still requires physical contact, and the scanners can be susceptible to wear and tear, leading to reliability issues.

3. Proxy Attendance: The existing system lacks the ability to prevent proxy attendance, where one person marks attendance on behalf of another. This compromises the accuracy and reliability of attendance records.

4. Lack of Real-time Updates: With manual or traditional biometric methods, attendance data may not be instantly available. This delays access to attendance information, making it challenging to take immediate actions or track real-time attendance trends.

5. Scalability Challenges: As the number of users increases, managing attendance manually or with basic biometric scanners becomes increasingly challenging. This limits the system's scalability for larger institutions or organizations.

6. Inflexibility: The existing system might not be able to integrate with other systems or support additional features like attendance reports, analytics, or automated notifications.

To overcome these challenges and improve the overall attendance management process, the adoption of a Face Recognition-based Attendance Management System is proposed. The new system will leverage advanced facial recognition technology to automate the attendance tracking process, eliminating the need for manual data entry or physical contact. With facial recognition, users can mark their attendance by simply facing a camera, making the process efficient, accurate, and hygienic. The system can also address the issue of proxy attendance by verifying each user's identity through facial recognition.

Furthermore, the Face Recognition-based Attendance Management System will offer real-time updates, allowing administrators and educators to access attendance data instantly. It will support scalability, making it suitable for organizations of varying sizes. Additionally, the system can integrate with existing databases and systems, enabling seamless data exchange and comprehensive attendance reporting and analytics.

2.3 Proposed System

The proposed system of Face Attendance Management System is a modern and innovative way of managing employee attendance, which utilizes facial recognition technology to automate the process of marking attendance. This system offers several advantages over the existing

attendance marking system, which typically involves manual methods such as using paper-based attendance registers or biometric devices like fingerprint scanners.

Here are some of the key advantages of the Face Attendance Management System compared to the existing attendance marking system:

- **Accuracy:** The Face Attendance Management System offers a high level of accuracy in marking attendance since it uses facial recognition technology, which is highly reliable and efficient. This system eliminates the risk of errors that can occur with manual attendance marking, such as mistakes in recording attendance data or fraudulent activities like buddy punching.
- **Speed:** The Face Attendance Recognition Based Management System is much faster than the existing attendance marking system since it only takes a few seconds to recognize and register an employee's face. This system eliminates the need for employees to wait in long queues to mark their attendance, which can be time-consuming and frustrating.
- **Convenience:** The Face Recognition Based Attendance Management System is very convenient for employees since they do not need to carry any physical attendance marking devices like smart cards or fingerprint scanners. Instead, they can simply use their face to mark their attendance, which is a natural and effortless process.
- **Security:** The Face Recognition Based Attendance Management System offers enhanced security features such as live face detection, anti-spoofing, and facial recognition algorithms that prevent fraudulent activities like using photos or videos to mark attendance. This system ensures that only authorized employees can mark their attendance, which is critical for maintaining workplace security.
- **Cost-effectiveness:** The Face Recognition Based Attendance Management System is a cost-effective solution since it eliminates the need for expensive biometric devices or paper-based attendance registers. This system only requires a webcam or camera-enabled device, which is readily available in most workplaces.

2.4 Feasibility Study

A feasibility study aims to objectively and rationally uncover the strengths and weaknesses of an existing system or proposed system, opportunities and threats present in the natural environment, the resources required to carry through, and ultimately the prospects for success. A feasibility study takes into account various constraints within which the system should be implemented and operated. In this stage, the resource needed for the implementation such as computing equipment, manpower and costs are estimated. The estimated are compared with available resources and a cost benefit analysis of the system is made. Evaluated the feasibility of the system in terms of the following categories:

- Technical Feasibility
- Economic Feasibility
- Operational Feasibility

2.4.1 Technical Feasibility

Proposed system is technically feasible since all the required tools are easily available. Technical issues involved are the necessary technology existence, technical guarantees of accuracy, reliability, ease of access, data security, and aspects of future expansion. Once the technical feasibility is established, it is important to consider the monetary factors for evaluating this, economic feasibility of the proposed system is carried out. The application is technically feasible because all the technical resources required for the development and working of the application is easily available and reliable. The project is implemented in Python. Since Python supports a various libraries and packages that make the project development easier, the project was technically feasible. The technical feasibility of a face detection-based attendance management system using the Local Binary Pattern Histogram (LBPH) algorithm is well-established. LBPH is a widely used method for facial recognition due to its simplicity, computational efficiency, and moderate accuracy. Implementing such a system involves utilizing cameras or webcams to capture images or video streams, which are then processed using the LBPH algorithm for face detection and recognition.

The LBPH algorithm, known for its speed and efficiency, enables real-time or near real-time processing of facial data, making it suitable for attendance management systems that require quick and seamless tracking of individuals entering or leaving a location. Its relatively easy implementation and low resource requirements make it accessible for a wide range of applications, from schools and offices to events and public spaces.

However, the accuracy of LBPH depends on factors such as the quality of input images, lighting conditions, and the presence of facial occlusions or variations. Thorough testing and optimization are necessary to ensure satisfactory recognition performance in specific use cases. While LBPH exhibits adaptability to some extent, it may face challenges in recognizing faces with significant occlusions or partial views.

To implement a face detection-based attendance management system using LBPH, organizations need to maintain a well-curated database of pre-registered faces. Regular updates and management of the database are crucial to accommodate changes in the user base and ensure accurate attendance tracking.

It's important to address privacy and security concerns associated with biometric data handling. Robust measures should be implemented to protect the sensitive information, and compliance with privacy regulations should be a top priority to gain user trust and ensure data integrity.

In conclusion, the technical feasibility of a face detection-based attendance management system using LBPH is evident. Its efficiency, ease of implementation, and adaptability make it a viable solution for various settings. Proper planning, testing, and attention to privacy and security aspects are essential for a successful implementation that streamlines attendance tracking processes and enhances operational efficiency.

2.4.2 Economical Feasibility

In this proposed system, all the tools used are free and open source. So that development cost is minimum. The hardware is used for run the system the microphone and the camera. It is built in our laptop. And also hardware requirement is feasible and not much Face Recognition Based Smart Attendance System maintenance is required. The development of the system will not need a huge amount of money. It will be economically feasible. And the money spend for the application will be worth. Hence, the proposed system is economically feasible.

2.4.3 Operational Feasibility

The operational feasibility of a face detection-based attendance management system using the Local Binary Pattern Histogram (LBPH) algorithm is an important consideration for its successful implementation. LBPH is widely used due to its simplicity, efficiency, and moderate accuracy in facial recognition tasks. Its computational efficiency enables real-time or near real-time face detection and recognition, making it suitable for attendance management systems that require quick and seamless tracking of individuals. However, its accuracy depends on factors such as image quality, lighting conditions, and facial occlusions, which must be thoroughly tested and optimized to meet specific use case requirements.

LBPH's ease of implementation is advantageous as it reduces development time and costs, making it accessible to a broader range of users. Additionally, its low resource requirements enable deployment on various hardware configurations, including standard computers and low-power devices. While LBPH demonstrates a certain degree of adaptability to variations in facial appearances and lighting conditions, it may encounter challenges with significant occlusions or partial views, affecting recognition accuracy in certain scenarios. Therefore, it is essential to be mindful of such limitations and consider the system's deployment environment accordingly.

For reliable recognition, the system requires a well-maintained database of pre-registered faces, necessitating regular updates and management to accommodate changes in the user base. Addressing privacy and security concerns is critical, as facial recognition systems handle sensitive biometric data. Adequate measures must be implemented to safeguard this information and comply with relevant privacy regulations.

User acceptance is a significant factor for the successful implementation of any attendance management system. Organizations must effectively communicate the benefits, address concerns, and provide adequate training to ensure users are comfortable with the technology and its use. Overall, with proper planning, thorough testing, and compliance with privacy regulations, a face detection-based attendance management system using LBPH can streamline attendance tracking processes and enhance operational efficiency in various settings.

2.5 Software Engineering Paradigm Applied

The Face Recognition-Based Attendance Management System (FRAMS) project can be developed using the Agile software engineering paradigm. Agile is a flexible and iterative

approach that emphasizes collaboration, adaptability, and customer involvement throughout the development process. Given the dynamic nature of face recognition technology and the need to address potential challenges and user requirements, Agile is well-suited for this project. Here's how the Agile paradigm can be applied to the FRAMS project:

1. **Iterative Development:** The project can be divided into small, manageable iterations or sprints, where each iteration focuses on delivering specific features or functionalities of this system user registration, training, attendance marking. This allows for incremental progress and the ability to adapt based on feedback and changing requirements.
2. **User Involvement:** Agile encourages constant communication and collaboration with end-users, in this case, administrators and employees/students, throughout the development process. Regular feedback from users helps ensure that the system meets their needs and expectations.
3. **Adaptability:** As face recognition technology evolves and user requirements may change, Agile's adaptive nature allows the development team to incorporate new developments and adjust project priorities accordingly.
4. **Continuous Testing and Quality Assurance:** Agile promotes continuous testing throughout development to identify and address issues early on. This ensures a higher-quality final product. This help efficient Marking and Tracking Attendance
5. **Cross-Functional Teams:** Agile encourages the formation of cross-functional teams comprising developers, designers, testers, and stakeholders. This promotes better collaboration and collective ownership of the project's success.
6. **Working Prototypes:** Each iteration produces a working prototype, allowing stakeholders to visualize progress and provide valuable input, which fosters greater transparency and reduces the risk of misunderstandings.
7. **Time-to-Market:** Agile's iterative approach enables quicker delivery of essential features, allowing the FRAMS project to have a faster time-to-market while continually improving and expanding its capabilities.
8. **Scope Management:** With Agile's focus on prioritizing user stories and delivering the most valuable features first, scope creep can be effectively managed to keep the project on track.

Chapter 3 System Design

3.1 Introduction

The system design for the Face Recognition-Based Attendance Management System (FRAMS) is a crucial phase that lays the foundation for the development and implementation of an efficient, accurate, and user-friendly attendance management solution. The primary objective of this system design is to create a robust and scalable framework that seamlessly integrates face recognition technology with attendance tracking functionalities. By leveraging cutting-edge algorithms and adhering to best practices, the system design aims to address the unique challenges of face recognition, ensure data security, and provide a smooth user experience.

The system design will start by defining the system's architecture, outlining its components and their interactions. This will include the user interface for registration and attendance marking, the face recognition engine responsible for user identification, the attendance database, and the reporting and analytics module. The design will emphasize modularity, allowing for easy extensibility and integration with existing systems.

Moreover, the system design will detail the face recognition model's development, focusing on selecting the appropriate algorithms for accurate face detection and feature extraction. To improve recognition accuracy and adaptability, the model will undergo rigorous training with diverse datasets to accommodate various facial expressions, poses, and lighting conditions.

Ensuring data security and privacy will be a top priority in the system design. Robust encryption and access control mechanisms will be implemented to protect facial data and user information, adhering to industry standards and data protection regulations.

Additionally, the design will highlight the system's real-time monitoring capabilities, enabling administrators to track attendance records as they are marked. Reporting functionalities will be designed to generate comprehensive attendance statistics, trends, and user-specific reports, empowering administrators with valuable insights for decision-making.

The system design phase will involve continuous collaboration with stakeholders and users, soliciting feedback to fine-tune the user interface and user experience. Regular iterative updates and prototyping will help validate design decisions and ensure that the final system aligns precisely with user needs and expectations.

In conclusion, the system design for the Face Recognition-Based Attendance Management System will serve as a detailed blueprint for the development team, guiding them in creating a reliable, secure, and efficient solution. By embracing best practices, advanced face recognition technology, and user-centric design principles, the FRAMS system design will pave the way for a transformative attendance management system that simplifies the tracking process and enhances overall organizational efficiency.

3.2 Database Design

3.2.1 Table Name: Student

Refer Annexure 1.1

3.2.2 Table Name: Attendance

Refer Annexure 1.2

3.2.3 Table Name: Issue Reporting

Refer Annexure 1.3

3.3 Object Oriented Design-UML Diagrams

Refer Annexure 2

3.3.1 Use Case Diagram

Refer Annexure 2.1

3.3.2 Activity Diagram

Refer Annexure 2.2

3.3.3 Class Diagram

Refer Annexure 2.3

3.3.4 Sequence Diagram

Refer Annexure 2.4

3.4 Modular Design

Module 1: User Registration:

- Introduction: This feature allows administrators to add new users to the system and users to register their faces for face recognition.
- Inputs: User information (e.g., name, employee ID), user photo or face image.
- Processing: Validate and store user information and face images in the system's database.
- Outputs: Confirmation message indicating successful Register.

Module 2: Face Training:

- Introduction: This feature involves training the face recognition model with enrolled user faces to improve accuracy.
- Inputs: Enrolled user face images.
- Processing: Use the collected face images to train the face recognition model, update the model parameters, and improve its ability to recognize Register users.
- Outputs: Updated face recognition model ready for attendance marking.

Module 3: Attendance Marking

- Introduction: This feature allows users to mark their attendance using the face recognition system.
- Inputs: Captured face image or video stream.

- Processing: Compare the captured face with enrolled faces in the system's database, perform face recognition, and determine user identity.
- Outputs: Confirmation of successful attendance marking, along with the recognized user's name or ID.

Module 4. Attendance Tracking

- Introduction: This feature tracks and records attendance data for all users.
- Inputs: Id and email.
- Processing: Store the attendance records in a centralized database, including date, time, and user identification.
- Outputs: Updated attendance records for individual users, enabling for tracking and reporting.

Module 5. Reporting issue:

- Introduction: This feature generates report of issue related marking attendances
- Inputs: Id and Email.
- Processing: Analyze the attendance data, generate reports based on specified criteria
- Outputs: generated a report of reason for occurring issue

Module 6. Managing User

- Introduction: This feature allows administrators to manage user accounts
- Inputs: User information, such as name, ID, Email and Department.
- Processing: Perform operations like deleting users, modifying user information,
- Outputs: Confirmation of successful user updated user information

3. 5 Design

3.5.1 Login

Refer Annexure 3.1

3.5.2 User Registration

Refer Annexure 3.2

3.5.3 Training Image

Refer Annexure 3.2

3.5.4 Marking Attendance

Refer Annexure 3.3

3.4.5 Admin Home Page

Refer Annexure 3.4

3.5.6 User Home Page

Refer Annexure 3.5

3.5.7 Tracking Attendance by admin

Refer Annexure 3.6

3.5.8 Attendance Track by User

Refer Annexure 3.7

3.5.9 Issue Reporting

Refer Annexure 3.8

3.5.10 User Profile

Refer Annexure 3.10

Chapter 4 System Environment

4.1 Software Requirements Specification

The Software Requirements Specification (SRS) for the Face Recognition-Based Attendance Management System encompasses crucial functionalities, with an added emphasis on issue reporting. This project aims to create an automated attendance tracking system leveraging advanced face recognition technology. The system will enable secure logins for administrators, teachers, and students, ensuring proper access control through unique credentials. Users' faces will be registered and securely stored in a database, linked to their respective personal information. Attendance capture will be efficient and real-time, eliminating manual tracking by swiftly recognizing individuals' faces. A robust database will manage facial data, user profiles, and attendance records, while notifications will keep teachers and administrators informed of attendance status and exceptions.

Comprehensive attendance reports and analytics will offer valuable insights into attendance trends. Seamless integration with existing hardware and management systems will be a key feature. Data privacy and security will be paramount, with stringent measures in place to protect sensitive information. The user interface will prioritize usability and accessibility, ensuring an intuitive experience for all stakeholders. Graceful error handling will provide meaningful messages for efficient troubleshooting. The system will be optimized for high performance and scalability to accommodate large user bases and data.

Comprehensive documentation and reliable technical support will be provided for easy maintenance and assistance. Additionally, the system will feature a robust issue reporting mechanism, empowering users to promptly flag and report any technical or operational problems. Administrators will have the tools to track, prioritize, and address reported issues efficiently, promoting a seamless and hassle-free user experience. Through these requirements and issue reporting capabilities, the Face Recognition-Based Attendance Management System will streamline attendance tracking, enhance organizational efficiency, and ensure a reliable and user-friendly solution for managing attendance effectively.

4.2 Tools, Platforms

4.2.1 Front End Tool: Tkinter

In the realm of modern attendance management systems, face recognition technology has emerged as an innovative and efficient solution, streamlining attendance tracking processes. One of the popular libraries used to develop user interfaces for such systems is 'tkinter'. 'Tkinter' is a Python library that provides tools to create graphical user interfaces (GUIs). By combining the power of face recognition algorithms with the user-friendly capabilities of 'tkinter', developers can craft intuitive and interactive attendance management applications.

In this face recognition-based attendance management system, 'tkinter' serves as the frontend, offering a visually appealing interface that enables users to interact with the system effortlessly. The face recognition component operates in the backend, handling the complex tasks of detecting and recognizing faces.

When a user launches the application, the 'tkinter' interface will welcome them with a clean and straightforward display. They can choose from a variety of options, such as marking their attendance, viewing attendance history, or managing their profile.

Upon selecting the "Mark Attendance" option, the webcam is activated, and the face recognition algorithm springs into action. As the user's face is detected and matched with the database of registered faces, the system records the time and date of their entry.

In case a registered user's face is not identified correctly or there are discrepancies, the system provides immediate feedback to the user through the 'tkinter' interface. This real-time feedback ensures the user can quickly resolve any issues and successfully mark their attendance.

Another essential feature of this system is its ability to handle multiple users simultaneously. Whether it's a classroom, office, or any other gathering, the face recognition algorithm efficiently processes multiple faces, accurately marking the attendance of all individuals within the frame. For administrators, the system offers a separate login through the 'tkinter' interface, granting access to attendance reports and user management options. Administrators can view comprehensive attendance records, generate reports, and add or remove users from the system with ease.

4.2.2 Back End Tool

- Python

In the Face Recognition-based Attendance Management System, Python serves as the backend, taking on a crucial role in handling data processing, database management, and communication with the front-end user interface. This choice of backend programming language allows for efficient and effective management of the system's functionalities, which include user registration, training image submission, marking and tracking attendance, managing user data, and facilitating issue reporting.

With Python handling user registration, individuals can create accounts by providing necessary information, and the backend will ensure secure storage of user data, such as usernames, passwords, and other relevant details. Additionally, during the registration process, users will be prompted to submit training images, and Python will process and store these images in the database to create unique facial recognition profiles for each user.

When it comes to attendance marking and tracking, Python utilizes the facial recognition algorithm to identify and verify users based on their submitted training images. Upon successful recognition, attendance records are seamlessly updated in the database, accurately indicating the user's presence.

Python further excels in managing user data, taking responsibility for tasks like updating user information, password reset functionalities, and handling account deletions or deactivations with ease.

Moreover, the backend plays a vital role in managing issue reporting. Users can report attendance-related issues, such as recognition failures or discrepancies, through the user interface. Python processes these reports, records them in the database, and promptly notifies administrators for further investigation and resolution.

- Mysql

A face recognition-based attendance management system, utilizing MySQL as its database, revolutionizes the process of tracking attendance in schools, colleges, offices, and other organizations. By incorporating advanced face recognition technology, this system offers a precise and secure method for attendance monitoring.

To begin, individuals are enrolled in the system by capturing their facial features, which are then converted into unique templates and stored in the MySQL database alongside relevant identification details. Subsequently, when attendance needs to be marked, individuals approach the face recognition device, and their facial images are captured. The system's sophisticated algorithms process these images and generate facial templates for comparison.

The templates are matched against the existing records in the MySQL database to identify the individual accurately. Once the recognition process is successful, attendance is promptly recorded with a timestamp. In case of any discrepancies or unrecognizable faces, exceptions are raised, necessitating manual intervention.

MySQL's real-time data processing capabilities ensure that attendance records are promptly updated, providing reliable and up-to-date information. The database efficiently manages attendance data, facilitating easy querying and analysis for generating reports, calculating attendance percentages, and detecting attendance patterns.

With MySQL's scalability, the system can effortlessly handle attendance records for organizations of any size. Its efficiency and accuracy streamline the attendance process, reducing administrative burdens and increasing overall productivity and effectiveness.

In conclusion, a face recognition-based attendance management system integrated with MySQL is a powerful solution that simplifies attendance tracking while prioritizing accuracy, security, and privacy. Its implementation marks a significant advancement in attendance management, benefiting diverse organizations in numerous ways.

