

STUDENT ATTENDANCE MANAGEMENT SYSTEM USING FACE RECOGNITION

*A Project Report
submitted in partial fulfilment of
the
requirements for the award of the degree of*

**Bachelor of Technology
In
Artificial Intelligence and Machine Learning**

by

P. Sai Teja (21H71A6140)

Under the esteemed guidance of

Dr. G. Sai Chaitanya Kumar

Head of The Department



Department of Artificial Intelligence

DVR & Dr. HS

MIC College of Technology
(Autonomous)

Kanchikacherla– 521180, NTR Dist., Andhra Pradesh

March 2025

DECLARATION

We here by declare that

- a. The work contained in this report is original and has been done by us under the guidance of our supervisor(s).
- b. This work has not been submitted to any other institute for obtaining any degree or diploma.
- c. We have followed the guidelines provided by the institute in preparing the report. We have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.
- d. Whenever we have used materials (data, theoretical analysis, figures, and text) from other sources, we have given due credit to them by citing in the text of the report and giving their details in the references.

Place: Kanchikacherla

Date:

PROJECT MEMBER :

P. Sai Teja (21H71A6140)



DVR & Dr. HS
MIC College of Technology
ISO 9001:2015 Certified Institute
(Approved by AICTE & Permanently Affiliated to JNTUK, Kakinada)
Kanchikacherla - 521180, NTR Dist., A.P., India
Phones : 08678 - 273535 / 94914 57799 / 73826 16824
E mail: office@mictech.ac.in, Website: www.mictech.edu.in



CERTIFICATE

This is to certify that the project report entitled “**STUDENT ATTENDANCE MANAGEMENT SYSTEM USING FACE RECOGNITION**” submitted by **P. Sai Teja(21H71A6140)** to the DVR& Dr. HS MIC College of Technology in partial fulfillment of the requirements for the award of the Degree Bachelor of Technology in **Artificial Intelligence and Machine Learning** is a bonafide record of work carried out by him/her under my/our guidance and supervision. The contents of this report, in full or in parts, have not been submitted to any other institute for the award of any degree.

Supervisor

Head of the Department

Principal

Internal Examiner

External Examiner

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant guidance and engagement crown all the efforts with success. We thank our college management and respected **Sri D. PANDURANGA RAO**, CEO for providing us the necessary infrastructure to carry out the project work.

We express our sincere thanks to our beloved Principal **Dr. T. Vamsee Kiran** who has been a great source of information for our work.

We would like to express our sincere thanks to **Dr. G. Sai Chaitanya Kumar**, Head, Department of AI for extending his support to carry on this project work.

We would like to thank my project supervisor, **Dr. G. Sai Chaitanya Kumar**, Head, Department of AI for the guidance and support, especially for the valuable ideas and knowledge shared to me throughout the Project.

We take this opportunity to express our thanks to one and all who directly or indirectly helped us in bringing this effort to present form.

Finally, my special thanks go to my family for their continuous support and help throughout my academic years and for their continual support and encouragement for the completion of the project.

CONTENTS

Title Page	I	
Declaration	II	
Certificate	III	
Acknowledgement	IV	
Contents	V	
List of Tables	VII	
List of Figures	VIII	
List of Abbreviations	IX	
Abstract	X	
Chapter 1	Introduction	1
	1.1 Introduction	1
	1.2 Motivation	2
	1.3 Problem Statement	3
	1.4 Objective of the project	4
Chapter 2	Literature Review	5 - 6
Chapter 3	Methodology	7 - 31
	3.1 Workflow of the Proposed Methodology	7 – 9
	3.2 Architecture	10 - 12
	3.3 Algorithms	13 - 15
	3.4 System Analysis	16 - 19
	3.5 System Specifications	20 - 21
	3.6 System Design	22 - 24