4.2.3 Operating System: Windows 10

In a face recognition-based attendance management system, Windows 10 assumes a pivotal role as the operating system that underpins the entire infrastructure. Acting as the backbone of the system, Windows 10 supports the integration of hardware and software components, ensuring a seamless and efficient attendance tracking process. Its extensive compatibility with various hardware devices, including cameras and webcams, allows the system to capture facial images accurately. With robust driver support, Windows 10 ensures that these peripherals are seamlessly recognized and integrated, eliminating compatibility issues.

The user-friendly interface of Windows 10 simplifies the deployment and interaction with the face recognition software. Administrators and users can easily access attendance-related information, receive status updates, and follow intuitive instructions. Additionally, Windows

10's security features, such as user account management, login credentials, and encryption protocols, bolster the overall security of the attendance management system. This ensures the protection of sensitive attendance data and restricts access to authorized users only.

Moreover, Windows 10 supports network connectivity, enabling the system to connect to a central server or cloud-based database for real-time data processing and synchronization. The versatility of the operating system facilitates the development of sophisticated face recognition algorithms, seamlessly integrated into the attendance management system. Regular updates and technical support from Microsoft further contribute to the system's stability and reliability over time. Windows 10's capability to accommodate multiple user accounts on a single device is particularly beneficial for attendance management in diverse environments, such as schools and offices. Furthermore, its seamless integration with the MySQL database allows for efficient storage and retrieval of attendance data, ensuring the system can handle large volumes of information effectively.

Chapter 5 System Implementation

The system implementation of a face recognition-based attendance management system involves the utilization of computer vision and machine learning techniques to accurately identify and record individuals' attendance through facial recognition. This system captures real-time images of individuals, extracts unique facial features, and matches them against a pre-registered database of faces. The process involves several steps, including face detection, feature extraction, face matching, and attendance recording. The heart of the system lies in the code, which comprises algorithms for face detection using methods. The code orchestrates these processes to seamlessly recognize faces, link them to individuals, and maintain attendance records, providing an efficient and accurate solution for attendance management.

5.1 Coding

Refer Annexure 4

Chapter 6 System Testing

6.1 Introduction

Testing is a crucial phase in a face recognition-based attendance management system, ensuring its reliability, accuracy, and overall performance. The testing process involves various essential aspects. Unit testing verifies individual components or modules in isolation, such as the face recognition algorithm and database integration. Integration testing evaluates interactions between components to ensure seamless communication between the face recognition software, MySQL database, cameras, and other peripherals. Facial recognition accuracy testing focuses on evaluating the algorithm's precision and minimizing false positives or negatives. Performance testing ensures the system can handle peak loads efficiently. Security testing examines vulnerabilities and safeguards sensitive biometric data. Usability testing assesses user-friendliness, and scalability testing checks the system's capacity for managing large datasets. Stress testing assesses stability under extreme conditions, while compatibility testing ensures functionality across different devices and platforms running Windows 10. Comprehensive testing leads to a reliable and efficient attendance management system, enhancing accuracy, security, and user satisfaction throughout the attendance tracking process.

6.2 Unit Testing

Test Case ID	Test Objective	Precondition	Steps/Cases	Test Data	Expected Result	Post Condition
TC_MI_01	Successful Admin/User login to Home Page	1. A valid User account to login to be available	1. In the login Panel, enter the username "	"valid username"	user/Admin is logged in successfully.	The use is logged into the system and has access to thappropriate resources and features based on their user role and permissions.

		2. The application is launched on a compatible browser	2. Enter the Password	"valid Password "		
			3. Click "Login" button			
TC_MI_O 2	To Successfully register user into the system	1. Admin account to login to be available 2. The application is launched on a compatible browser for Registering User	1. Log in to the system using valid credentials 2. Click on "User Registration" and Enter the details of user 3. Click On "Submit"	" valid username and Password " "valid User Details"	User Successfully Register into system	The user's information and facial image are stored in the system.
TC_MI_O 3	To Successfully Train User	1. Admin account to	1. Log in to the system	"Valid username	Admin Successfully	The user's image Image is trained and can be

	Image	login to be available	using valid credentials	and Password”	train user image into system	successfully recognized during attendance marking.
		2. The application is launched on a compatible browser for Training User Image	2.Click on “Training Image”	“Valid User Image”		
TC_MI_O 4	To Successfully Mark Attendances	1. The application is launched on a compatible browser for Marking Attendances	1.Click on “Marking Attendance”	“Valid User Image”	User Successfully Mark Attendances	The attendance is successfully marked and recorded in the system for further tracking and analysis.
			2.Camera Open and Scan The face			
TC_MI_O 5	To Successfully Track or view Attendances	1.Admin/User account to login to be available	1.Log in to the system using valid credentials	“Valid username and Password”	User /Admin Successfully Track or view Attendances	The attendance records are tracked and displayed accurately, enabling effective

		2. The application is launched on a compatible browser for Tracking Attendances	2. Click on "Tracking Attendance"	"Valid User Details"		attendance monitoring and analysis.
TC_MI_O 6	To Successfully Manage User	1. Admin account to login to be available 2. The application is launched on a compatible browser for Managing User	1. Log in to the system using valid credentials 2. Click on the "Edit" button and Enter a enter the details of user or Click On "Delete button" to remove user from system 3. click on "submit"	" valid User name and password " " Valid User details"	Successfully Manage User Details	The user's information should be updated and stored accurately in the system
TC_MI_O 7	To Successfully Report Issue	1. Admin account to login to be available	1. Log in to the system using valid credentials	" valid User name and password "	Successfully Report issue	The issue details should be stored accurately in the system and

		2. The application is launched on a compatible browser for Managing User	2. Click on the “Issue report and enter the issue details” 3. Click on “submit”	“ Valid Issue details”		solve reported issue
--	--	--	--	------------------------	--	----------------------

Test Plan

In unit testing for the face recognition-based attendance management system, the focus is on thoroughly testing individual components or units in isolation to ensure their functionality and accuracy. Here's a test plan for unit testing in this project:

1. Face Recognition Algorithm:

- Test the algorithm with a variety of facial images, including different lighting conditions, facial expressions, and angles.
- Verify that the algorithm correctly identifies enrolled individuals and minimizes false positives or false negatives.
- Test the algorithm's response to corrupted or incomplete facial images to ensure proper error handling.

2. MySQL Database Integration:

- Test the database connectivity and verify that the system can establish a connection to the MySQL database.
- Validate that attendance records are accurately stored and retrieved from the database during test scenarios.
- Verify error handling when the database connection is unavailable or if there are any data-related issues.

3. Camera and Webcam Integration:

- Test the integration of different camera models and webcams to ensure compatibility and proper functioning.
- Check that the system can capture facial images from the cameras and webcams consistently.
- Validate the image quality and resolution for accurate face recognition.

4. Network Connectivity:

- Test the system's ability to communicate with a central server or cloud-based database over a network connection.
- Verify that data synchronization occurs correctly in real-time scenarios.
- Validate the system's behavior during network disruptions or unstable connections.

5. Exception Handling:

- Test the system's response to exceptional scenarios, such as unrecognized faces or invalid inputs.
- Verify that appropriate error messages are displayed, and the system gracefully handles unexpected situations.

6. Usability:

- Perform usability testing with different users to assess the system's ease of use and understandability.

- Gather feedback on the user interface and make necessary improvements for a seamless user experience.

7. Code Coverage:

- Measure code coverage to ensure that all parts of the codebase have been adequately tested.
- Aim for high code coverage to minimize the chances of undiscovered bugs.

6.3 Integration Testing

The integration testing procedure aims to validate the interaction and integration between different components of the face recognition based attendance management system. It ensures that the system functions correctly as a whole and that the integrated components work together seamlessly. The following sections outline the testing procedure, test cases, and their expected result

Testing procedure for integration

1. Identify System Components: Determine the key system components that need to be integrated for testing. This may include modules or subsystems such as user registration, face recognition, user management, attendance marking, attendance tracking, and issue reporting.
2. Define Integration Approach: In this project, the integration approach will be a combination of both top-down and bottom-up integration.

Top-down integration involves starting with higher-level components and gradually integrating lower-level components. In this case, the integration may begin with the user interface and user management components, followed by the face recognition and attendance tracking components. This approach allows for early testing of the user-facing functionalities and ensures that the overall system behavior is validated from the top-level down to the underlying components. Bottom-up integration, on the other hand, starts with the lower-level components and gradually integrates higher-level components. In this project, it may involve initially integrating and testing the face recognition and attendance marking components, and then gradually integrating the user management and user interface components. This approach allows for early identification of issues at the component level and ensures the integration of key functionalities before reaching the higher-level components.

3. Create Integration Test Environment: Set up a test environment that replicates the real-world deployment environment. This includes configuring hardware like camera, software, and network components necessary for integration testing.

4. Develop Integration Test Cases: Design test cases that focus on the interaction and communication between integrated components. Test cases should cover various integration scenarios, including different data flows, error handling, and system boundaries. In the context of this project, the development of integration test cases is crucial to validate the interaction and communication between integrated components. These test cases should encompass a range of integration scenarios, including different data flows, error handling during training-marking attendance, and system boundaries.

Integration test cases should be designed to assess how the components work together and verify that the integration is functioning correctly. This involves creating test cases that simulate various data inputs, evaluating the accuracy of data transfer between components, and assessing the system's ability to handle error conditions and exceptions. By developing comprehensive integration test cases, the project team can identify any issues or gaps in the integration process, validate the smooth flow of data between components, and ensure that the system behaves as expected during integrated operations. This testing approach helps enhance the reliability, functionality, and overall quality of the face recognition-based attendance management system.

5. Execute Integration Test Cases: Execute the integration test cases, following the defined integration approach. Monitor the interactions between components and verify that data is correctly passed between them. Capture any errors or unexpected behavior during integration.

6. Record Test Results: During the integration testing phase of the project, it is essential to meticulously record the results of each integration test case. The test results should include comprehensive documentation of any discrepancies, failures, or issues encountered during testing. This involves capturing relevant details such as error messages, logs, and test environment configurations. The test results documentation should provide a clear and concise overview of the outcomes of each integration test case. It should highlight any deviations from expected behavior, errors that occurred during the testing process, and any inconsistencies or unexpected results observed. Overall, maintaining a detailed record of the integration test results contributes to effective issue tracking, efficient bug fixing, and ultimately, the successful delivery of a high-quality face recognition-based attendance management system.

7. Debug and Resolve Issues: Analyze the identified issues, troubleshoot the causes, and determine appropriate resolutions. Collaborate with the development team to fix defects or make necessary adjustments to ensure proper integration.
8. Retest and Validate Fixes: Once fixes or adjustments are implemented, rerun the affected integration test cases to validate that the issues have been resolved and the components integrate successfully.
9. Repeat Steps 5-8: Repeat the execution, recording, debugging, and resolution process for all remaining integration test cases until all specified integration scenarios have been tested.
10. Finalize Integration Testing: Evaluate the overall integration test results, review the test documentation, and ensure that all integration requirements and objectives have been met. Prepare a summary report of the integration testing process, including any recommendations for further improvements or additional testing if needed.

Test Cases and Their Purpose:

1. Test Case: Integration of User Registration with Face Recognition

Purpose: Verify that user registration data seamlessly integrates with the face recognition component, allowing new users to be enrolled for attendance tracking .

2. Test Case: Integration of User Management with Attendance Tracking

Purpose: Ensure that user management functions, such as adding or removing users, seamlessly integrate with attendance tracking, and reflect changes accurately.

3. Test Case: Integration of Attendance Marking with Face Recognition

Purpose: Validate that attendance marking using face recognition integrates smoothly, ensuring that attendance records are accurately captured and associated with the correct users.

4. Test Case: Integration of Attendance Tracking with Issue Reporting

Purpose: Verify that attendance tracking integrates effectively with the issue reporting functionality, allowing any attendance-related issues or discrepancies to be reported and logged correctly.

Expected Results:

1. User Registration Integration:

- New user registration data successfully passed to the face recognition component.
- Enrolled users can be recognized and identified by the face recognition system.

2. User Management Integration:

- Adding or removing users in the user management component reflects changes accurately in the attendance tracking system.
- User roles and permissions are correctly applied in attendance-related processes.

3. Attendance Marking Integration:

- Face recognition-based attendance marking correctly associates attendance records with the respective users.
- Attendance data is accurately captured and stored in the attendance tracking component.

4. Attendance Tracking Integration:

- Attendance records are accurately tracked and maintained over time.
- Attendance reports reflect correct attendance calculations, summaries, and historical data.

5. Issue Reporting Integration:

- Attendance-related issues reported through the system are logged correctly.
- Reported issues are appropriately assigned and resolved within the issue reporting component.

6.4 System Testing

The system testing procedure for this project involves specific tests to evaluate the overall performance and functionality of the attendance marking system. The following sections outline the specific system tests, including the test procedures, test cases, purposes, specialized requirements, and pass/fail criteria:

Test Plan: Face Recognition-based Attendance Management System

1.Introduction:

The purpose of this test plan is to outline the testing approach and procedures for the Face Recognition-based Attendance Management System. The system utilizes face recognition technology to register users, manage attendance records, and track attendance data. The testing process will include System Integration Testing, Functional Testing, Usability Testing, Performance Testing, and Security Testing.

2. Objectives:

The primary objectives of testing the Face Recognition-based Attendance Management System are as follows:

- Validate the accuracy and effectiveness of the face recognition technology in registering users and marking attendance.
- Ensure all functionalities, such as user registration, attendance marking, and attendance tracking, work as intended.
- Evaluate the system's usability and user-friendliness.
- Measure the system's performance under normal and peak usage scenarios.
- Verify the system's security measures to protect sensitive user data.

3. Test Environment:

The test environment for the Face Recognition-based Attendance Management System will consist of the following:

- Operating System: Windows 10
- Web Browser: Google Chrome, Mozilla Firefox
- Database: MySQL 8.0
- Face Recognition Library: OpenCV

4. Test Procedures:

- Functional Testing:
- Procedure:

- Test all functional aspects of the system, including user registration, attendance marking, and attendance tracking functionalities.

- Test Cases:

- a) User Registration:

- Verify that users can register successfully with valid details.
 - Check for error handling when invalid data is submitted during registration.

- b) Attendance Marking:

- Test face recognition accuracy in marking attendance for registered users.
 - Validate handling of unknown faces or unrecognized users.

- c) Attendance Tracking:

- Verify that the system accurately displays and updates attendance records.
 - Check filtering and sorting options for attendance data.

- Pass/Fail Criteria:

- All functionalities should work as intended without any critical errors.

- Usability Testing:

- Procedure:

- Assess the system's user-friendliness and ease of navigation for both administrators and end-users.

- Test Cases:

- a) User Interface Evaluation:

- Evaluate the user interface design for clarity and ease of use.

- b) User Registration Workflow:

- Test the registration process and assess its intuitiveness.

- c) Attendance Marking Process:

- Evaluate the process of marking attendance for users.

- Pass/Fail Criteria:

- The system should have an intuitive user interface and smooth user workflows.

- Performance Testing:

- Procedure:

- Measure the system's performance under normal usage and peak load conditions.

- Test Cases:

a) Normal Load Performance:

- Evaluate the response times for user registration and attendance marking under typical usage.

b) Peak Load Performance:

- Test the system's response times and resource utilization under a high number of concurrent users.

- Pass/Fail Criteria:

- The system's response times should meet predefined performance targets under both normal and peak load conditions.

- Security Testing:

- Procedure:

- Assess the system's security measures to protect user data and prevent unauthorized access.

- Test Cases:

a) Data Protection:

- Verify that user data and attendance records are appropriately encrypted during storage and transmission.

b) Access Control:

- Test the system's access control mechanisms to ensure users can only access authorized functionalities.

- Pass/Fail Criteria:

- The system should pass security tests without any critical vulnerabilities or unauthorized access.

- Compatibility Testing:

- Procedure:

- Test the system's compatibility with various devices, browsers, and operating systems.

- Test Cases:

- a) Device Compatibility:

- Verify that the system functions correctly on different devices (e.g., smartphones, tablets, desktops).

- b) Browser Compatibility:

- Test the system across multiple browsers (e.g., Chrome, Firefox, Safari, Edge) to ensure consistent performance.

- Pass/Fail Criteria:

- The system should be compatible with a wide range of devices and browsers without significant functionality issues.

Chapter 7 System Maintenance

7.1 Introduction

System maintenance in the Face Recognition-based Attendance Management System is crucial to ensure the system's continuous and reliable operation. Regular maintenance activities involve monitoring the health and performance of the face recognition components, as well as other system functionalities. Software updates and patches are applied promptly to keep the system up-to-date and secure against potential vulnerabilities. Database backups are taken regularly to safeguard attendance records and user data, ensuring data integrity and availability. Temporary files are cleared to optimize system resources and prevent any unwanted accumulation of data. Database queries are optimized to enhance the system's performance, enabling faster attendance marking and data retrieval.

Security maintenance is a top priority in this system. System administrators must stay vigilant about the latest security threats and apply necessary security measures to protect sensitive user data and prevent unauthorized access. Regular security audits and vulnerability assessments are conducted to identify and address any potential security weaknesses.

Routine testing and monitoring are essential components of system maintenance. Performance testing is conducted to measure the system's response times and resource utilization under various loads. This helps identify any performance bottlenecks and ensures the system can handle a high number of concurrent users during peak times. Additionally, usability testing is performed to evaluate the user interface and workflows, ensuring a user-friendly experience for both administrators and end-users. Any reported issues or anomalies are promptly addressed to maintain a seamless user experience.

7.2 Maintenance

1. **Maintenance Documentation:** Comprehensive maintenance documentation will be created, including detailed information about the face recognition-based attendance management system's architecture, configuration settings, software components, integration points, and functionality like user registration, attendances marking etc. This documentation will serve as a reference for future maintenance activities, allowing maintainers to understand the system's structure and effectively manage it.

2. **System Monitoring:** A system monitoring plan will be established to continuously monitor the performance to track attendances details, availability of system for marking attendance ,and security of the system. Monitoring tools will be implemented to track system metrics, set up alerts for potential issues, and regularly review system logs. This proactive monitoring approach helps detect and address any anomalies or problems promptly.
3. **Bug Tracking and Issue Resolution:** A bug tracking system or issue tracking tool will be used to capture and manage reported issues or bugs. A streamlined process will be established to ensure that issues are properly documented, prioritized, assigned to relevant team members, and resolved in a timely manner. This helps maintainers efficiently track and resolve reported issues, improving the system's overall stability.
4. **Software Updates and Patches:** A plan will be outlined for managing software updates and patches. This includes staying informed about the latest updates related to the system's software components, operating systems, libraries, and frameworks. Regular updates and patches will be applied to address security vulnerabilities, enhance system performance, and incorporate new features or bug fixes.
5. **Hardware Maintenance:** Guidelines will be provided for hardware maintenance specifically focusing on the camera used in the system. These guidelines may include instructions for cleaning, calibration, and troubleshooting common hardware issues. Regular maintenance checks and inspections will be recommended to ensure the camera's optimal performance and reliability.
6. **Training for Future Maintenance:** Training sessions or materials will be developed to educate future maintainers about the system's architecture, functionality, and maintenance processes. This training will enable them to effectively manage the system, understand its components, troubleshoot issues, and perform necessary updates or modifications. By providing proper training, the student ensures that future maintainers have the necessary skills and knowledge to maintain and support the face recognition-based attendance management system.

Chapter 8 System Security Measures

8.1 Introduction

System security is a critical aspect of the Face Recognition-based Attendance Management System. This chapter outlines the security measures implemented at various levels to protect sensitive data, prevent unauthorized access, and ensure the system's integrity and confidentiality.

8.2 Operating System-Level Security:

At the operating system level, several security measures are implemented to safeguard the server and its resources.

User Access Control:

User access control is enforced to limit administrative privileges to authorized personnel only. Each user is assigned specific roles and permissions based on their responsibilities within the attendance system.

Firewall Configuration:

A firewall is set up to control incoming and outgoing network traffic. The firewall rules are carefully configured to allow only necessary connections, blocking potential malicious attempts.

System Updates and Patches:

Regular updates and patches are applied to the operating system to address known vulnerabilities and enhance security. Automated update mechanisms are employed to ensure timely updates.

Intrusion Detection System (IDS):

An Intrusion Detection System is deployed to monitor the system for any suspicious activities or unauthorized access attempts to mark attendance. Any detected anomalies trigger immediate alerts for investigation.

File and Directory Permissions:

File and directory permissions are set appropriately to restrict access to sensitive files and directories. Only authorized users can access, modify, or execute critical system files. Here admin can only do it.

8.3 Database Level Security:

Database security in a Face Recognition Attendance Management System is a critical aspect that ensures the protection, confidentiality, and integrity of the stored attendance and user data. As the system relies on facial recognition technology to identify and authenticate users, the database becomes a repository of sensitive information. To safeguard this data, robust security measures must be implemented. This includes employing strong encryption techniques to prevent unauthorized access to the database, implementing access controls to restrict user privileges, and regularly auditing and monitoring database activities for any suspicious behavior. Additionally, measures like data backups and disaster recovery plans are essential to ensure data availability and continuity in case of unforeseen events. By prioritizing database security, the attendance management system can maintain user trust and compliance with privacy regulations, mitigating potential risks associated with data breaches or unauthorized data access.

Data security at the database level is of utmost importance to protect attendance records and user information.

Data Encryption:

Sensitive data, such as attendance records and user details, is encrypted at rest in the database. Encryption algorithms are employed to ensure data confidentiality, even if unauthorized access occurs.

Role-Based Access Control (RBAC):

RBAC is implemented to control access to the database. Each user is assigned specific roles with predefined privileges, limiting their actions to authorized data.

Database Backups and Recovery:

Regular database backups are taken to ensure data availability and integrity. In the event of data loss or corruption, the system can be restored to a previous state.

Parameterized Queries:

To prevent SQL injection attacks, parameterized queries are utilized when interacting with the database. This protects against malicious attempts to manipulate the database.

Audit Trail:

An audit trail is maintained to log and monitor all database activities. This helps track changes made to the database and aids in identifying any unauthorized modifications.

Chapter 9 System Planning and Scheduling

9.1 Introduction

In this section, we will discuss the system planning and scheduling for the development of a face recognition system, starting from May 24th and concluding on July 15th. The development process will follow the standard software development life cycle, which includes system study, system analysis, design, coding and testing, and implementation.

System Planning and Scheduling:

1. System Study (May 24 – June 8):

- During this phase, the project team will conduct a thorough study to understand the requirements and objectives of the face recognition system. They will identify the key stakeholders, gather user expectations, and assess the existing systems, if any, to determine the system's scope and feasibility. By the end of this phase, the team will produce a comprehensive system study report.

2. System Analysis (June 9 - June 15):

- In the system analysis phase, the team will analyze the requirements gathered during the system study. They will identify the various functional like marking and tracking attendance, registration etc. and non-functional requirements like performance etc. , define use cases, and create data flow diagrams to understand the system's architecture and flow of information. The team will also perform risk analysis and develop a detailed system analysis report.

3. Design (June 16 - June 30):

- During the design phase, the team will create a blueprint for the face recognition system. They will develop system architecture, design database schemas, and prepare mockups for the user interface. The team will also decide on the technologies, frameworks, and (lbph)to be used in the face recognition system. At the end of this phase, the team will have a comprehensive design document.

4. Coding and Testing (July 1 - July 8):

- In the coding and testing phase, the actual development of the face recognition system will take place. The team will write code based on the design specifications and implement the

chosen algorithms(Lbph). As development progresses, continuous testing will be carried out to identify and fix any bugs or issues. Unit testing, integration testing, and system testing will be conducted to ensure the system's functionality like marking attendance and reliability.

5. Implementation (July 9 - July 19):

- The implementation phase involves deploying the face recognition system in a production environment. The team will conduct user training sessions and provide documentation to facilitate smooth adoption by end-users. They will also set up necessary servers, configure the system, and conduct performance tests. Once all aspects are validated, the face recognition system will be officially launched for operational use.

9.2 GANNT Chart

A Gantt chart is a visual project management tool used to plan, schedule, and track tasks and activities in a project. It provides a clear and easy-to-understand representation of project timelines, tasks, and their interdependencies. The chart consists of horizontal bars representing each task, with the length of the bars corresponding to the task's duration. Gantt charts typically display the tasks along the vertical axis and the timeline along the horizontal axis.

Each task on the Gantt chart is associated with a start date and an end date, allowing project managers and team members to visualize the project's progress over time. Tasks can be organized into phases, and dependencies between tasks can be shown using links or arrows, indicating the order in which tasks need to be completed.

Tasks	Duration	June				July			
Days		1	2	3	4	1	2	3	4
System study	2 weeks								
System analysis	1 weeks								
Design	2 weeks								
Coding and testing	1 weeks								
Implementation	2 weeks								

Chapter 10 System Cost Estimation

10.1 Introduction

In this chapter on the System Cost Estimation of a Face Recognition-based Attendance Management System, we delve into estimating the expenses involved in developing and implementing such a system. The primary objective of this system is to streamline attendance tracking processes for institutions, organizations, or businesses by leveraging facial recognition technology. By eliminating traditional attendance methods and allowing users to mark their attendance through facial scans, this system aims to enhance efficiency and accuracy.

The cost estimation process is critical for effective project planning, as it helps identify potential expenses, allocate resources, and ensure the project's financial feasibility. We outline the costs in various categories, including hardware like camera , software, personnel, implementation, maintenance, and support. Under hardware costs, we consider items such as cameras, servers, storage like database, and networking equipment, as well as any additional facial recognition devices required. Moving on to software costs, we include expenses related to licensing face recognition algorithms, attendance management software, and database management systems. Personnel costs encompass the development team, project management, testing, and quality assurance, all of which are crucial for the successful execution of the project.

Implementation costs involve integrating the system with existing setups, user face training, and data migration, if necessary. Maintenance and support costs account for system updates, upgrades, technical support, and troubleshooting.

To optimize the budget, we explore cost-saving measures, such as considering open-source alternatives, utilizing cloud services and hosting options, and ensuring optimal resource allocation. We also address scalability and future considerations, such as accommodating a growing user base, incorporating additional features, and projecting costs for future upgrades. By the end of this chapter, stakeholders will have a comprehensive understanding of the financial requirements for the Face Recognition-based Attendance Management System. Armed with this information, they can make informed decisions, enhance cost-effectiveness, and plan for the system's sustainable and successful implementation in the long run.

10.2 LOC Based Estimation / Function Point based Estimation

LOC-based estimation involves counting the lines of code in a software project to estimate its size and potential complexity. However, it's essential to understand that LOC-based estimation has its limitations and may not always reflect the actual effort or complexity of a project accurately. The number of lines of code can vary depending on the programming language, coding style, and implementation approach, and it doesn't directly correlate with the overall project effort or quality.

1. Login Functionality:

The login functionality typically involves the code related to user authentication and session management. It includes components like login validation, password encryption,. For a basic login functionality, the LOC estimate might range from 900 to 1000 lines of code, depending on the implementation complexity.

2. Attendance Functionality:

The attendance functionality encompasses features to mark attendance and store attendance records, and generate attendance reports. This functionality could be moderately complex, involving database interactions, and report generation. The LOC estimate for the attendance module might range from 300 to 500 lines of code.

3. User Management Functionality:

User management functionality includes code related to viewing, editing, and deleting user accounts, as well as handling user permissions and roles. It may also involve features like password reset, profile settings, and user-related data handling. The LOC estimate for user management could range from 300 to 500 lines of code.

4. Issue Management Functionality:

The issue management functionality deals with handling and resolving user-reported issues. It involves components such as creating, assigning issues. Depending on the complexity of the issue tracking system, the LOC estimate for this functionality might range from 200 to 300 lines of code.

5. Training Functionality:

This Functionality include training of user face. the LOC estimate for this functionality might range from 50 to 100 lines of code.

Function Point (FP) based estimation is a technique used to estimate the size and complexity of software projects based on the functionalities and requirements of the system. In the context of the Face Recognition-based Attendance Management System, we will use FP analysis to estimate the effort and resources required for the system development.

Chapter 11 Conclusion

In conclusion, the implementation of a face recognition-based attendance management system marks a significant stride towards efficient and secure attendance marking and tracking. By leveraging advanced facial recognition technology, this system offers numerous benefits, such as eliminating the need for manual attendance taking, reducing errors associated with traditional methods, and enhancing overall administrative productivity. Furthermore, it enhances security by accurately verifying the identity of individuals through unique facial features, minimizing the possibility of proxy attendance or unauthorized access to user data. While the system does present technical challenges and privacy concerns that must be addressed, advancements in biometric technology and data protection measures are continuously refining its effectiveness and reliability. As organizations strive for streamlined operations and data-driven decision-making, the face recognition-based attendance management system stands as a testament to the potential of innovative solutions in reshaping conventional practices.

Chapter 12

Future Enhancement and Scope of further Development

12.1 Introduction

Face recognition-based attendance management systems have seen remarkable advancements, providing accurate and efficient solutions for attendance tracking in various settings. Looking ahead, the scope for further development is vast, with potential enhancements to enhance accuracy, security, and user experience. One area of focus is multi-modal biometric integration, combining face recognition with other biometric modalities like fingerprint or iris recognition for more robust authentication. Additionally, 3D face recognition technology can improve accuracy by overcoming variations in lighting and pose. Real-time emotion recognition can offer insights into individuals' emotional states, while masked face recognition addresses the need to identify individuals with partially covered faces, especially in post-pandemic scenarios. Strengthening privacy and security measures, embracing edge computing for improved efficiency, and implementing continuous authentication are essential aspects to consider. The integration of AI-based adaptive learning can enhance system accuracy, particularly in challenging environments. The scope of development also extends to scalability to handle growing user bases and attendance records, cross-platform compatibility, and integration with existing management systems. Attendance analytics can provide valuable insights for data-driven decision-making, and customization options ensure adaptability to specific requirements. Ethical considerations, such as privacy, consent, and fairness, will remain a crucial focus for developers as they strive to revolutionize attendance management across diverse applications. By addressing these aspects, face recognition-based attendance management systems will continue to be a vital and transformative tool for organizations and institutions in the future.

12.2 Merits of the System

The face recognition-based attendance management system offers several merits that make it a compelling solution for organizations and institutions:

1. Accuracy: Face recognition technology provides a high level of accuracy in identifying and verifying individuals, significantly reducing the chances of errors in attendance tracking. This ensures that attendance records are reliable and trustworthy.
2. Efficiency: The system automates the attendance process, eliminating the need for manual tracking and paper-based registers. This saves time and resources for administrators, teachers, or event organizers, enabling them to focus on more critical tasks.
3. Contactless Solution: The system is contactless, making it particularly relevant in the context of the COVID-19 pandemic and other contagious diseases. It eliminates the need for physical contact with attendance devices, reducing the risk of transmission.
4. Convenience: Users find the system convenient as they can quickly check-in or check-out by simply presenting their faces to the camera. This ease of use encourages higher compliance with attendance procedures.
5. Real-time Tracking: The system provides real-time attendance tracking, enabling administrators to access up-to-date information on attendance patterns and address any attendance-related issues promptly.
6. Scalability: Face recognition systems can handle a large number of users simultaneously, making them suitable for deployment in organizations of various sizes, from small businesses to large educational institutions or corporations.
7. Improved Security: The biometric nature of face recognition enhances security, as it is difficult to forge or manipulate facial features. This reduces the risk of proxy attendance and ensures that only authorized individuals can record their attendance.
8. Non-Intrusive: Unlike other biometric systems that require physical contact, such as fingerprint scanners, face recognition is non-intrusive and can be less intimidating or uncomfortable for users.
9. Integration Possibilities: The system can easily integrate with other management systems, such as payroll, HR, or student information systems, streamlining administrative processes and data management.

10. **Data Insights:** By collecting and analyzing attendance data, institutions and organizations can gain valuable insights into attendance patterns, trends, and correlations, helping them make informed decisions to improve efficiency and productivity.

11. **Reduced Administrative Burden:** With automated attendance tracking, the system lightens the administrative burden on educators, HR personnel, or event organizers, allowing them to focus on more strategic tasks.

12. **Versatility:** The system's versatility allows it to be deployed in various environments, such as schools, colleges, universities, corporate offices, events, or government agencies, catering to diverse attendance management needs.

In conclusion, the face recognition-based attendance management system offers a range of merits, including accuracy, efficiency, contactless operation, convenience, real-time tracking, scalability, enhanced security, and integration possibilities. These advantages make it an attractive and reliable solution for modern attendance management requirements.

12.3 Limitations of the System

- Despite its merits, the face recognition-based attendance management system also has some limitations and challenges:
- **Privacy Concerns:** The use of biometric data, such as facial features, raises privacy concerns as it involves collecting and storing sensitive information about individuals. Ensuring robust data protection and complying with relevant privacy regulations is crucial.
- **Ethical Considerations:** There are ethical considerations related to consent and user awareness regarding the use of biometric data for attendance tracking. Transparent communication and obtaining proper consent from users are essential to maintain ethical standards.
- **Accuracy and False Positives/Negatives:** While face recognition technology has improved significantly, it is not entirely immune to errors. Variations in lighting, facial expressions, or changes in appearance can lead to false positives or negatives, impacting attendance records.
- **Light and Pose Variations:** The system's accuracy may be affected by challenging lighting conditions or individuals presenting their faces from different angles or poses, leading to potential identification errors.

- Masked Faces: Face recognition struggles with identifying individuals with masks or face coverings, which may be common in certain settings or post-pandemic scenarios.
- Implementation Costs: Deploying a face recognition-based attendance management system can be costly, involving expenses related to hardware, software, and maintenance. This can be a barrier for smaller organizations or institutions with limited budgets.
- Network Dependence: Cloud-based face recognition systems rely on internet connectivity for real-time processing and storage, making them susceptible to disruptions in the network.
- Vulnerability to Attacks: Face recognition systems are not immune to potential cyber-attacks, such as spoofing or presentation attacks, where adversaries attempt to deceive the system by using photos or videos to impersonate legitimate users.
- User Acceptance: Not all individuals may feel comfortable with the use of biometric technology, leading to reluctance in adopting the system or potential pushback from users.
- Environmental Factors: External factors like weather conditions or environmental disturbances can impact the system's performance, especially in outdoor settings.
- System Bias and Accuracy Disparities: The accuracy of face recognition systems can vary based on the vendor, algorithm, or data used for training, leading to disparities in performance across different systems.

12 .4 Future Enhancement of the System

The future enhancements of the face recognition-based attendance management system aim to overcome current limitations and further improve its functionality. Here are some potential future enhancements:

1. Masked Face Recognition: As face masks become a common accessory in certain settings, developing algorithms that can accurately recognize individuals with partially covered faces will be crucial for maintaining system effectiveness.
2. 3D Face Recognition: Integrating 3D face recognition technology using depth-sensing cameras or structured light can enhance accuracy by capturing facial features from multiple angles, overcoming challenges related to pose variations.

3. Multi-Modal Biometrics: Combining face recognition with other biometric modalities, such as fingerprint or iris recognition, can create a more robust and secure authentication system, reducing the risk of false positives or negatives.
4. Emotion Recognition: Integrating real-time emotion recognition capabilities into the system can provide valuable insights into individuals' emotional states, enabling applications in education, employee well-being, and other areas.
5. Continuous Authentication: Implementing continuous authentication using behavioral biometrics, such as keystroke dynamics or gait analysis, can ensure that the authorized person remains present during the attendance duration, enhancing security.
6. Edge Computing: Leveraging edge computing to perform face recognition processing locally on devices or cameras can reduce latency, enhance privacy, and decrease dependence on cloud infrastructure.
7. Addressing Bias: Research and development efforts should focus on reducing algorithmic bias and ensuring that the system's accuracy is consistent across diverse demographic groups.
8. Privacy and Data Protection: Enhancing privacy features and adopting privacy-preserving algorithms will be crucial to safeguarding user data and complying with evolving data protection regulations.
9. Environmental Adaptability: Improving the system's ability to handle challenging lighting conditions, varying environmental factors, and outdoor settings will enhance its performance and reliability.
10. User Interface and Experience: Investing in user-friendly interfaces and seamless integration with existing systems will encourage greater user acceptance and adoption of the system.
11. Real-time Analytics: Developing real-time attendance analytics can provide valuable insights into attendance trends, enabling organizations to make data-driven decisions and optimize operations.
12. Customization Options: Offering customization options to tailor the system to specific organizational needs and requirements will increase its versatility and adaptability.

13. Integration with IoT Devices: Integrating the system with Internet of Things (IoT) devices can expand its applications and allow for attendance tracking in new and innovative ways.

14. AI-Driven Self-Learning: Implementing artificial intelligence to continuously learn from attendance data can improve the system's accuracy over time, especially in dynamic environments.

15. Cross-Platform Compatibility: Ensuring compatibility with various devices and operating systems will make the system accessible and widely applicable in different settings.

Chapter 13 Annexure

13.1 Organization profile

Signature Resource Hub is a dynamic organization dedicated to the development of Industrial and Academic projects in various streams of technology. Signature Resource Hub was established in 2016 by a group of engineers with a passion for technology makeovers and a vision to build technology enterprises. We are the premier provider of Industrial training, Training of IT aspirates, Industrial and Technical assistance, Placement training, Research, and Development across Kerala.

Our organization has a well-trained and motivated talent pool working cohesively to deliver solutions based on 10 years of experience in the field of technology. The main attraction of our institution is its top expert and experienced masters meticulously chosen to maintain the best standards. For the past 10 years, we have been preparing our major clients including students and faculties from different colleges including premium engineering colleges and IT start-ups.

Regardless of Industry, our wide enterprise in product design, Software development. Technical and support make us a partner of choice for challenging projects. One strength can value lie in one ability to help you succeed by following your idea, your needs, and your strategic directions.