	3.7 UML Diagrams	25 - 31
Chapter 4	Software Installation	32 - 35
	4.1 Installing Visual Studio Code	32 - 33
	4.2 Installing XAMPP	34 - 35
Chapter 5	Implementation of PHP Website	36 - 59
	5.1 Project Codes	36 - 55
Chapter 6	Result Analysis	57 - 60
Chapter 7	Conclusion and Future Work	61 - 62
	References	63 - 64

LIST OF TABLES

Table	Title	Page
5.2.4	Database Tables	61

LIST OF FIGURES

Figure	Title	Page
3.1	Workflow of the proposed system	9
3.2	Architecture	10
3.3	Process Diagram	14
3.7(a)	Use case diagram	25
3.7(b)	Class Diagram	26
3.7(c)	Sequence Diagram	27
3.7(d)	Deployment Diagram	28
3.7(e)	Component Diagram	29
3.7(f)	Activity Diagram	30
4.1	VS Code Installation Page	31
4.2	XAMPP Download Page	33
5.2.4	Database Tables	61

LIST OF ABBREVIATIONS

VS Code	Visual Studio Code
XAMPP	Cross-Platform (X), Apache (A), MariaDB (M), PHP (P), and Perl (P)
HTML	Hypertext Markup language
CSS	Cascading Style Sheets
JS	JavaScript
UML	Unified Modelling Language
ER diagram	Entity Relationship Diagram
DBMS	Database Management System

ABSTRACT

The **Student Attendance Management System Using Face Recognition** is a web-based application designed to automate and enhance the traditional attendance tracking process in educational institutions. This system leverages **face recognition technology** to accurately record student attendance, eliminating the need for manual entry and reducing the risk of proxy attendance.

The system consists of multiple modules, including **Admin**, **Lecture**, and **Student** sections, each equipped with role-specific functionalities. The **Admin panel** allows for managing courses, students, lectures, and venues, while the **Lecture module** enables instructors to take attendance using real-time facial recognition. The system uses **Face-api.js** and deep learning models for face detection and recognition, ensuring high accuracy and efficiency.

Key features of the system include **student enrollment**, **lecture management**, **real-time attendance marking**, **attendance reports**, and **data export functionalities**. The backend is powered by **PHP and MySQL**, while the frontend utilizes **JavaScript, HTML, and CSS**. Additionally, security features such as session management and database authentication ensure data integrity and privacy.

This project aims to **streamline attendance tracking**, minimize human errors, and improve overall efficiency in educational institutions. By integrating face recognition technology, the system enhances security, ensures accuracy, and provides a **seamless, automated solution** for attendance management.

CHAPTER-1

INTRODUCTION

1.1 Introduction

In educational institutions, managing student attendance is a crucial yet time-consuming task. Traditional attendance methods, such as manual roll calls and RFID card systems, can be inefficient, prone to errors, and susceptible to proxy attendance. To address these challenges, we present the **Student Attendance Management System Using Face Recognition**, an automated and efficient approach leveraging facial recognition technology.

This system utilizes advanced deep learning models to identify students based on their facial features, ensuring accurate and tamper-proof attendance records. The project is designed with a multi-user environment, allowing administrators and lecturers to manage attendance seamlessly through a web-based interface. The system integrates a **real-time facial recognition module**, database connectivity, and a structured user authentication mechanism to provide a secure and efficient solution.

The **Student Attendance Management System** is built using **PHP, JavaScript, MySQL**, and **face-api.js**, among other technologies. The **face recognition models** included in the project enable automatic detection, recognition, and storage of attendance records. This system enhances the overall efficiency of attendance tracking, minimizes administrative workload, and eliminates fraudulent attendance practices.

By implementing this solution, educational institutions can move towards a **more reliable, automated, and secure attendance management system**, ultimately improving student monitoring and engagement.

1.2 Motivation

Traditional methods of recording student attendance, such as manual roll calls and paper-based systems, are time-consuming, prone to errors, and vulnerable to proxy attendance. As educational institutions strive to improve efficiency, accuracy, and security, there is a pressing need for an automated and reliable solution. The **Student Attendance Management System Using Face Recognition** is designed to address these challenges by leveraging cutting-edge facial recognition technology to ensure seamless and accurate attendance tracking.