Address:

Name : Signature Resource Hub

Address : Opp. Bharat Petrol Pump, Aramanapady,

Near Post Office Junction,

SH 1, MC Rd, Thottumkalpeedika,

Muvattupuzha, Kerala 686661

13.2 Annexure 1

Tables

1.1 Table Name: Student

Purpose : To Store User details

SI NO.	Field Name	Type	Constraints	Description
1	Id	Int	Primary Key	To Store User Id
2	Name	varchar(20)	Not Null	To Store Name of User
3	Password	Int	Not Null	To Store Password Of user
4	Email	Varchar	Not Null	To Store Email of User
5	Department	Varchar(20)	Not Null	To Store Department name
6	Image	Varchar(20)	Not Null	To store The Image of User

1.2 Table Name: Attendance

Purpose : To Store Attendance details

SI NO.	Field Name	Type	Constraints	Description
1	Id	Int	Foreign key	To Store User Id
2	Name	varchar(20)	Not Null	To Store Name of User
3	Attendance Id	Int	Primary Key	To Store id attendance of each user
4	Time	Time	Not Null	To Store Time which Attendance Marked
5	Department	Varchar(20)	Not Null	To Store Department name
6	Date	Varchar(20)	Not Null	To Store Date of Attendance Marked
7	status	varchar(20)	Not Null	To store Status of attendance

1.3 Table Name: Issue Reporting

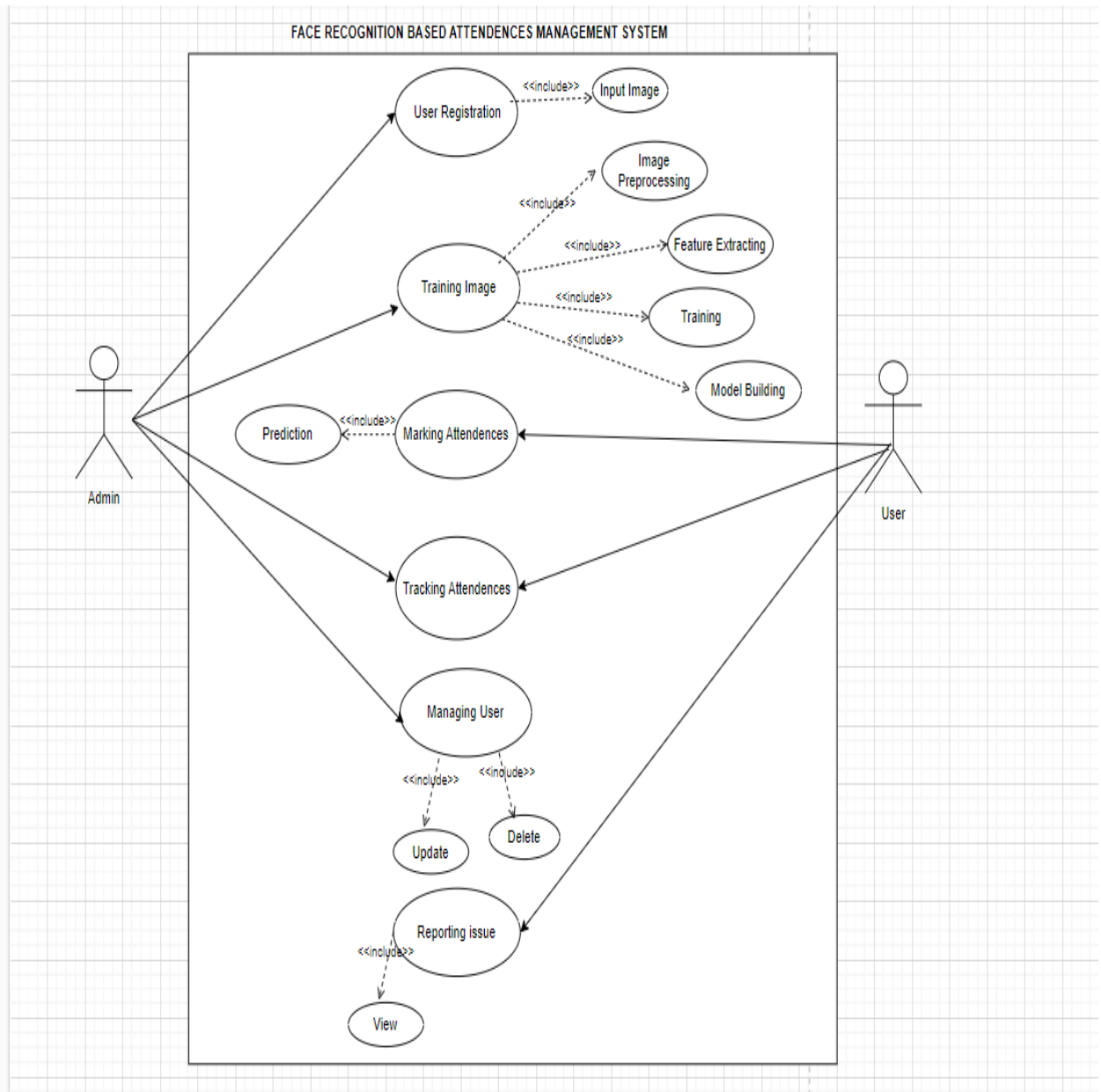
Purpose : To Store Issue details

SI NO.	Field Name	Type	Constraints	Description
1	Issue Id	Int	Primary key	To Store Issue Id
2	Name	varchar(20)	Not Null	To Store Name of User
3	Attendance Id	Int	Foreign Key	To Store id of Department
4	Id	int	Foreign Key	To Store user Id
5	Description	Varchar(20)	Not Null	To Store Description of Issue
6	Response	varchar(20)	Not Null	To store Response from admin