This project is motivated by the growing advancements in artificial intelligence and computer vision, which enable real-time facial recognition with high precision. By integrating face recognition into attendance management, this system minimizes human intervention, reduces administrative workload, and enhances overall security. Unlike traditional systems, which can be manipulated, this technology ensures that only the registered student is marked present, thereby eliminating fraudulent attendance practices.

Another driving factor for this project is the convenience it offers to both educators and students. In large classrooms, taking attendance manually can consume significant instructional time, reducing productivity. With an automated system, attendance is recorded within seconds, allowing educators to focus more on delivering quality education. Additionally, students benefit from a hassle-free experience, eliminating the need for ID cards or manual sign-ins.

The system also enhances data management and accessibility. Attendance records are securely stored in a database, reducing the risks of data loss, tampering, or misplacement. This digital approach enables real-time monitoring, generates attendance reports, and provides insights into student participation trends, which can be useful for academic analysis and decision-making.

By implementing this **Face Recognition-based Attendance System**, institutions can modernize their administrative processes, improve accuracy, and create a more secure and efficient learning environment. This project represents a step toward embracing technology-driven solutions that enhance educational administration while ensuring fairness and transparency.

1.3 Problem Statement

In educational institutions, tracking student attendance is a crucial task that ensures discipline, engagement, and academic performance. However, conventional attendance-taking methods, such as roll-calling or paper-based sign-in sheets, are inefficient, time-consuming, and prone to errors. These manual processes can also be easily manipulated, leading to proxy attendance, inaccurate records, and administrative burdens for educators.

The **Student Attendance Management System Using Face Recognition** aims to overcome these challenges by leveraging advanced facial recognition technology to automate the attendance process. This system ensures accurate, real-time student identification, eliminating the need for manual verification and reducing the risk of fraudulent attendance marking. By utilizing machine learning models and facial recognition algorithms, the system quickly detects and matches student faces with stored records, making the attendance process seamless and efficient.

Furthermore, the system provides a centralized digital attendance database that allows faculty and administrators to easily monitor, analyze, and retrieve attendance records when needed. This enhances transparency, minimizes errors, and facilitates better academic management. Additionally, the automated approach saves valuable classroom time, allowing educators to focus more on teaching rather than administrative tasks.

By implementing this face recognition-based attendance system, educational institutions can improve accuracy, security, and efficiency in attendance tracking, fostering a more streamlined and technology-driven academic environment.

1.4 Objective of the Project:

The **Student Attendance Management System Using Face Recognition** is designed to automate and streamline the attendance-taking process in educational institutions. Traditional methods of recording attendance, such as manual roll calls or ID card swiping, are time-consuming, prone to errors, and susceptible to fraudulent activities like proxy attendance. This project aims to leverage **face recognition technology** to provide a more accurate, efficient, and secure system for tracking student attendance.

The primary objective of this system is to **enhance accuracy and reliability** in attendance monitoring by using facial recognition, which eliminates the chances of impersonation or attendance falsification. The system captures students' facial features and matches them with stored records, ensuring that only authenticated individuals are marked present. This approach significantly reduces human intervention and automates attendance tracking, thereby **saving time for both students and instructors**.

Another key goal of this project is to **improve data management and accessibility**. The system maintains a centralized database where attendance records are securely stored and can be retrieved when needed. This ensures that faculty members and administrators can generate reports, analyze student attendance trends, and make data-driven decisions to enhance academic engagement. The system also provides an intuitive user interface, making it easy for educators to manage student records, create courses, and track attendance history seamlessly.

Additionally, the project aims to **enhance security and scalability** by integrating modern facial recognition models and encryption techniques. By utilizing advanced machine learning algorithms, the system ensures high accuracy in facial recognition, even under varying lighting conditions or slight facial changes. The architecture is designed to support scalability, allowing institutions of different sizes to implement and expand the system as needed.

Overall, the **Student Attendance Management System Using Face Recognition** enhances efficiency, accuracy, and security in attendance tracking. It reduces administrative workload, ensures a seamless experience for students and teachers, and provides a reliable solution for institutions aiming to modernize their attendance management processes.

CHAPTER-2

LITERATURE REVIEW

The adoption of biometric systems for attendance management has gained significant attention in recent years, particularly in educational institutions. Traditional methods of attendance tracking, such as manual roll calls and RFID-based systems, are often time-consuming, prone to errors, and susceptible to proxy attendance. To address these challenges, face recognition technology has emerged as an efficient and reliable solution for automating attendance management systems.

Face recognition technology utilizes advanced machine learning and artificial intelligence algorithms to identify and verify individuals based on their facial features. Various studies have demonstrated the efficacy of facial recognition in improving accuracy and reducing administrative burdens associated with attendance tracking. According to Zhao et al. (2003), face recognition techniques can achieve high levels of accuracy by leveraging deep learning models such as convolutional neural networks (CNNs). Similarly, Schroff et al. (2015) introduced the FaceNet model, which significantly improved the performance of face recognition through deep metric learning.

In educational environments, automated attendance systems based on facial recognition have been implemented to enhance efficiency and transparency. Jain et al. (2018) conducted a study on the implementation of a smart attendance system that used real-time face detection and recognition. The study concluded that such systems could effectively eliminate fraudulent attendance practices and streamline attendance tracking processes. Moreover, Priya et al. (2019) highlighted the integration of cloud-based face recognition systems, which allowed for remote access to attendance data, further improving the accessibility and usability of such systems.

The use of deep learning frameworks such as OpenCV, TensorFlow, and Dlib has significantly contributed to the advancement of face recognition applications. Studies by Parkhi et al. (2015) demonstrated that deep learning-based face recognition models could

outperform traditional hand-crafted feature-based techniques in terms of speed and accuracy. Additionally, the availability of large-scale datasets, such as the Labeled Faces in the Wild (LFW) and Microsoft's MS-Celeb-1M, has facilitated the training of robust models for face recognition tasks.

Despite its advantages, face recognition technology also presents certain challenges. Factors such as variations in lighting conditions, occlusions, and changes in facial expressions can affect recognition accuracy. To mitigate these issues, researchers have explored the use of hybrid models that combine multiple biometric modalities, such as face and voice recognition, to enhance system reliability (Abate et al., 2007). Moreover, ethical concerns related to data privacy and security have been widely discussed. Ensuring compliance with data protection regulations, such as the General Data Protection Regulation (GDPR), is crucial for maintaining user trust and safeguarding sensitive biometric data.

In conclusion, face recognition-based attendance management systems have proven to be a viable and efficient alternative to traditional methods. With continuous advancements in deep learning and computer vision, these systems are expected to become even more robust and widely adopted in educational institutions. Future research should focus on addressing the challenges of face recognition under varying environmental conditions and enhancing security measures to prevent unauthorized access to biometric data.

CHAPTER-3

METHODOLOGY

3.1 Workflow of the Proposed Methodology

The **Student Attendance Management System Using Face Recognition** follows a structured workflow to ensure seamless attendance tracking and management. The methodology consists of several key stages, including user authentication, face detection, recognition, attendance marking, and data management.

1. User Authentication and Role-Based Access

The system categorizes users into different roles such as **Admin, Lecturer, and Student**, each having distinct access privileges. The authentication process requires users to log in using their credentials. The admin manages student and lecturer records, while lecturers handle attendance-taking and reports.

2. Student Enrollment and Data Storage

Before attendance tracking, students must be enrolled in the system. The enrollment process involves capturing facial images of students and associating them with their unique ID in the database. This data is securely stored and used for future face recognition.