13.3 Annexure 2

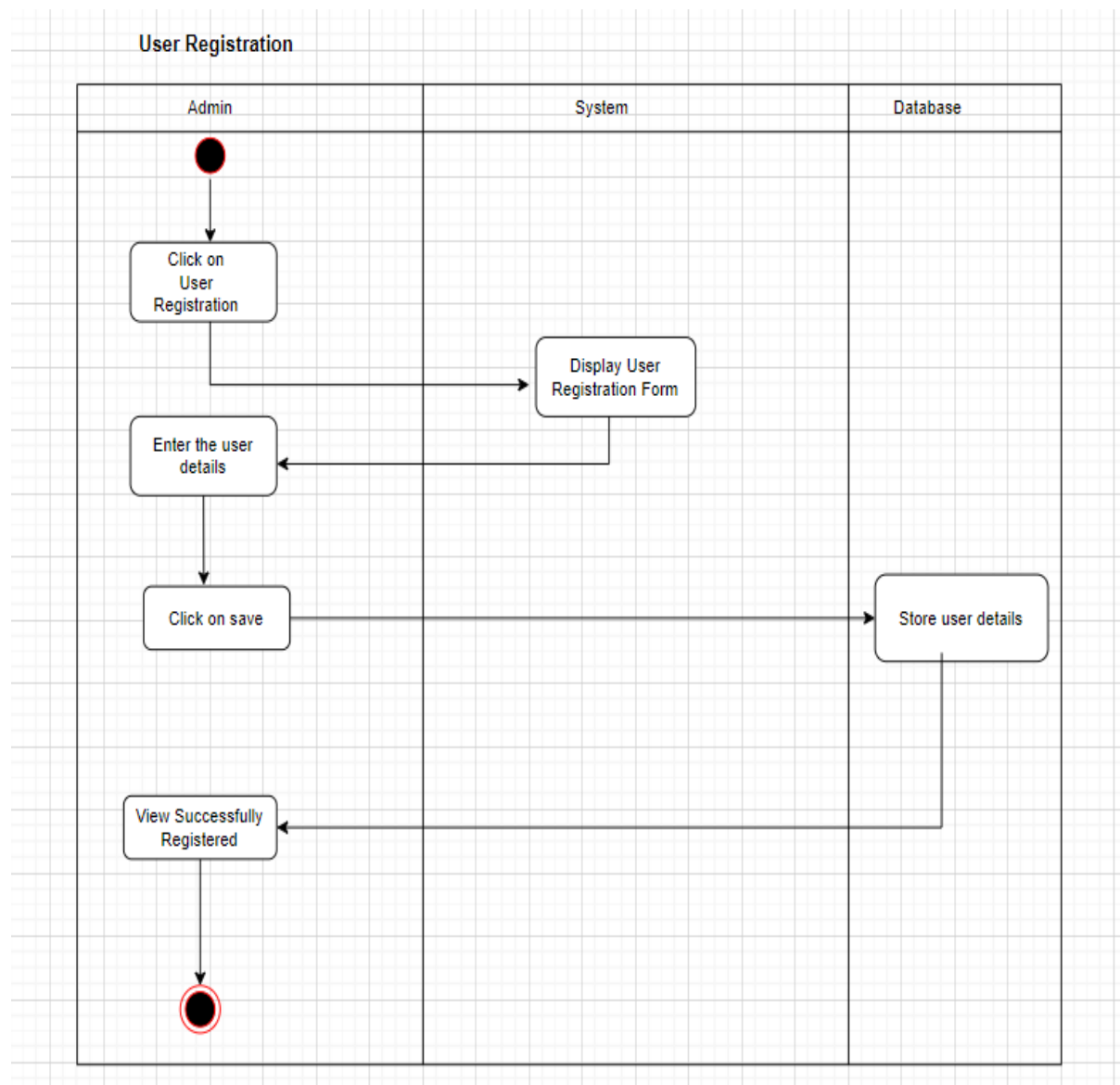
Diagram

2.1 Use Case Diagram

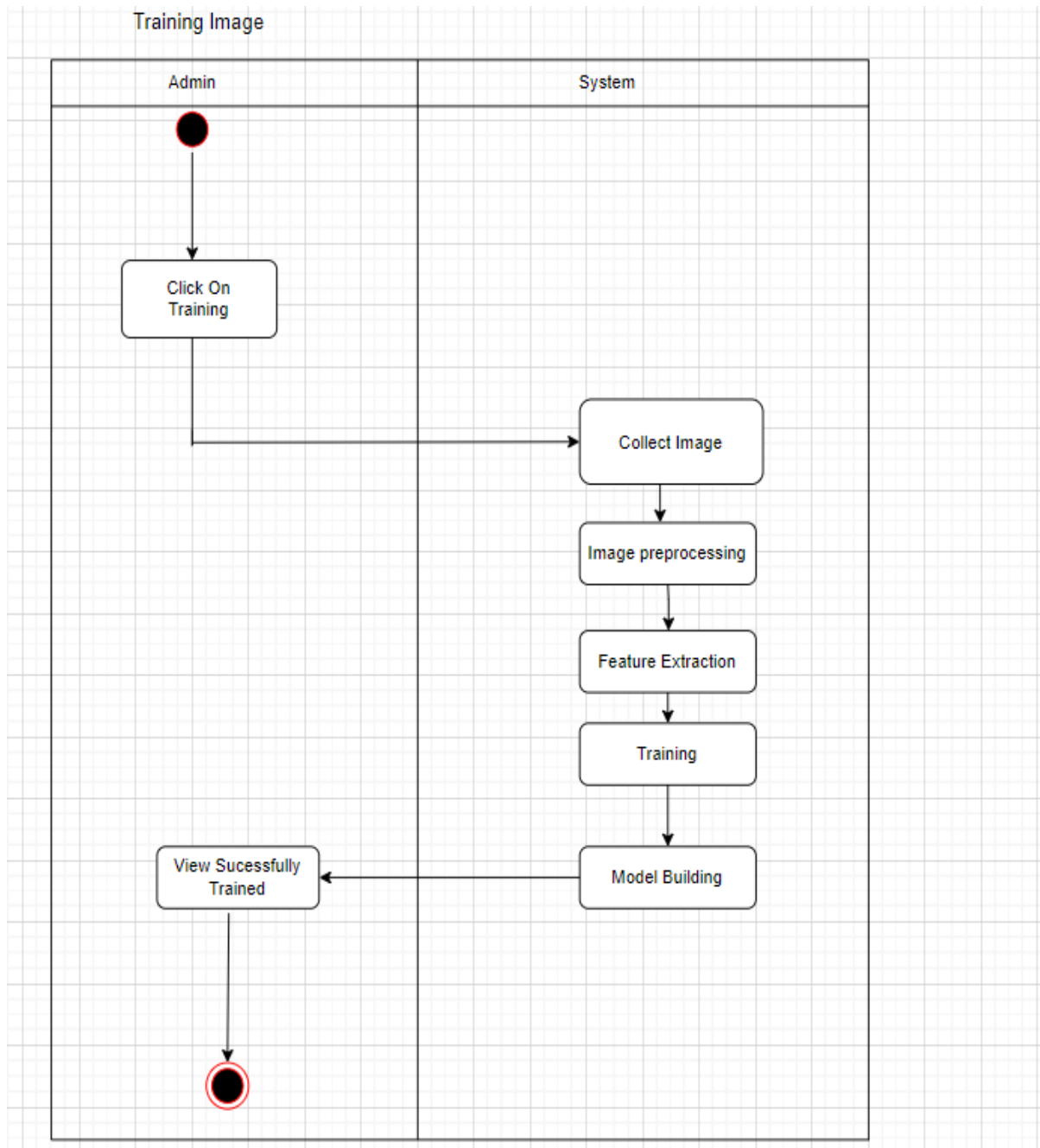


2.2 Activity Diagram

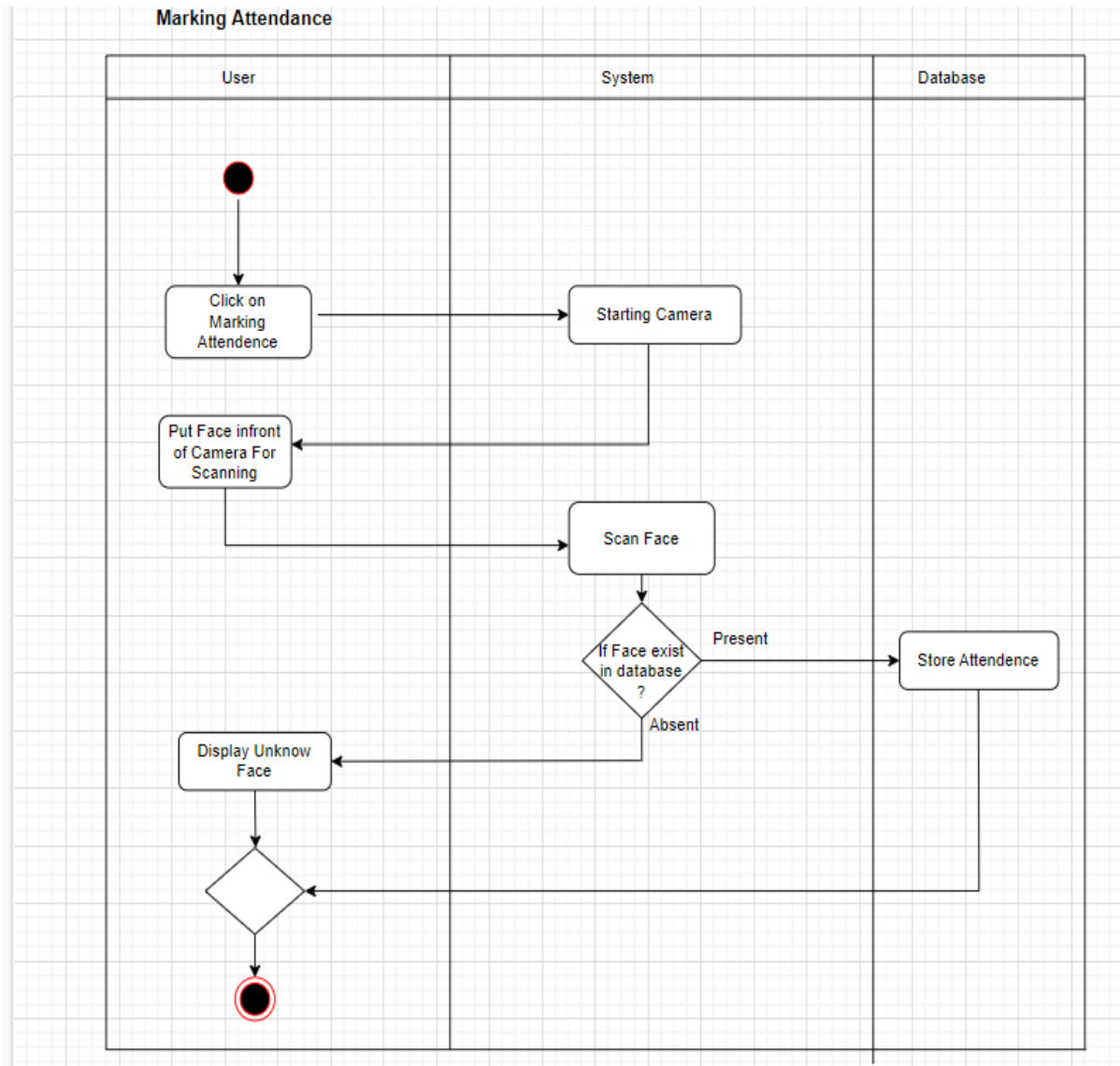
User Registration



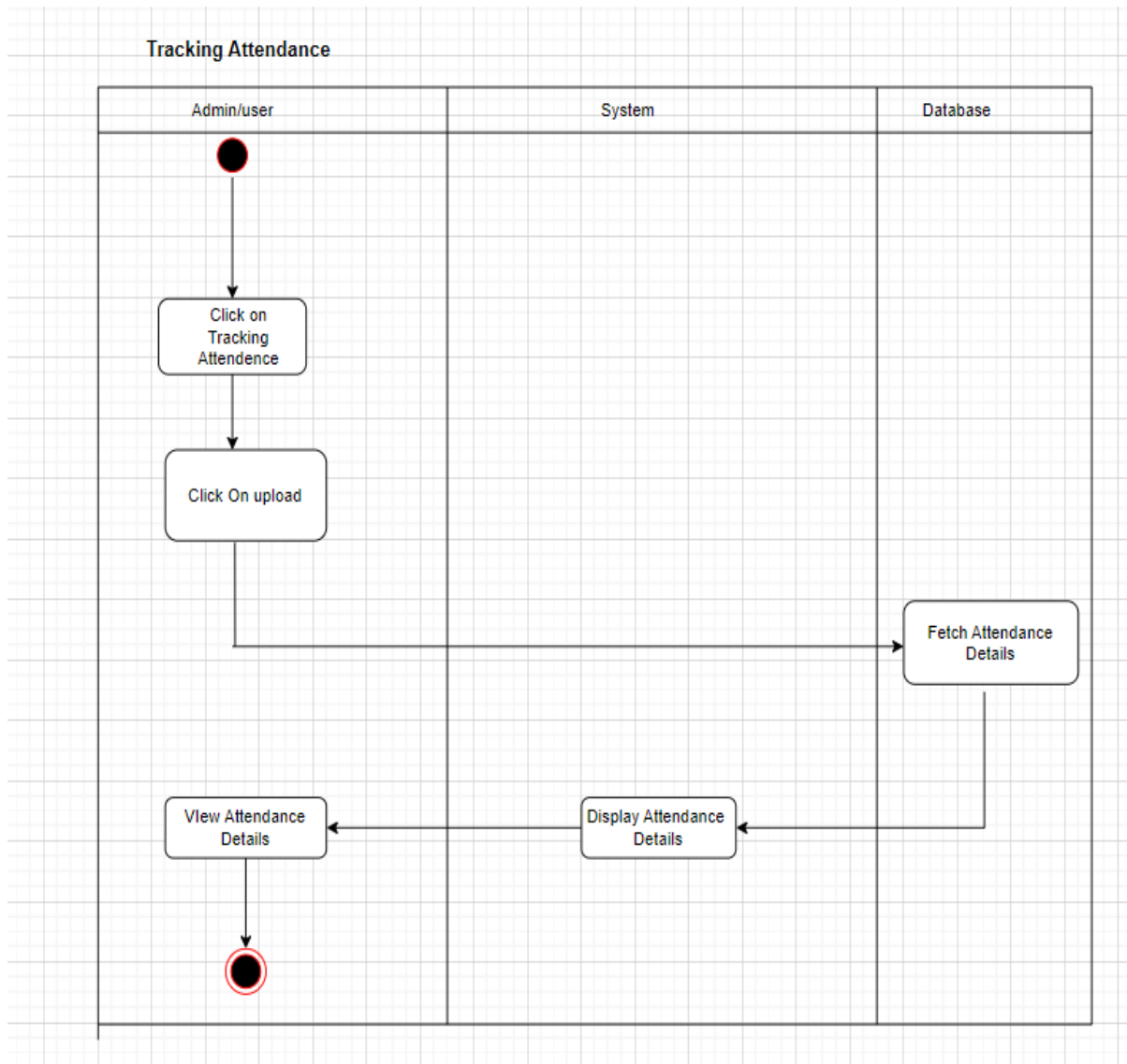
Training Image



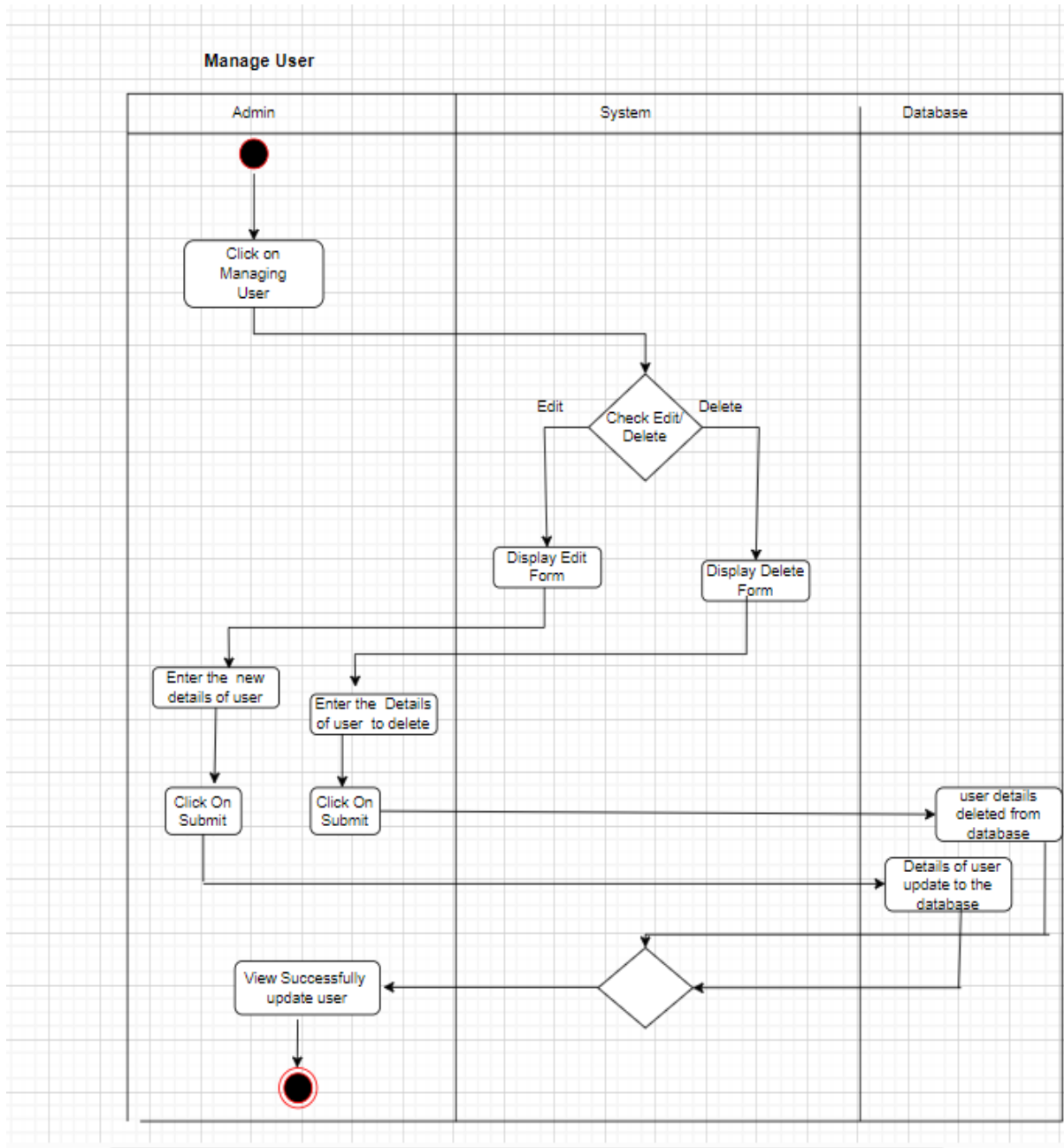
Marking Attendance



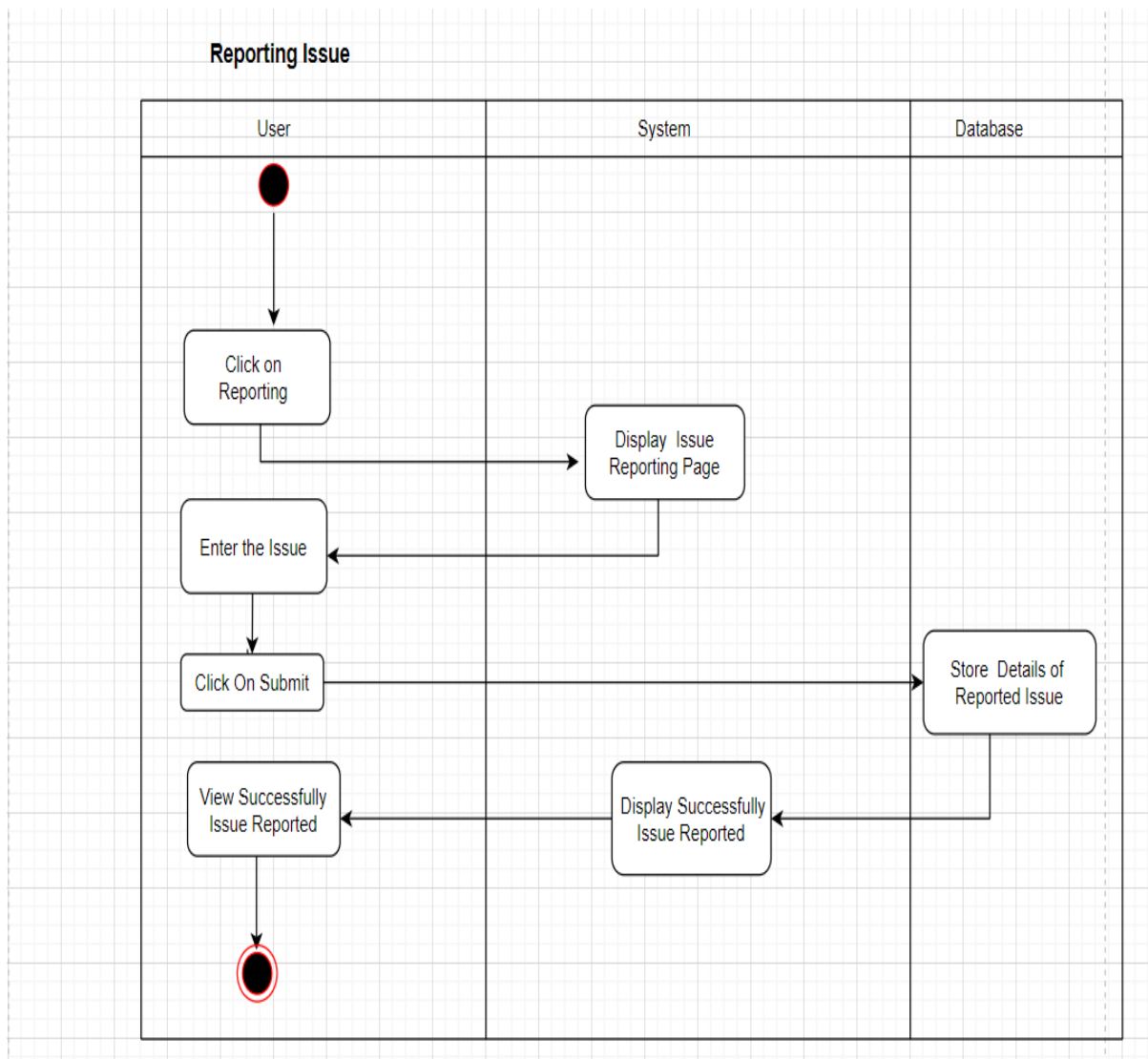
Tracking Attendance



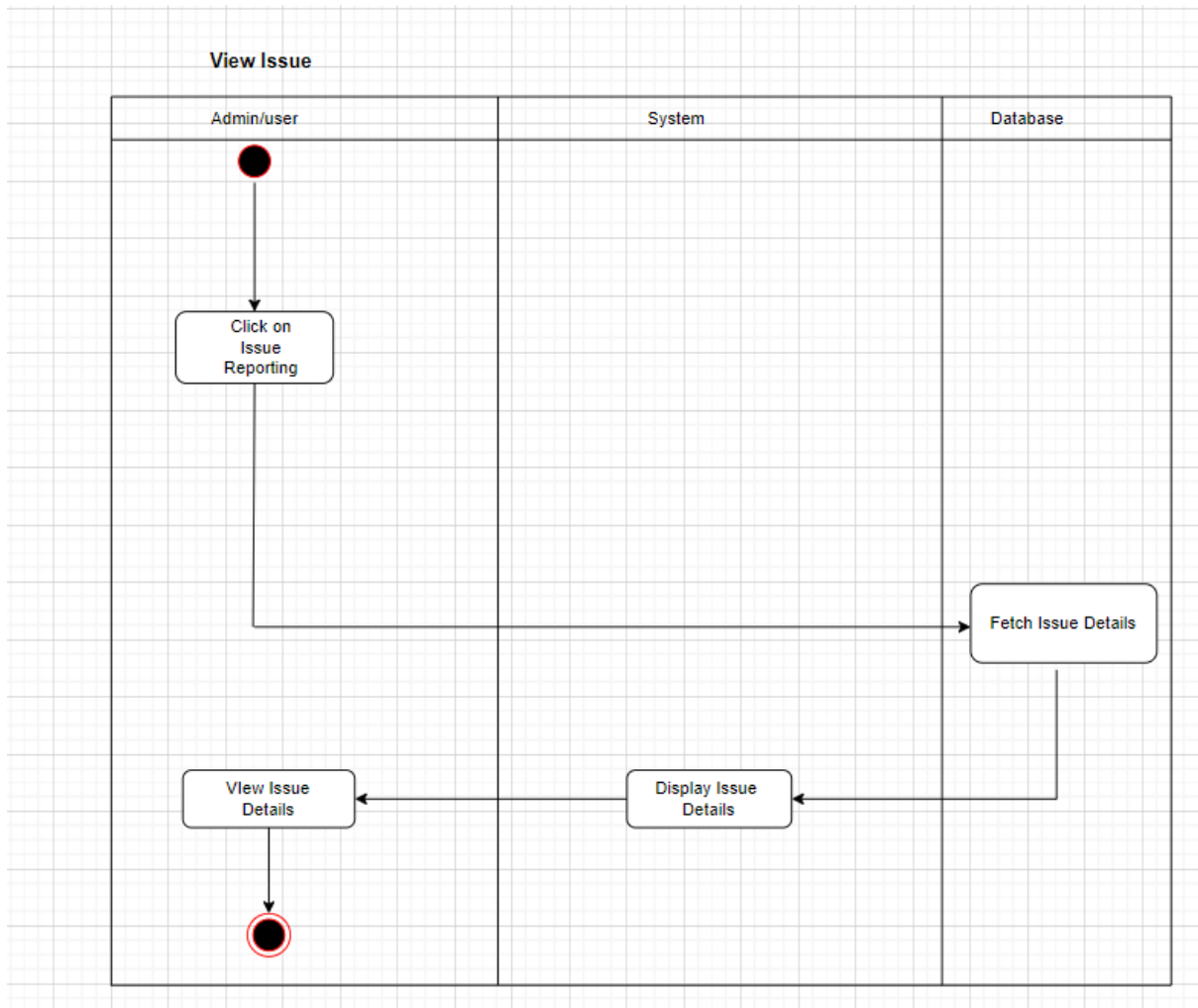
Managing User



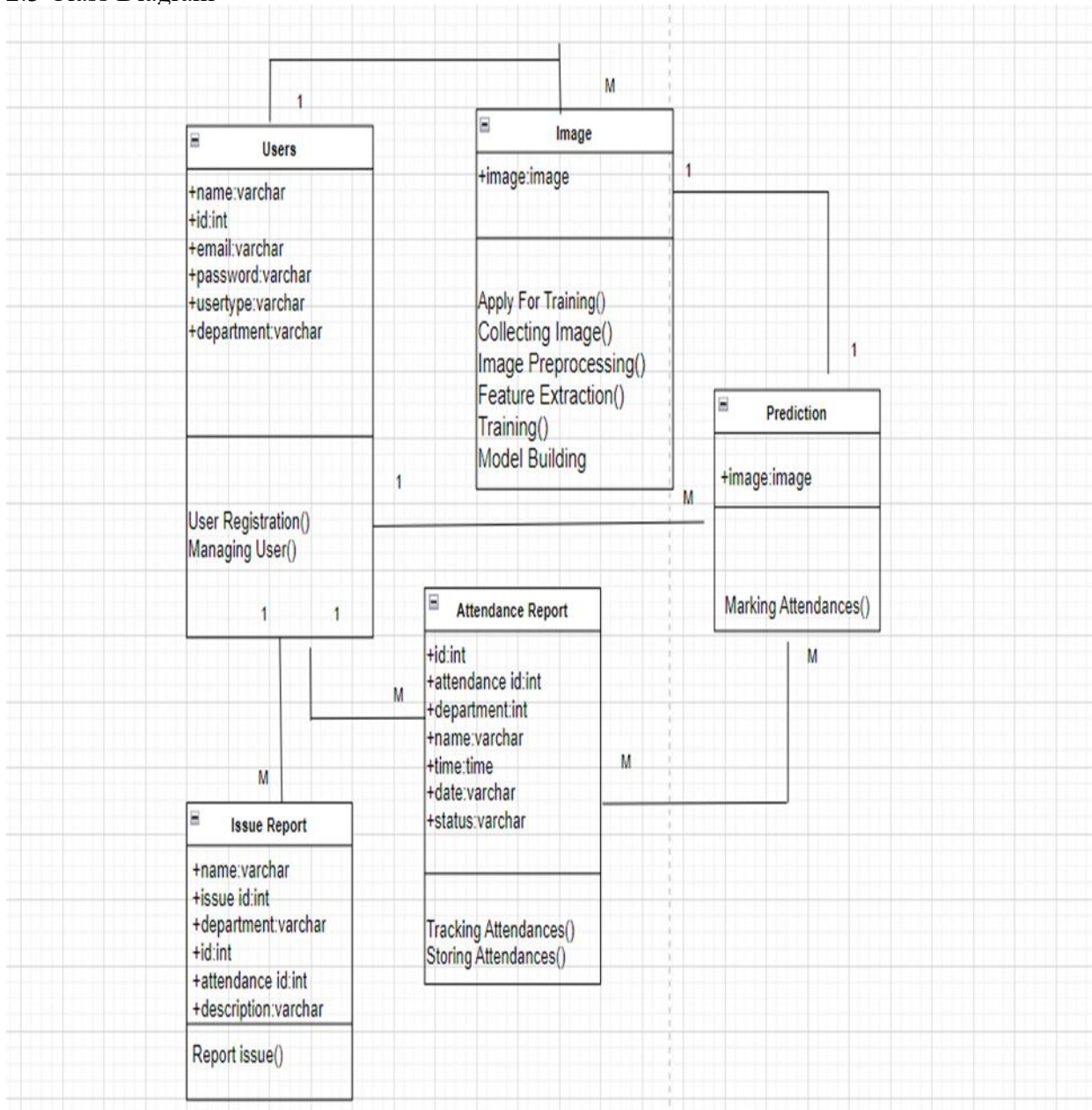
Issue Reporting



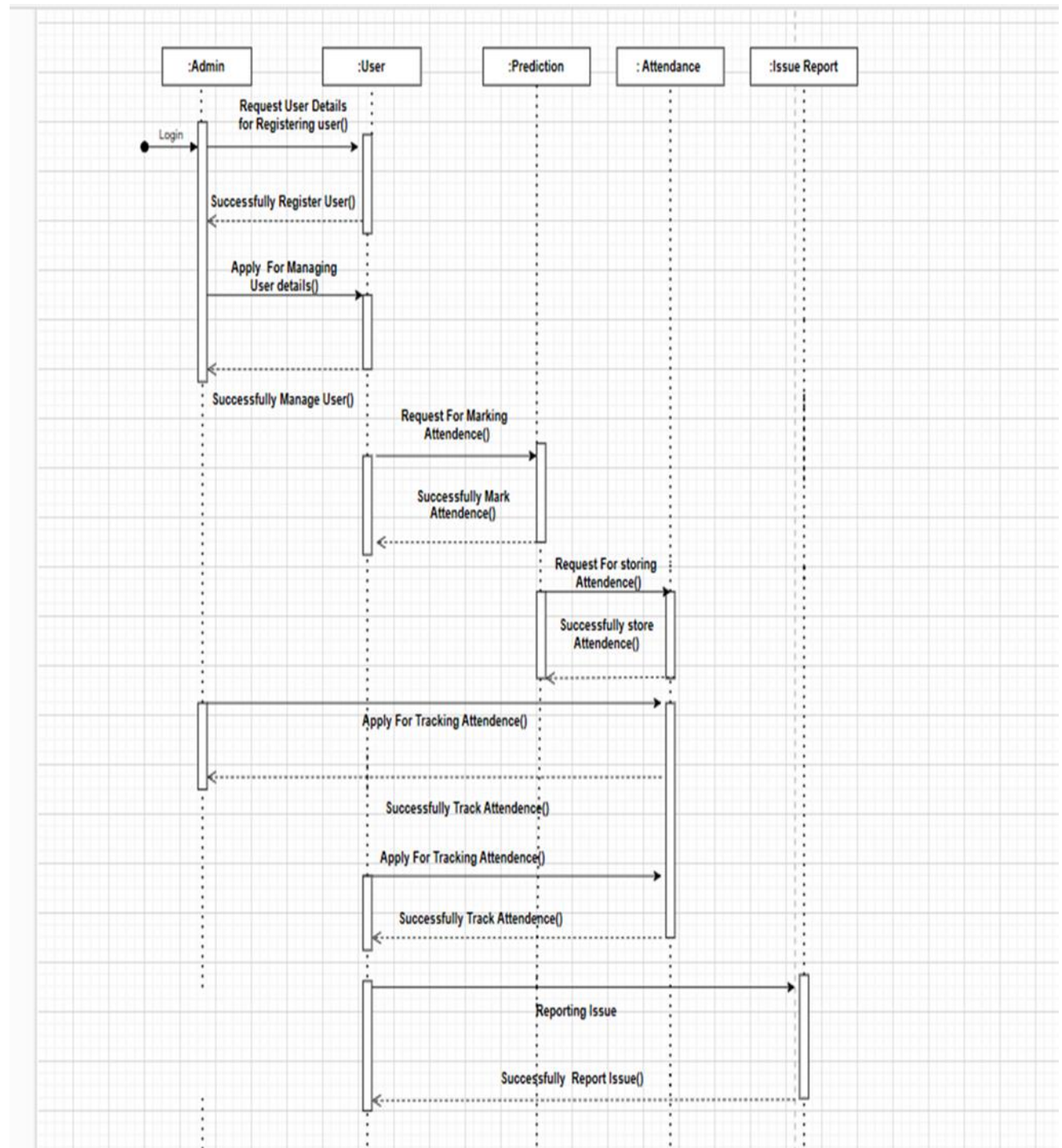
View Issue



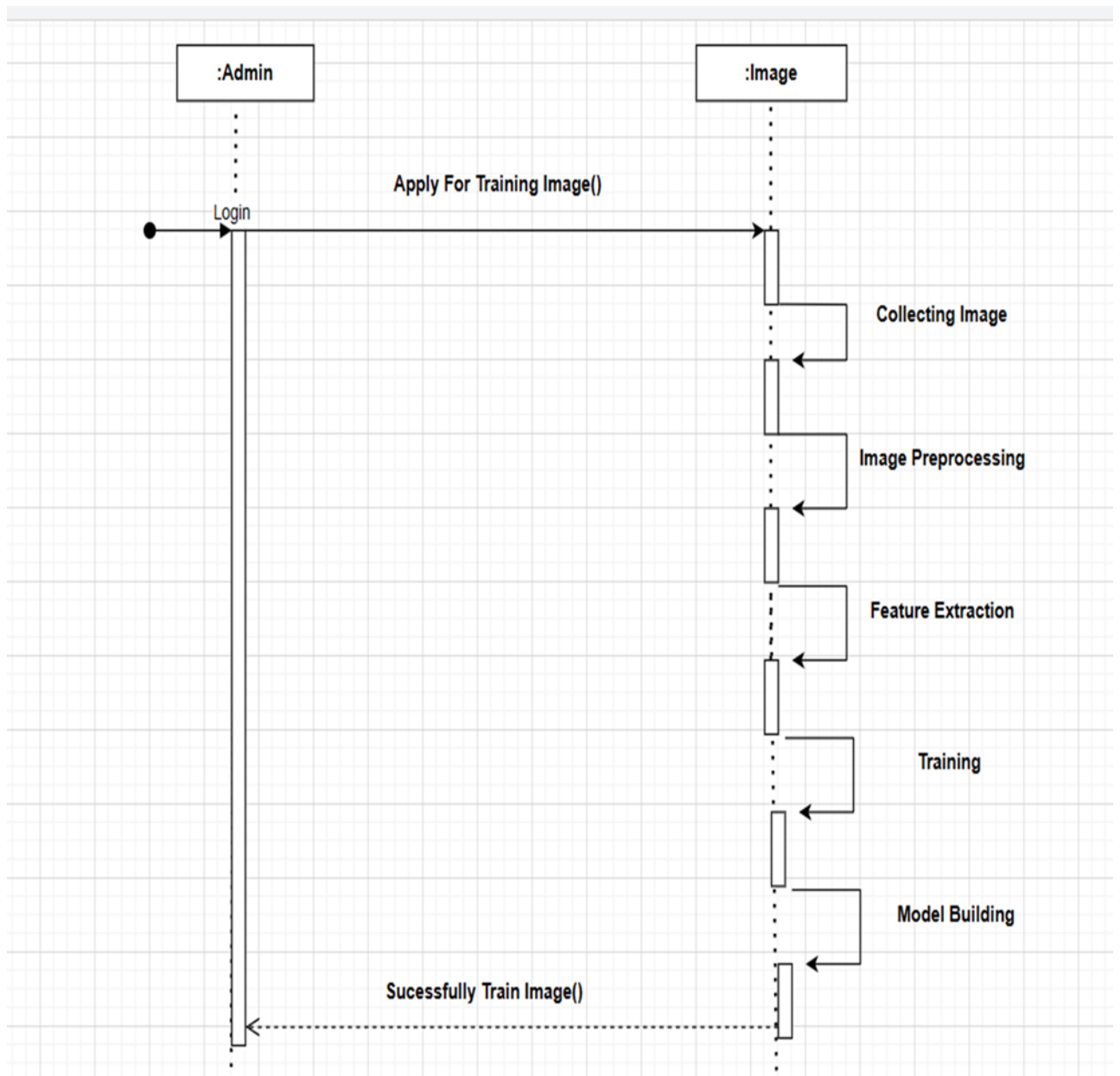
2.3 Class Diagram



2.4 Sequence Diagram



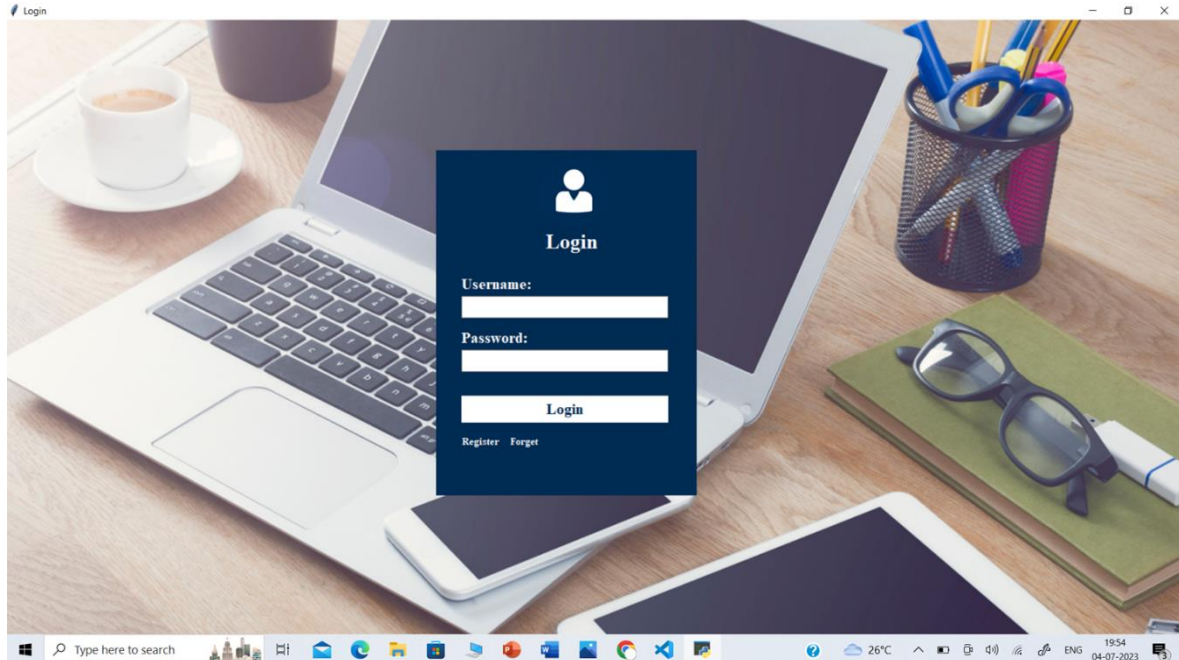
Sequences Diagram



13.4 Annexure 3

Figure

3.1 Login



3.2 User Registration

USER REGISTRAION

User Details

Information

Name

John George

Email:

johngm1999@gmail

Password

Department

MCA

Save

Update

Delete

Reset

Take Pic

Update Pic

Student Details

Search System

Search:

Select

Search

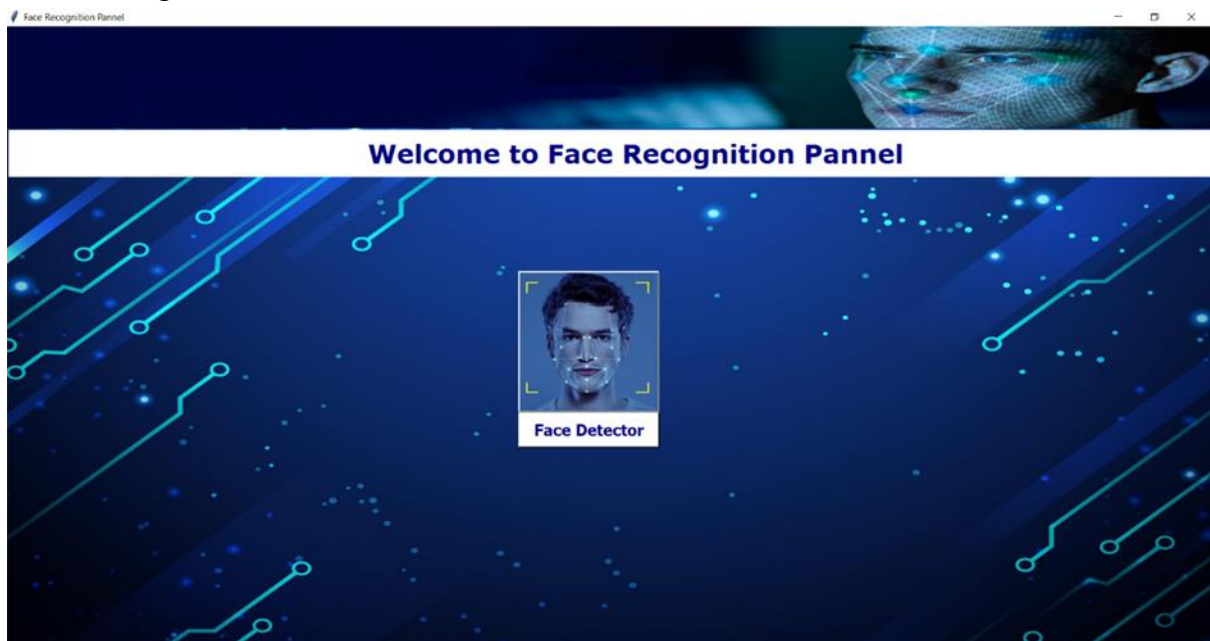
Show All

StudentID	Name	Department	Email	Password
-----------	------	------------	-------	----------

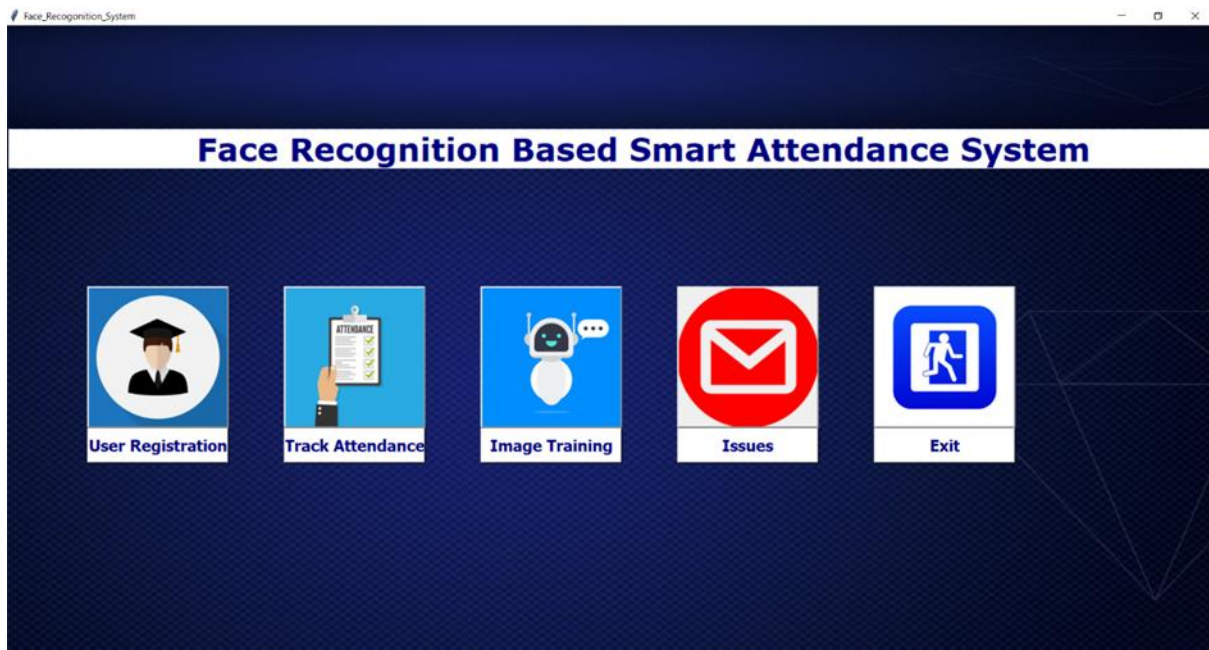
3. 3 Training Image



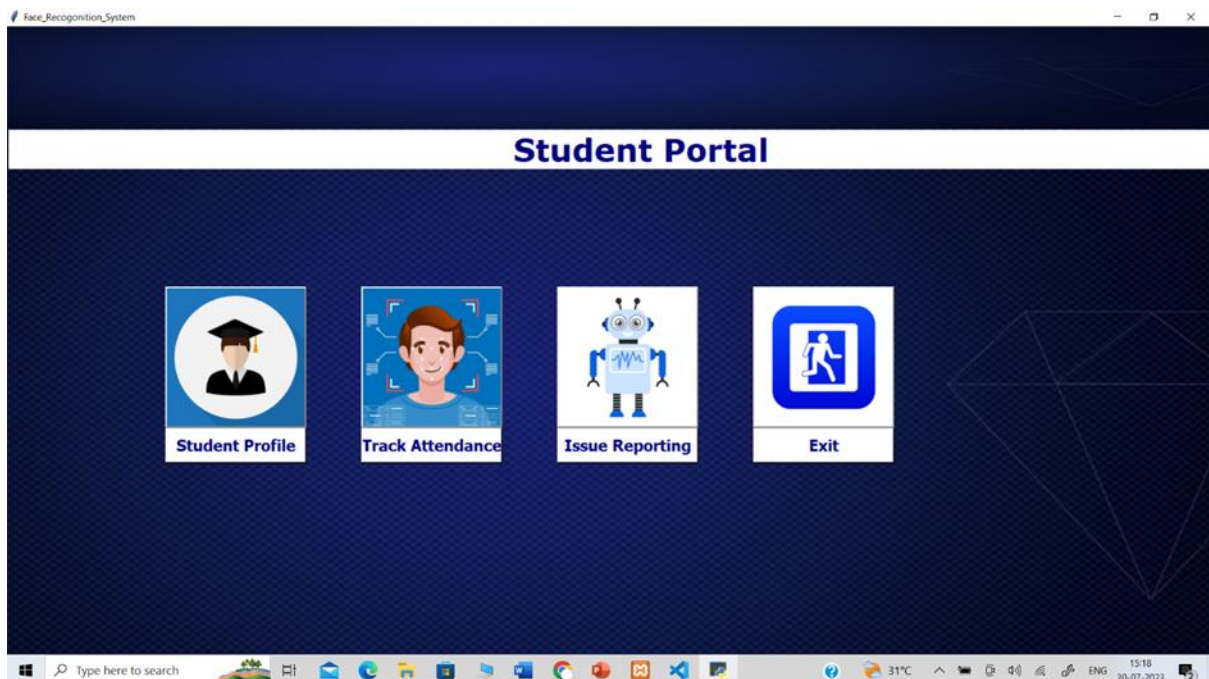
3. 4 Marking Attendance



3.5 Admin Home Page



3.6 User Home Page



3.7 Tracking Attendance by admin

Attendance Pannel

Welcome to Attendance Pannel

Student Details

Att-ID: Std-ID:

Department: Std-Name:

Time: Date:

Attend-status:

Student Details

Att_Id	Std-ID	Department	Std-Name	Time	Date	
74	1	MCA	John	12:56	29/07/2023	P

3. 8 Attendance Track By User

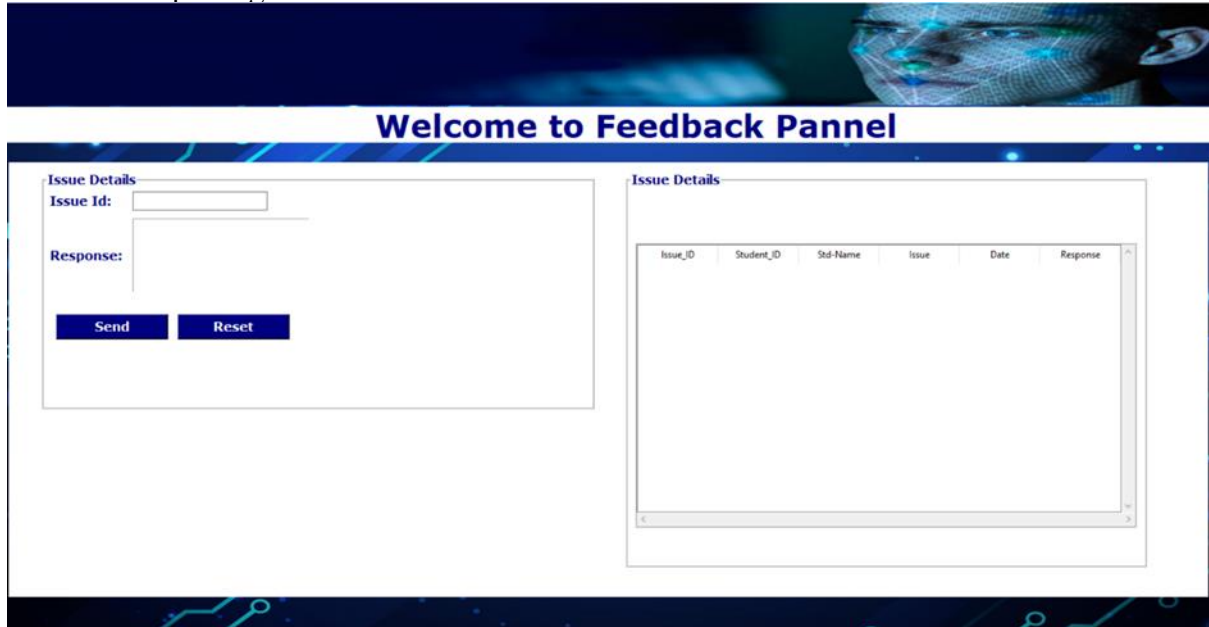
Student Profile

Student Portal

Attendance

Att_Id	Std-ID	Department	Std-Name	Time	Date	Attend-status
74	1	MCA	John	12:56	29/07/2023	Present

3. 9 Issue Reporting



The interface features a dark blue header with a futuristic face graphic on the right. Below the header is a white banner with the text "Welcome to Feedback Pannel". The main content area is divided into two panels. The left panel, titled "Issue Details", contains a form with an "Issue Id:" label and a text input field, a "Response:" label and a larger text area, and two buttons labeled "Send" and "Reset". The right panel, also titled "Issue Details", contains a table with the following headers: "Issue_ID", "Student_ID", "Std-Name", "Issue", "Date", and "Response". The table body is currently empty.

Welcome to Feedback Pannel

Issue Details

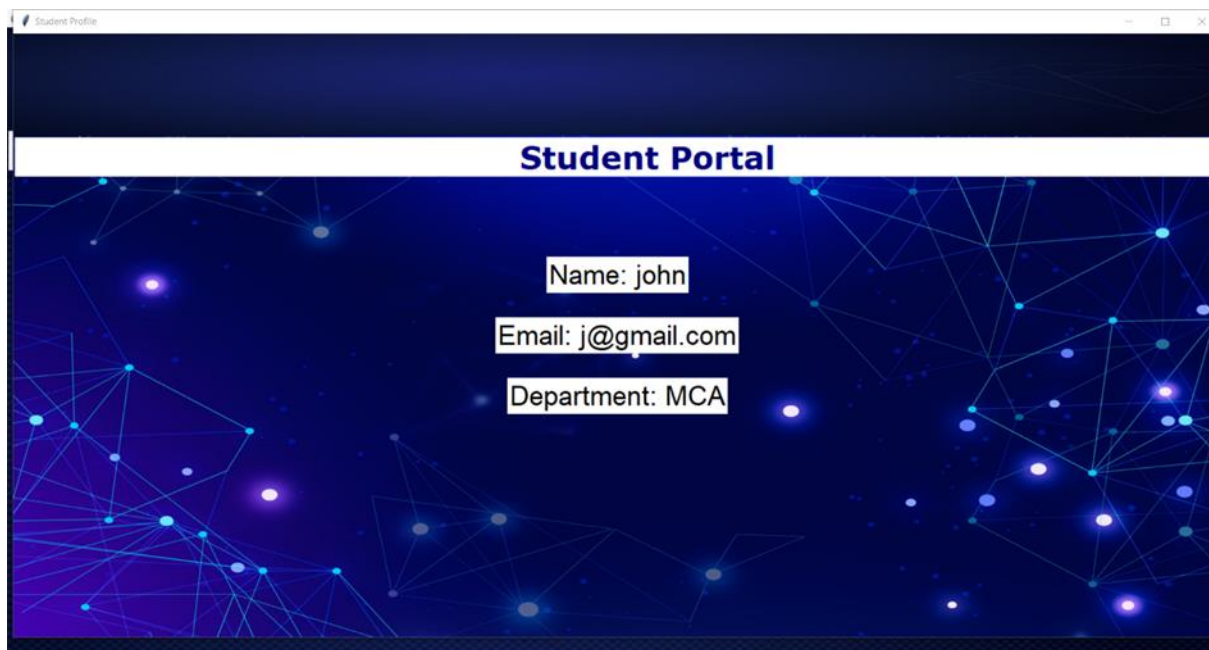
Issue Id:

Response:

Issue Details

Issue_ID	Student_ID	Std-Name	Issue	Date	Response
----------	------------	----------	-------	------	----------

3.10 User Profile



The interface is displayed within a browser window titled "Student Profile". It has a dark blue header with a futuristic face graphic on the right. Below the header is a white banner with the text "Student Portal". The main content area has a dark blue background with a network of glowing blue nodes and lines. In the center, there are three white text boxes containing the following information: "Name: john", "Email: j@gmail.com", and "Department: MCA".