3. Face Detection and Recognition

During an attendance session, the system utilizes a **pre-trained face recognition model** to detect and recognize students' faces. The face detection algorithm scans and identifies facial features using the **Face API** and deep learning models. These models, stored under the "models" directory, include **face landmark detection, face recognition, and expression analysis**.

4. Attendance Marking Process

Once a student's face is successfully recognized, their attendance is automatically recorded in the database. The system logs details such as **student ID, name, course, date, and time of attendance**. If a student is not recognized or is absent, the system generates a corresponding record.

5. Attendance Data Storage and Management

The attendance records are stored in a **structured database** managed via the `dbcon.php` file. The database ensures real-time updating and retrieval of records. Lecturers can view, edit, and download attendance reports through **PHP interfaces** like `viewAttendance.php` and `downloadRecord.php`.

6. Report Generation and Exporting

The system provides functionalities to generate **attendance reports**, which can be downloaded in different formats such as **Excel or PDF**. Files related to exporting, such as `filesaver.js` and `xlsx.js`, facilitate smooth report generation.

7. Security and Session Management

To maintain security, the system employs **session management** through `session.php`, ensuring that only authorized users can access sensitive data. Additionally, logout functionalities (`logout.php`) prevent unauthorized access.

8. Scalability and Future Enhancements

The system is designed for scalability, allowing the addition of new features like **real-time monitoring, enhanced recognition accuracy, and integration with external systems**. The modular approach ensures flexibility in expanding functionalities as needed.

This workflow ensures an efficient, secure, and automated approach to student attendance tracking using advanced face recognition technology.