Student Profile

Student Portal

Name: john

Email: j@gmail.com

Department: MCA

13.5 Annexure 4

Code

Student.py

```
from tkinter import*
from tkinter import ttk
from PIL import Image,ImageTk
from tkinter import messagebox
import mysql.connector
import cv2
# Testing Connection
"""

conn = mysql.connector.connect(username='root',
password="",host='localhost',database='face_recognition',port=3306)
cursor = conn.cursor()

cursor.execute("show databases")

data = cursor.fetchall()

print(data)

conn.close()
"""
class Student:
    def __init__(self,root):
        self.root=root
        self.root.geometry("1366x768+0+0")
        self.root.state('zoomed')
        self.root.title("Student Pannel")

        #-----Variables-----

        self.var_std_id=StringVar()
        self.var_std_name=StringVar()
        self.var_email=StringVar()
        self.var_pass=StringVar()
        self.var_dep=StringVar()

        # This part is image labels setting start
        # first header image
        img=Image.open(r"C:\Users\91628\Desktop\Face-Recognition-Attendance1\Images_GUI\banner.jpg")
        img=img.resize((1600,130),Image.ANTIALIAS)
        self.photoimg=ImageTk.PhotoImage(img)
```

```

# set image as lable
f_lb1 = Label(self.root,image=self.photoimg)
f_lb1.place(x=0,y=0,width=1600,height=130)

# backgorund image
bg1=Image.open(r"C:\Users\91628\Desktop\Face-Recognition-
Attendance1\Images_GUI\bg3.jpg")
bg1=bg1.resize((1600,768),Image.ANTIALIAS)
self.photobg1=ImageTk.PhotoImage(bg1)

# set image as lable
bg_img = Label(self.root,image=self.photobg1)
bg_img.place(x=0,y=130,width=1600,height=768)

#title section
title_lb1 = Label(bg_img,text="USER
REGISTRAION",font=("verdana",30,"bold"),bg="white",fg="navyblue")
title_lb1.place(x=0,y=0,width=1600,height=45)

# Creating Frame
main_frame = Frame(bg_img,bd=2,bg="white") #bd mean border
main_frame.place(x=5,y=70,width=1520,height=550)

# Left Label Frame
left_frame = LabelFrame(main_frame,bd=2,bg="white",relief=RIDGE,text="User
Details",font=("verdana",12,"bold"),fg="navyblue")
left_frame.place(x=40,y=10,width=700,height=500)

#Class Student Information
class_Student_frame =
LabelFrame(left_frame,bd=2,bg="white",relief=RIDGE,text="Information",font=("verdana",
12,"bold"),fg="navyblue")
class_Student_frame.place(x=10,y=20,width=635,height=230)

#student id
student_id_label = Label(class_Student_frame,text="Std-
Id:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
student_id_label.grid(row=0,column=0,padx=5,pady=5,sticky=W)

```

```

        student_id_entry =
tkk.Entry(class_Student_frame,textvariable=self.var_std_id,width=15,font=("verdana",12,"bold"))
        student_id_entry.grid(row=0,column=1,padx=5,pady=5,sticky=W)

        #Student name
        student_name_label = Label(class_Student_frame,text="Std-
Name:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
        student_name_label.grid(row=1,column=0,padx=5,pady=5,sticky=W)

        student_name_entry =
tkk.Entry(class_Student_frame,textvariable=self.var_std_name,width=15,font=("verdana",12,"bold"))
        student_name_entry.grid(row=1,column=1,padx=5,pady=5,sticky=W)

        #Email
        student_email_label =
Label(class_Student_frame,text="Email:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
        student_email_label.grid(row=1,column=2,padx=5,pady=5,sticky=W)

        student_email_entry =
tkk.Entry(class_Student_frame,textvariable=self.var_email,width=15,font=("verdana",12,"bold"))
        student_email_entry.grid(row=1,column=3,padx=5,pady=5,sticky=W)

        #Password
        student_password_label =
Label(class_Student_frame,text="Password:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
        student_password_label.grid(row=2,column=0,padx=5,pady=5,sticky=W)

        student_password_entry =
tkk.Entry(class_Student_frame,textvariable=self.var_pass,width=15,font=("verdana",12,"bold"))
        student_password_entry.grid(row=2,column=1,padx=5,pady=5,sticky=W)

        dep_label=Label(class_Student_frame,text="Department",font=("verdana",12,"bold"),bg="white",fg="navyblue")
        dep_label.grid(row=2,column=2,padx=5,pady=15)

        dep_combo=ttk.Combobox(class_Student_frame,textvariable=self.var_dep,width=15,font=("verdana",12,"bold"),state="readonly")
        dep_combo["values"]=("Select Department","MCA","MTECH","BTECH")

```

```

dep_combo.current(0)
dep_combo.grid(row=2,column=3,padx=5,pady=15,sticky=W)

#Button Frame
btn_frame = Frame(left_frame,bd=2,bg="white",relief=RIDGE)
btn_frame.place(x=10,y=390,width=635,height=60)

#save button
save_btn=Button(btn_frame,command=self.add_data,text="Save",width=7,font=("verdana",12,"bold"),fg="white",bg="navyblue")
save_btn.grid(row=0,column=0,padx=5,pady=10,sticky=W)

#update button
update_btn=Button(btn_frame,command=self.update_data,text="Update",width=7,font=
("verdana",12,"bold"),fg="white",bg="navyblue")
update_btn.grid(row=0,column=1,padx=5,pady=8,sticky=W)

#delete button
del_btn=Button(btn_frame,command=self.delete_data,text="Delete",width=7,font=("verdana",12,"bold"),fg="white",bg="navyblue")
del_btn.grid(row=0,column=2,padx=5,pady=10,sticky=W)

#reset button
reset_btn=Button(btn_frame,command=self.reset_data,text="Reset",width=7,font=("verdana",12,"bold"),fg="white",bg="navyblue")
reset_btn.grid(row=0,column=3,padx=5,pady=10,sticky=W)

#take photo button
take_photo_btn=Button(btn_frame,command=self.generate_dataset,text="Take Pic",width=9,font=("verdana",12,"bold"),fg="white",bg="navyblue")
take_photo_btn.grid(row=0,column=4,padx=5,pady=10,sticky=W)

#-----
# Right Label Frame
right_frame = LabelFrame(main_frame,bd=2,bg="white",relief=RIDGE,text="Student Details",font=("verdana",12,"bold"),fg="navyblue")
right_frame.place(x=780,y=10,width=660,height=500)

#Searching System in Right Label Frame
search_frame = LabelFrame(right_frame,bd=2,bg="white",relief=RIDGE,text="Search System",font=("verdana",12,"bold"),fg="navyblue")
search_frame.place(x=10,y=5,width=635,height=80)

#Phone Number
search_label =
Label(search_frame,text="Search:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
search_label.grid(row=0,column=0,padx=5,pady=5,sticky=W)
self.var_searchTX=StringVar()

```

```

#combo box
search_combo=ttk.Combobox(search_frame,textvariable=self.var_searchTX,width=12,font=
("verdana",12,"bold"),state="readonly")
search_combo["values"]=("Select","Student ID")
search_combo.current(0)
search_combo.grid(row=0,column=1,padx=5,pady=15,sticky=W)

self.var_search=StringVar()
search_entry =
ttk.Entry(search_frame,textvariable=self.var_search,width=12,font=("verdana",12,"bold"))
search_entry.grid(row=0,column=2,padx=5,pady=5,sticky=W)

search_btn=Button(search_frame,command=self.search_data,text="Search",width=9,font=
("verdana",12,"bold"),fg="white",bg="navyblue")
search_btn.grid(row=0,column=3,padx=5,pady=10,sticky=W)

showAll_btn=Button(search_frame,command=self.fetch_data,text="Show
All",width=8,font=("verdana",12,"bold"),fg="white",bg="navyblue")
showAll_btn.grid(row=0,column=4,padx=5,pady=10,sticky=W)

# -----Table Frame-----
#Table Frame
#Searching System in Right Label Frame
table_frame = Frame(right_frame,bd=2,bg="white",relief=RIDGE)
table_frame.place(x=10,y=120,width=635,height=360)

#scroll bar
scroll_x = ttk.Scrollbar(table_frame,orient=HORIZONTAL)
scroll_y = ttk.Scrollbar(table_frame,orient=VERTICAL)

#create table
self.student_table =
ttk.Treeview(table_frame,column=("ID","Name","Email","Pass","Dep"),xscrollcommand=scroll_x.set,yscrollcommand=scroll_y.set)

scroll_x.pack(side=BOTTOM,fill=X)
scroll_y.pack(side=RIGHT,fill=Y)
scroll_x.config(command=self.student_table.xview)
scroll_y.config(command=self.student_table.yview)

self.student_table.heading("ID",text="StudentID")
self.student_table.heading("Name",text="Name")
self.student_table.heading("Email",text="Email")
self.student_table.heading("Pass",text="Password")
self.student_table.heading("Dep",text="Department")

self.student_table["show"]="headings"

# Set Width of Columns

```



```

self.student_table.column("ID",width=100)
self.student_table.column("Name",width=100)
self.student_table.column("Email",width=100)
self.student_table.column("Pass",width=100)
self.student_table.column("Dep",width=100)

self.student_table.pack(fill=BOTH,expand=1)
self.student_table.bind("<ButtonRelease>",self.get_cursor)
self.fetch_data()
# =====Function Decleration=====
import re

def add_data(self):
    import re
    if self.var_dep.get() == "Select Department" or self.var_std_name.get() == "" or
self.var_email.get() == "" or self.var_pass.get() == "":
        messagebox.showerror("Error", "Please Fill All Fields are Required!",
parent=self.root)
    else:
        # Check if the entered email is in a valid format
        if not re.match(r"^[^@]+\.[^@]+\.[^@]+$", self.var_email.get()):
            messagebox.showerror("Error", "Invalid Email Format!", parent=self.root)
        else:
            try:
                conn = mysql.connector.connect(username='root', password="", host='localhost',
database='face_recognition', port=3306)
                mycursor = conn.cursor()

                mycursor.execute("insert into
student(StudentID,Name,Email>Password,Department) values(%s,%s,%s,%s,%s)", (
                    self.var_std_id.get(),
                    self.var_std_name.get(),
                    self.var_email.get(),
                    self.var_pass.get(),
                    self.var_dep.get()
                ))

                conn.commit()
                self.fetch_data()
                conn.close()
                messagebox.showinfo("Success", "All Records are Saved!", parent=self.root)
            except Exception as es:
                if "1062 (23000): Duplicate entry" in str(es):
                    messagebox.showerror("Error", "Already Registered Email or Student Id",
parent=self.root)
                else:
                    messagebox.showerror("Error", f"Due to: {str(es)}", parent=self.root)

```

```

# =====Fetch data form database to table
=====

def fetch_data(self):
    conn = mysql.connector.connect(username='root',
password=",host='localhost',database='face_recognition',port=3306)
    mycursor = conn.cursor()

    mycursor.execute("select * from student")
    data=mycursor.fetchall()

    if len(data)!= 0:
        self.student_table.delete(*self.student_table.get_children())
        for i in data:
            self.student_table.insert("",END,values=i)
        conn.commit()
        conn.close()

#=====get cursor
function=====

def get_cursor(self,event=""):
    cursor_focus = self.student_table.focus()
    content = self.student_table.item(cursor_focus)
    data = content["values"]
    print(data)
    self.var_std_id.set(data[0]),
    self.var_std_name.set(data[1]),
    self.var_email.set(data[2]),
    self.var_pass.set(data[3]),
    self.var_dep.set(data[4])

# =====Update
Function=====

def update_data(self):
    if self.var_dep.get()=="Select Department" or self.var_std_name.get==" or
self.var_email.get()==" or
self.var_pass.get()==" :      messagebox.showerror("Error","Please Fill All Fields are
Required!",parent=self.root)
    else:
        try:
            Update=messagebox.askyesno("Update","Do you want to Update this Student
Details!",parent=self.root)
            if Update > 0:
                conn = mysql.connector.connect(username='root',
password=",host='localhost',database='face_recognition',port=3306)
                mycursor = conn.cursor()
                mycursor.execute("update student set Name=%s,Email=%s>Password=%s,
Department=%s where StudentID =%s ",(

```

```

        self.var_std_name.get(),
        self.var_email.get(),
        self.var_pass.get(),
        self.var_dep.get(),
        self.var_std_id.get()
    ))

    else:
        if not Update:
            return
        messagebox.showinfo("Success", "Successfully Updated!", parent=self.root)
        conn.commit()
        self.fetch_data()
        conn.close()
    except Exception as es:
        messagebox.showerror("Error", f"Due to: {str(es)}", parent=self.root)

#=====Delete
Function=====
def delete_data(self):
    if self.var_std_id.get()=="":
        messagebox.showerror("Error", "Student Id Must be Required!", parent=self.root)
    else:
        try:
            delete=messagebox.askyesno("Delete", "Do you want to Delete?", parent=self.root)
            if delete>0:
                conn = mysql.connector.connect(username='root',
password=" ", host='localhost', database='face_recognition', port=3306)
                mycursor = conn.cursor()
                sql="delete from student where StudentID=%s"
                val=(self.var_std_id.get(),)
                mycursor.execute(sql, val)
            else:
                if not delete:
                    return

            conn.commit()
            self.fetch_data()
            conn.close()
            messagebox.showinfo("Delete", "Successfully Deleted!", parent=self.root)
        except Exception as es:
            messagebox.showerror("Error", f"Due to: {str(es)}", parent=self.root)

# Reset Function

def reset_data(self):
    self.var_std_id.set(""),
    self.var_std_name.set(""),

```

```

self.var_email.set(""),
self.var_pass.set(""),
self.var_dep.set("Select Department")

# =====Search Data=====
def search_data(self):
    if self.var_search.get()==" or self.var_searchTX.get()=="Select":
        messagebox.showerror("Error","Select Combo option and enter entry
box",parent=self.root)
    else:
        try:
            conn = mysql.connector.connect(username='root',
password=" ,host='localhost',database='face_recognition',port=3306)
            my_cursor = conn.cursor()
            sql = "SELECT StudentID,Name,Email,Password,Department FROM student
where StudentId=" +str(self.var_search.get()) + ""
            my_cursor.execute(sql)
            # my_cursor.execute("select * from student where Roll_No= "
+str(self.var_search.get())+" "+str(self.var_searchTX.get())+"")
            rows=my_cursor.fetchall()
            if len(rows)!=0:
                self.student_table.delete(*self.student_table.get_children())
                for i in rows:
                    self.student_table.insert("",END,values=i)
            else:
                messagebox.showerror("Error","Data Not Found",parent=self.root)
                conn.commit()
            conn.close()
        except Exception as es:
            messagebox.showerror("Error",f"Due To :{str(es)}",parent=self.root)

#=====This part is related to Opencv Camera
part=====
# =====Generate Data set take
image=====
def generate_dataset(self):
    if self.var_dep.get()=="Select Department" or self.var_std_name.get==" or
self.var_email.get()==" or self.var_pass.get()=="":
        messagebox.showerror("Error","Please Fill All Fields are Required!",parent=self.root)
    else:
        try:
            conn = mysql.connector.connect(username='root',
password=" ,host='localhost',database='face_recognition',port=3306)
            mycursor = conn.cursor()
            mycursor.execute("select * from student")
            myreslut = mycursor.fetchall()
            print(myreslut)

```

```

id=0
id1 = self.var_std_id.get()
for x in myreslut:
    id+=1

    mycursor.execute("update student set Name=%s,Email=%s>Password=%s,
Department=%s where StudentID=%s", (
        self.var_std_name.get(),
        self.var_email.get(),
        self.var_pass.get(),
        self.var_dep.get(),
        self.var_std_id.get()
    ))
    conn.commit()
    self.fetch_data()
    self.reset_data()
    conn.close()

# =====part of opencv=====

face_classifier = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")

def face_cropped(img):
    # conver gary sacle
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    faces = face_classifier.detectMultiScale(gray,1.3,5)
    #Scaling factor 1.3
    # Minimum naber 5
    for (x,y,w,h) in faces:
        face_cropped=img[y:y+h,x:x+w]
    return face_cropped
cap=cv2.VideoCapture(0)
img_id=0
while True:
    ret,my_frame=cap.read()
    if face_cropped(my_frame) is not None:
        img_id+=1
        face=cv2.resize(face_cropped(my_frame),(200,200))
        face=cv2.cvtColor(face,cv2.COLOR_BGR2GRAY)
        file_path="data_img/student."+str(id1)+"."+str(img_id)+".jpg"
        cv2.imwrite(file_path,face)
        cv2.putText(face,str(img_id),(50,50),cv2.FONT_HERSHEY_COMPLEX,2,(0
,255,0),2)
        cv2.imshow("Capture Images",face)

    if cv2.waitKey(1)==13 or int(img_id)==100:
        break
cap.release()
cv2.destroyAllWindows()
messagebox.showinfo("Result","Generating dataset completed!",parent=self.root)

```

```
except Exception as es:
    messagebox.showerror("Error",f"Due to: {str(es)}",parent=self.root)
```

```
# main class object
```

```
if __name__ == "__main__":
    root=Tk()
    obj=Student(root)
    root.mainloop()
```

Attendance.py

```
import re
import re
from sys import path
from tkinter import*
from tkinter import ttk
from PIL import Image,ImageTk
import os
import mysql.connector
import cv2
import numpy as np
from tkinter import messagebox
from time import strftime
from datetime import datetime
import csv
from tkinter import filedialog

#Global variable for importCsv Function
mydata=[]
class Attendance:

    def __init__(self,root):
        self.root=root
        self.root.geometry("1600x768+0+0")
        self.root.state('zoomed')
        self.root.title("Attendance Pannel")

        #-----Variables-----
        self.var_attid=StringVar()
        self.var_id=StringVar()
        self.var_dep=StringVar()
        self.var_name=StringVar()
        self.var_dep=StringVar()
        self.var_time=StringVar()
        self.var_date=StringVar()
        self.var_attend=StringVar()
```

```

# This part is image labels setting start
# first header image
img=Image.open(r"C:\Users\91628\Desktop\Face-Recognition-Attendance1\Images_GUI\banner.jpg")
img=img.resize((1600,130),Image.ANTIALIAS)
self.photoimg=ImageTk.PhotoImage(img)

# set image as lable
f_lb1 = Label(self.root,image=self.photoimg)
f_lb1.place(x=0,y=0,width=1600,height=130)

# backgorund image
bg1=Image.open(r"Images_GUI\bg2.jpg")
bg1=bg1.resize((1600,768),Image.ANTIALIAS)
self.photobg1=ImageTk.PhotoImage(bg1)

# set image as lable
bg_img = Label(self.root,image=self.photobg1)
bg_img.place(x=0,y=130,width=1600,height=768)

#title section
title_lb1 = Label(bg_img,text="Welcome to Attendance
Pannel",font=("verdana",30,"bold"),bg="white",fg="navyblue")
title_lb1.place(x=0,y=0,width=1600,height=45)

#=====Section
Creating=====

# Creating Frame
main_frame = Frame(bg_img,bd=2,bg="white") #bd mean border
main_frame.place(x=5,y=70,width=1520,height=550)

# Left Label Frame
left_frame = LabelFrame(main_frame,bd=2,bg="white",relief=RIDGE,text="Student
Details",font=("verdana",12,"bold"),fg="navyblue")
left_frame.place(x=40,y=10,width=700,height=500)

# =====Text boxes and Combo
Boxes=====
studentattId_label = Label(left_frame,text="Att-
ID:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
studentattId_label.grid(row=0,column=0,padx=5,pady=5,sticky=W)

studentattId_entry =
ttk.Entry(left_frame,textvariable=self.var_attid,width=15,font=("verdana",12,"bold"))
studentattId_entry.grid(row=0,column=1,padx=5,pady=5,sticky=W)

```

```

#Student id
studentId_label=Label(left_frame,text="Std-
ID:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
studentId_label.grid(row=0,column=2,padx=5,pady=5,sticky=W)

studentId_entry =
ttk.Entry(left_frame,textvariable=self.var_id,width=15,font=("verdana",12,"bold"))
studentId_entry.grid(row=0,column=3,padx=5,pady=5,sticky=W)

#Student Roll
student_roll_label =
Label(left_frame,text="Department:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
student_roll_label.grid(row=1,column=0,padx=5,pady=5,sticky=W)

student_roll_entry =
ttk.Entry(left_frame,textvariable=self.var_dep,width=15,font=("verdana",12,"bold"))
student_roll_entry.grid(row=1,column=1,padx=5,pady=5,sticky=W)

#Studnet Name
student_name_label = Label(left_frame,text="Std-
Name:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
student_name_label.grid(row=1,column=2,padx=5,pady=5,sticky=W)

student_name_entry =
ttk.Entry(left_frame,textvariable=self.var_name,width=15,font=("verdana",12,"bold"))
student_name_entry.grid(row=1,column=3,padx=5,pady=5,sticky=W)

#Department
# dep_label =
Label(left_frame,text="Department:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
# dep_label.grid(row=1,column=2,padx=5,pady=5,sticky=W)

# dep_entry =
ttk.Entry(left_frame,textvariable=self.var_dep,width=15,font=("verdana",12,"bold"))
# dep_entry.grid(row=1,column=3,padx=5,pady=5,sticky=W)

#time
time_label =
Label(left_frame,text="Time:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
time_label.grid(row=2,column=0,padx=5,pady=5,sticky=W)

time_entry =
ttk.Entry(left_frame,textvariable=self.var_time,width=15,font=("verdana",12,"bold"))
time_entry.grid(row=2,column=1,padx=5,pady=5,sticky=W)

#Date
date_label =
Label(left_frame,text="Date:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
date_label.grid(row=2,column=2,padx=5,pady=5,sticky=W)