FACE RECOGNITION SYSTEM

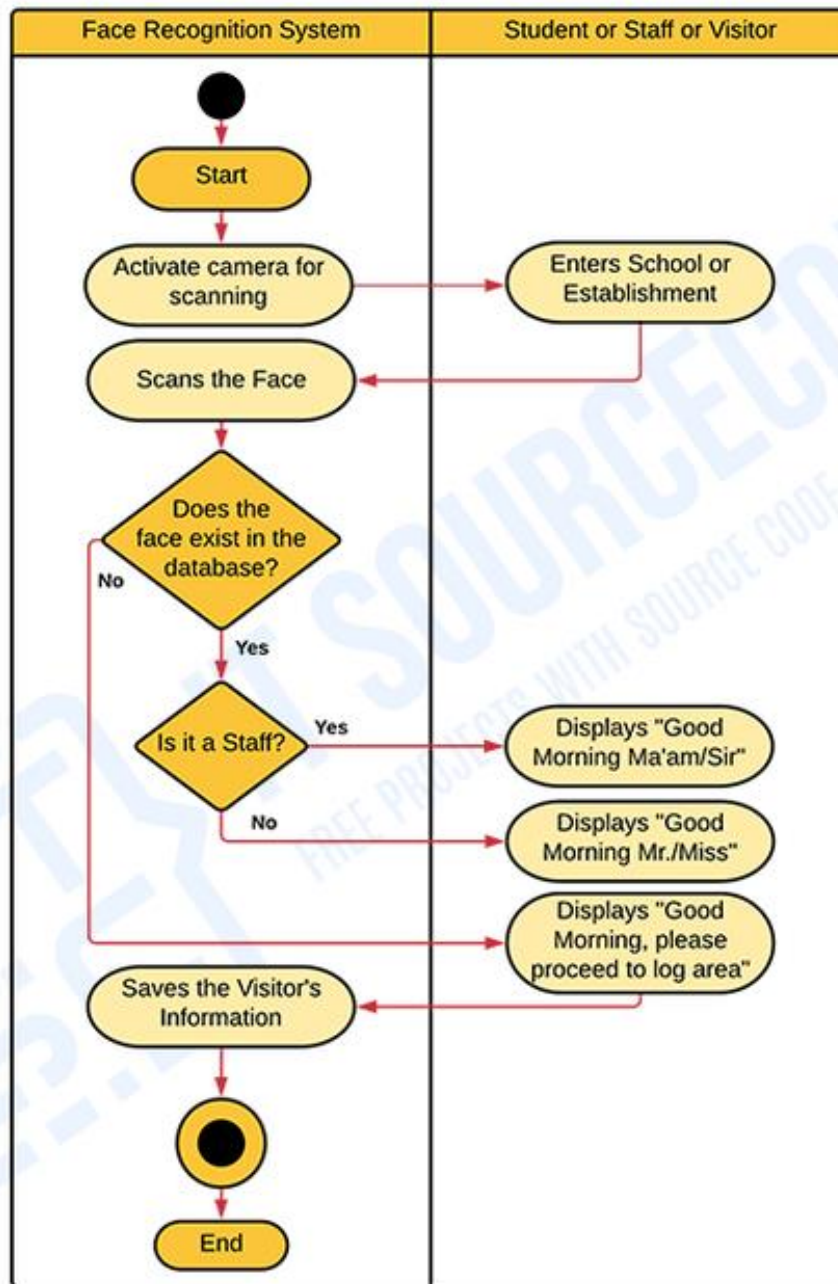


Fig 3.1 : Workflow of the proposed system

3.2 Architecture

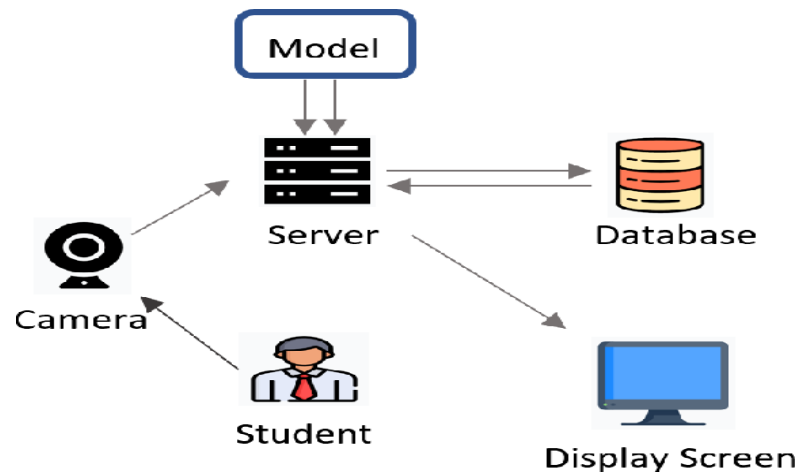


Fig 3.2 : Architecture

The **Student Attendance Management System using Face Recognition** is a web-based application designed to automate and enhance the process of student attendance tracking using facial recognition technology. The system is structured into different modules, each serving a specific role in the application, ensuring smooth and efficient functionality.

System Architecture

The system follows a **three-tier architecture**, consisting of the **Frontend (User Interface)**, **Backend (Business Logic)**, and **Database Layer**. It integrates technologies such as PHP, JavaScript, MySQL, and machine learning models for facial recognition.

1. Frontend (User Interface)

The user interface is built using HTML, CSS, and JavaScript to provide an interactive and intuitive experience for administrators, lecturers, and students. The **Admin panel** and **Lecture panel** are designed separately to allow role-based access. CSS files (`styles.css`, `loginStyle.css`) define the styling, while JavaScript files (`main.js`, `search.js`, `addStudent.js`, etc.) handle dynamic interactions such as searching and form validation.

2. Backend (Business Logic)

The backend of the system is developed using **PHP**, handling user authentication, data processing, and database interactions. The **Admin module** includes functionalities such as creating courses, lectures, and students, which are managed through dedicated PHP scripts (`createCourse.php`, `createStudent.php`, etc.). The **Lecture module** is responsible for student attendance management, including taking attendance (`takeAttendance.php`), managing records (`manageAttendance.js`), and downloading attendance data (`downloadRecord.php`).

3. Database Layer

The system relies on **MySQL** for data storage, structured using the `attendancesystem.sql` database. The database holds student records, course details, lecture schedules, and attendance logs. A centralized database connection (`dbcon.php`) ensures seamless interaction between the application and the database.