```



```

date_entry =
tk.Entry(left_frame,textvariable=self.var_date,width=15,font=("verdana",12,"bold"))
date_entry.grid(row=2,column=3,padx=5,pady=5,sticky=W)

#Attendance
student_attend_label = Label(left_frame,text="Attend-
status:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
student_attend_label.grid(row=3,column=0,padx=5,pady=5,sticky=W)

attend_combo=ttk.Combobox(left_frame,textvariable=self.var_attend,width=13,font=(
"verdana",12,"bold"),state="readonly")
attend_combo["values"]=("Status","Present","Absent")
attend_combo.current(0)
attend_combo.grid(row=3,column=1,padx=5,pady=5,sticky=W)

# =====Table Sql Data
View=====

# =====button section=====

#Button Frame
btn_frame = Frame(main_frame,bd=2,bg="white",relief=RIDGE)
btn_frame.place(x=50,y=200,width=635,height=60)

#Exprot button
update_btn=Button(btn_frame,command=self.exportCsv,text="Clear
CSV",width=12,font=("verdana",12,"bold"),fg="white",bg="navyblue")
update_btn.grid(row=0,column=1,padx=6,pady=8,sticky=W)

#Update button
del_btn=Button(btn_frame,command=self.action,text="Upload",width=12,font=("verda
na",12,"bold"),fg="white",bg="navyblue")
del_btn.grid(row=0,column=2,padx=6,pady=10,sticky=W)

#reset button
reset_btn=Button(btn_frame,command=self.reset_data,text="Reset",width=12,font=("ve
rdana",12,"bold"),fg="white",bg="navyblue")
reset_btn.grid(row=0,column=3,padx=6,pady=10,sticky=W)

refresh_btn=Button(btn_frame,command=self.refresh_data,text="Refresh",width=12,font=
("verdana",12,"bold"),fg="white",bg="navyblue")
refresh_btn.grid(row=0,column=4,padx=6,pady=10,sticky=W)

```

```

# Right
section=====

# Right Label Frame
right_frame = LabelFrame(main_frame,bd=2,bg="white",relief=RIDGE,text="Student
Details",font=("verdana",12,"bold"),fg="navyblue")
right_frame.place(x=780,y=10,width=660,height=500)

# -----Table Frame-----
#Table Frame
#Searching System in Right Label Frame
table_frame = Frame(right_frame,bd=2,bg="white",relief=RIDGE)
table_frame.place(x=10,y=90,width=635,height=360)

#scroll bar
scroll_x = ttk.Scrollbar(table_frame,orient=HORIZONTAL)
scroll_y = ttk.Scrollbar(table_frame,orient=VERTICAL)

#create table
self.attendanceReport =
ttk.Treeview(table_frame,column=("Att_Id","ID","Department","Name","Time","Date","Att
end"),xscrollcommand=scroll_x.set,yscrollcommand=scroll_y.set)

scroll_x.pack(side=BOTTOM,fill=X)
scroll_y.pack(side=RIGHT,fill=Y)
scroll_x.config(command=self.attendanceReport.xview)
scroll_y.config(command=self.attendanceReport.yview)

self.attendanceReport.heading("Att_Id",text="Att_Id")
self.attendanceReport.heading("ID",text="Std-ID")
self.attendanceReport.heading("Department",text="Department")
self.attendanceReport.heading("Name",text="Std-Name")
self.attendanceReport.heading("Time",text="Time")
self.attendanceReport.heading("Date",text="Date")
self.attendanceReport.heading("Attend",text="Attend-status")
self.attendanceReport["show"]="headings"

# Set Width of COLUMNS
self.attendanceReport.column("Att_Id",width=100)
self.attendanceReport.column("ID",width=100)
self.attendanceReport.column("Department",width=100)
self.attendanceReport.column("Name",width=100)
self.attendanceReport.column("Time",width=100)
self.attendanceReport.column("Date",width=100)
self.attendanceReport.column("Attend",width=100)

self.attendanceReport.pack(fill=BOTH,expand=1)

```

```

        self.attendanceReport.bind("<ButtonRelease>",self.get_cursor_right)
        self.fetch_data()
        # =====update for mysql
button=====
        #Update button
        del_btn=Button(right_frame,command=self.update_data,text="Update",width=12,font=(
"verdana",12,"bold"),fg="white",bg="navyblue")
        del_btn.grid(row=0,column=1,padx=6,pady=10,sticky=W)
        #Update button
        del_btn=Button(right_frame,command=self.delete_data,text="Delete",width=12,font=(
"verdana",12,"bold"),fg="white",bg="navyblue")
        del_btn.grid(row=0,column=2,padx=6,pady=10,sticky=W)
        # =====update function for mysql
database=====
        def update_data(self):
            if self.var_id.get()==" or self.var_dep.get==" or self.var_name.get()==" or
self.var_time.get()==" or self.var_date.get()==" or self.var_attend.get()=="Status":
                messagebox.showerror("Error","Please Fill All Fields are Required!",parent=self.root)
            else:
                try:
                    Update=messagebox.askyesno("Update","Do you want to Update this Student
Attendance!",parent=self.root)
                    if Update > 0:
                        conn = mysql.connector.connect(username='root',
password=' ',host='localhost',database='face_recognition',port=3306)
                        mycursor = conn.cursor()
                        mycursor.execute("update stdattendance set
std_id=%s,std_department=%s,std_name=%s,std_time=%s,std_date=%s,std_attendance=%s
where std_att_id =%s", (
                            self.var_id.get(),
                            self.var_dep.get(),
                            self.var_name.get(),
                            self.var_time.get(),
                            self.var_date.get(),
                            self.var_attend.get(),
                            self.var_attid.get()
                        ))
                except:
                    if not Update:
                        return
                    messagebox.showinfo("Success","Successfully Updated!",parent=self.root)
                    conn.commit()
                    self.fetch_data()
                    conn.close()
                except Exception as es:
                    messagebox.showerror("Error",f"Due to: {str(es)}",parent=self.root)
        # =====Delete Attendance form my
sql=====
        def delete_data(self):

```

```

if self.var_attid.get()=="":
    messagebox.showerror("Error","Student Id Must be Required!",parent=self.root)
else:
    try:
        delete=messagebox.askyesno("Delete","Do you want to Delete?",parent=self.root)
        if delete>0:
            conn = mysql.connector.connect(username='root',
password=" ",host='localhost',database='face_recognition',port=3306)
            mycursor = conn.cursor()
            sql="delete from stdattendance where std_att_id=%s"
            val=(self.var_attid.get(),)
            mycursor.execute(sql,val)
        else:
            if not delete:
                return

            conn.commit()
            self.fetch_data()
            conn.close()
            messagebox.showinfo("Delete","Successfully Deleted!",parent=self.root)
    except Exception as es:
        messagebox.showerror("Error",f"Due to: {str(es)}",parent=self.root)
# =====fatch data form mysql attendance=====

def fetch_data(self):
    conn = mysql.connector.connect(username='root',
password=" ",host='localhost',database='face_recognition',port=3306)
    mycursor = conn.cursor()

    mycursor.execute("select std_att_id, std_id, std_department, std_name, std_time,
std_date, std_attendance from stdattendance")
    data=mycursor.fetchall()

    if len(data)!= 0:
        self.attendanceReport.delete(*self.attendanceReport.get_children())
        for i in data:
            self.attendanceReport.insert("",END,values=i)
        conn.commit()
        conn.close()

#=====Reset Data=====
def reset_data(self):
    self.var_attid.set(""),
    self.var_id.set("")
    self.var_dep.set("")
    self.var_name.set("")
    self.var_time.set("")
    self.var_date.set("")
    self.var_attend.set("Status")

```

```

def refresh_data(self):
# Step 1: Clear existing data
    self.attendanceReport.delete(*self.attendanceReport.get_children())

    # Step 2: Fetch updated data (assuming self.fetch_data() does this)
    updated_data = self.fetch_data() # Replace this with your data-fetching logic

    # Step 3: Populate the Treeview with the new data
    for record in updated_data:
        self.attendanceReport.insert("", "end", values=record)

# =====Fetch Data Import data =====

def fetchData(self,rows):
    global mydata
    mydata = rows
    self.attendanceReport_left.delete(*self.attendanceReport_left.get_children())
    for i in rows:
        self.attendanceReport_left.insert("",END,values=i)
        print(i)

def importCsv(self):
    mydata.clear()
    fln=filedialog.askopenfilename(initialdir=os.getcwd(),title="Open
CSV",filetypes=(("CSV File","*.csv"),("All File","*.")),parent=self.root)
    with open(fln) as myfile:
        csvread=csv.reader(myfile,delimiter=",")
        for i in csvread:
            mydata.append(i)
        self.fetchData(mydata)

#=====Experot CSV=====
def exportCsv(self):

    file_path = "attendance.csv"
    if file_path:
        with open(file_path, "w", newline=""):
            pass

    file_path1= "a.csv"
    if file_path1:
        with open(file_path)

```

```

        messagebox.showinfo("Successfully", "CSV Cleared  
Successfully!",parent=self.root)

```

```

#=====Cursur Function for CSV=====

```

```

def get_cursor_left(self,event=""):
    cursor_focus = self.attendanceReport_left.focus()
    content = self.attendanceReport_left.item(cursor_focus)
    data = content["values"]

    self.var_id.set(data[0]),
    self.var_dep.set(data[1]),
    self.var_name.set(data[2]),
    self.var_time.set(data[3]),
    self.var_date.set(data[4]),
    self.var_attend.set(data[5])

```

```

#=====Cursur Function for mysql=====

```

```

def get_cursor_right(self,event=""):
    cursor_focus = self.attendanceReport.focus()
    content = self.attendanceReport.item(cursor_focus)
    data = content["values"]

    self.var_attid.set(data[0])
    self.var_id.set(data[1]),
    self.var_dep.set(data[2]),
    self.var_name.set(data[3]),
    self.var_time.set(data[4]),
    self.var_date.set(data[5]),
    self.var_attend.set(data[6])

```

```

#=====Update  
CSV=====

```

```

    # export upadte
    import csv
    import mysql.connector

```

```

def action(self):
    try:
        conn = mysql.connector.connect(
            username='root',
            password="",
            host='localhost',
            database='face_recognition',
            port=3306
        )
        mycursor = conn.cursor()

```

```

with open("attendance.csv", "r") as csvfile:
    reader = csv.reader(csvfile)
    # Skip the header row if present

    for row in reader:
        print(row)
        std_id, std_department, std_name, std_time, std_date, std_attendance = row

        mycursor.execute("INSERT INTO stdattendance (std_id, std_department,
std_name, std_time, std_date, std_attendance) VALUES (%s, %s, %s, %s, %s, %s)", (
            std_id,
            std_department,
            std_name,
            std_time,
            std_date,
            std_attendance
        ))

        conn.commit() # Commit inside the loop for each row

    self.fetch_data() # Fetch data after all rows are inserted
    conn.close()
    messagebox.showinfo("Success", "All Records are Saved in the Database!",
parent=self.root)
    except Exception as es:
        messagebox.showerror("Error", f"Due to: {str(es)}", parent=self.root)

if __name__ == "__main__":
    root=Tk()
    obj=Attendance(root)
    root.mainloop()

```

Issue.py

```

import re
from sys import path
from tkinter import*
from tkinter import ttk
from PIL import Image,ImageTk
import os
import mysql.connector
import cv2
import numpy as np
from tkinter import messagebox
from time import strftime
from datetime import datetime
import csv

```

```

from tkinter import filedialog
import tkinter as tk

#Global variable for importCsv Function
mydata=[]
class Issue:

    def __init__(self,root):
        self.root=root
        self.root.geometry("1600x768+0+0")
        self.root.state('zoomed')
        self.root.title("Attendance Pannel")

        #-----Variables-----
        self.var_id=StringVar()
        self.var_dep=StringVar()
        self.var_name=StringVar()
        self.var_time=StringVar()
        self.var_date=StringVar()
        self.var_attend = tk.StringVar(value="")

        # This part is image labels setting start
        # first header image
        img=Image.open(r"C:\Users\91628\Desktop\Face-Recognition-Attendance1\Images_GUI\banner.jpg")
        img=img.resize((1600,130),Image.ANTIALIAS)
        self.photoimg=ImageTk.PhotoImage(img)

        # set image as lable
        f_lb1 = Label(self.root,image=self.photoimg)
        f_lb1.place(x=0,y=0,width=1600,height=130)

        # backgorund image
        bg1=Image.open(r"Images_GUI/bg2.jpg")
        bg1=bg1.resize((1600,768),Image.ANTIALIAS)
        self.photobg1=ImageTk.PhotoImage(bg1)

        # set image as lable
        bg_img = Label(self.root,image=self.photobg1)
        bg_img.place(x=0,y=130,width=1600,height=768)

        #title section
        title_lb1 = Label(bg_img,text="Welcome to Feedback
Pannel",font=("verdana",30,"bold"),bg="white",fg="navyblue")
        title_lb1.place(x=0,y=0,width=1600,height=45)

        #=====Section
Creating=====

```



```

# Creating Frame
main_frame = Frame(bg_img,bd=2,bg="white") #bd mean border
main_frame.place(x=5,y=70,width=1520,height=550)

# Left Label Frame
left_frame = LabelFrame(main_frame,bd=2,bg="white",relief=RIDGE,text="Issue
Details",font=("verdana",12,"bold"),fg="navyblue")
left_frame.place(x=40,y=10,width=700,height=300)

# =====Text boxes and Combo
Boxes=====

#Student id

studentId_id = Label(left_frame,text="Issue
Id:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
studentId_id.grid(row=0,column=0,padx=5,pady=5,sticky=W)

student_entry =
tk.Entry(left_frame,textvariable=self.var_id,width=15,font=("verdana",12,"bold"))
student_entry.grid(row=0,column=1,padx=5,pady=5,sticky=W)

studentId_label =
Label(left_frame,text="Response:",font=("verdana",12,"bold"),fg="navyblue",bg="white")
studentId_label.grid(row=1 ,column=0,padx=5,pady=5,sticky=W)

self.studentId_entry = tk.Text(left_frame, width=20, height=5, font=("verdana", 12,
"bold"))
self.studentId_entry.grid(row=1, column=1, padx=5, pady=5, sticky="w")

# =====Table Sql Data
View=====

# =====button section=====

#Button Frame
btn_frame = Frame(left_frame,bd=0,bg="white",relief=RIDGE)
btn_frame.place(x=10,y=150,width=635,height=60)

```

```

#Improt button
save_btn=Button(btn_frame,command=self.escalate_data,text="Send",width=12,font=("
verdana",12,"bold"),fg="white",bg="navyblue")
save_btn.grid(row=0,column=0,padx=6,pady=10,sticky=W)


#reset button
reset_btn=Button(btn_frame,command=self.reset_data,text="Reset",width=12,font=("ve
rdana",12,"bold"),fg="white",bg="navyblue")
reset_btn.grid(row=0,column=2,padx=6,pady=10,sticky=W)


# Right
section=====

# Right Label Frame
right_frame = LabelFrame(main_frame,bd=2,bg="white",relief=RIDGE,text="Issue
Details",font=("verdana",12,"bold"),fg="navyblue")
right_frame.place(x=780,y=10,width=660,height=500)


# -----Table Frame-----
#Table Frame
#Searching System in Right Label Frame
table_frame = Frame(right_frame,bd=2,bg="white",relief=RIDGE)
table_frame.place(x=10,y=70,width=635,height=360)


#scroll bar
scroll_x = ttk.Scrollbar(table_frame,orient=HORIZONTAL)
scroll_y = ttk.Scrollbar(table_frame,orient=VERTICAL)


#create table
self.attendanceReport =
ttk.Treeview(table_frame,column=("Issue_ID","Student_ID","Name","Issue","Description","
Response"),xscrollcommand=scroll_x.set,yscrollcommand=scroll_y.set)


scroll_x.pack(side=BOTTOM,fill=X)
scroll_y.pack(side=RIGHT,fill=Y)
scroll_x.config(command=self.attendanceReport.xview)
scroll_y.config(command=self.attendanceReport.yview)


self.attendanceReport.heading("Issue_ID",text="Issue_ID")
self.attendanceReport.heading("Student_ID",text="Student_ID")
self.attendanceReport.heading("Name",text="Std-Name")
self.attendanceReport.heading("Issue",text="Issue")
self.attendanceReport.heading("Description",text="Date")
self.attendanceReport.heading("Response",text="Response")

```

```

self.attendanceReport["show"]="headings"

# Set Width of Columns
self.attendanceReport.column("Issue_ID",width=100)
self.attendanceReport.column("Student_ID",width=100)
self.attendanceReport.column("Name",width=100)
self.attendanceReport.column("Issue",width=100)
self.attendanceReport.column("Description",width=100)
self.attendanceReport.column("Response",width=100)

self.attendanceReport.pack(fill=BOTH,expand=1)
self.attendanceReport.bind("<ButtonRelease>",self.get_cursor_right)
self.fetch_data()
# =====update for mysql
button=====

# =====update function for mysql
database=====
import mysql.connector
from tkinter import messagebox

def escalate_data(self):
    if self.var_id.get() == "":
        messagebox.showerror("Error", "Please Fill All Fields are Required!",
parent=self.root)
    else:
        try:
            Update = messagebox.askyesno("Response", "Do you want to Respond this
Issue?", parent=self.root)
            if Update:
                response_text = self.studentId_entry.get("1.0", END) # Get the text from the
Text widget
                conn = mysql.connector.connect(user='root', password="", host='localhost',
database='face_recognition', port=3306)
                mycursor = conn.cursor()

                update_query = "UPDATE issue SET Response=%s WHERE Is_id=%s"
                data = (response_text.strip(), self.var_id.get()) # Use response_text.strip() to
remove leading/trailing whitespaces
                mycursor.execute(update_query, data)

                conn.commit()
                mycursor.close()
                conn.close()

                messagebox.showinfo("Success", "Successfully Replied!", parent=self.root)
                self.fetch_data() # Assuming this function fetches and updates the data display
            else:
                return

```

```

except Exception as es:
    messagebox.showerror("Error", f"Due to: {str(es)}", parent=self.root)

# =====Delete Attendance form my
sql=====

# =====fatch data form mysql attendance=====

def fetch_data(self):
    conn = mysql.connector.connect(username='root',
password=" ",host='localhost',database='face_recognition',port=3306)
    mycursor = conn.cursor()

    mycursor.execute("SELECT * FROM issue")
    data=mycursor.fetchall()

    if len(data)!= 0:
        self.attendanceReport.delete(*self.attendanceReport.get_children())
        for i in data:
            self.attendanceReport.insert("",END,values=i)
        conn.commit()
        conn.close()

#=====Reset Data=====
def reset_data(self):

    self.var_attend.set("")

# =====Fetch Data Import data =====

def fetchData(self,rows):
    global mydata
    mydata = rows
    self.attendanceReport_left.delete(*self.attendanceReport_left.get_children())
    for i in rows:
        self.attendanceReport_left.insert("",END,values=i)
        print(i)

#=====Cursur Function for CSV=====

#=====Cursur Function for mysql=====

def get_cursor_right(self, event=""):
    cursor_focus = self.attendanceReport.focus()

```

```

content = self.attendanceReport.item(cursor_focus)
data = content["values"]

self.var_id.set(data[0])
self.var_dep.set(data[1])
self.var_name.set(data[2])
self.var_time.set(data[3])
self.var_date.set(data[4])

# Instead of setting self.var_attend to the widget, get its value
attend_text = data[5] if data[5] else "" # Check if data[5] is None, and set it to an empty
string if it is
self.var_attend.set(attend_text)

if __name__ == "__main__":
    root=Tk()
    obj=Issue(root)
    root.mainloop()

```

13.6 References

- S. S. Pawaskar and A. M. Chavan, "Face Recognition based Class Management and Attendance System," 2020 IEEE Bombay Section Signature Conference (IBSSC), Mumbai, India, 2020, pp. 180-185, doi: 10.1109/IBSSC51096.2020.9332212.
- Face Recognition based Attendance System using Haar Cascade and Local Binary Pattern Histogram Algorithm. Published in: 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184).
- Arjun Raj, M. Shoheb, K. Arvind and K. S. Chethan, "Face Recognition Based Smart Attendance System," 2020 International Conference on Intelligent Engineering and Management (ICIEM), London, UK, 2020, pp. 354-357, doi 10.1109/ICIEM48762.2020.9160184
- .Face Recognition: Understanding LBPH Algorithm | by Kelvin Salton do Prado | Towards Data Science