4. Face Recognition and Machine Learning Models

The core functionality of the system is enabled through **face recognition models** housed in the `models` directory. These include models for **face detection**, **landmark recognition**, **age and gender estimation**, and **face recognition** itself. The system leverages **Face-API.js**, a JavaScript-based face recognition library, to detect and identify students in real-time using their webcam. Models such as **SSD MobileNetV1**, **MTCNN**, and **Tiny Face Detector** ensure accurate facial feature extraction and recognition.

5. Security and Authentication

The system incorporates authentication and session management (`session.php`) to ensure that only authorized users can access specific functionalities. A login system is implemented with encrypted credentials for security. The `passwords.txt` file suggests a temporary storage mechanism that should be replaced with a more secure alternative like database-stored hashed passwords.

6. File and Attendance Management

Attendance data is recorded and stored securely, allowing administrators and lecturers to retrieve and analyze records through interfaces like `viewAttendance.php` and `studentTable.php`. Additional functionalities such as **uploading attendance records** (`attendanceUpload.php`), **managing attendance data** (`manageFolder.php`), and **exporting data to Excel** (using `xlsx.js` and `filesaver.js`) enhance the usability of the system.

Conclusion

The **Student Attendance Management System using Face Recognition** offers an efficient and automated method of tracking student attendance using machine learning and facial recognition. By integrating **a structured frontend, robust backend, and AI-powered recognition models**, the system ensures accuracy, efficiency, and security in attendance management. The modular design allows easy scalability, making it adaptable for various educational institutions.

3.3 Algorithms

Algorithms for Student Attendance Management System Using Face Recognition

1. Face Detection Algorithm

The system first detects the faces present in the camera frame using a deep learning-based face detection model. The **Single Shot MultiBox Detector (SSD) with MobileNetV1** or **Multi-task Cascaded Convolutional Network (MTCNN)** is used to efficiently detect faces in real time. These models process the image and extract bounding boxes around detected faces.

2. Face Alignment and Preprocessing

Once a face is detected, the system aligns and normalizes it for better recognition accuracy. This is done using **Facial Landmark Detection**, where models such as **Face Landmark 68 Model** identify key facial points (eyes, nose, mouth, and jawline). The detected face is then cropped, resized, and converted into grayscale or normalized using pixel intensity adjustments.

3. Feature Extraction and Face Embedding

After preprocessing, the system extracts unique facial features using a **Face Recognition Model** like **FaceNet** or **Dlib's ResNet-based model**. These models convert the face image into a numerical representation (embedding) that captures distinctive features. The embeddings are then compared against stored embeddings in the database.

4. Face Matching and Recognition

The extracted embeddings are compared with the stored embeddings of registered students using a **Cosine Similarity** or **Euclidean Distance** algorithm. A threshold is set to determine if a match is found. If the distance between embeddings is below the threshold, the student is identified and marked as present.

5. Attendance Logging and Database Update

Upon successful recognition, the system logs the attendance in a database. The system records the student's ID, name, date, and time of attendance. The database update can be done using **SQL queries in PHP (MySQL database)** to ensure real-time record maintenance.

6. Liveness Detection (Optional)

To prevent spoofing attacks using printed photos or videos, the system can incorporate liveness detection techniques. This can include **blink detection, head movement analysis, or texture analysis using Convolutional Neural Networks (CNNs)**.

7. Data Storage and Export

The attendance records can be stored and exported in different formats such as Excel (using **XLSX.js**) or CSV for administrative purposes. The system allows users to download attendance records and manage them via a web interface.

8. Error Handling and Retraining

If a student is not recognized, the system can prompt re-registration or retraining of the model by collecting new face images. This ensures improved accuracy over time and minimizes false negatives.

By integrating these algorithms, the face recognition-based student attendance system provides a seamless, automated, and accurate method for tracking student attendance while reducing manual intervention.

FACE RECOGNITION ATTENDANCE SYSTEM

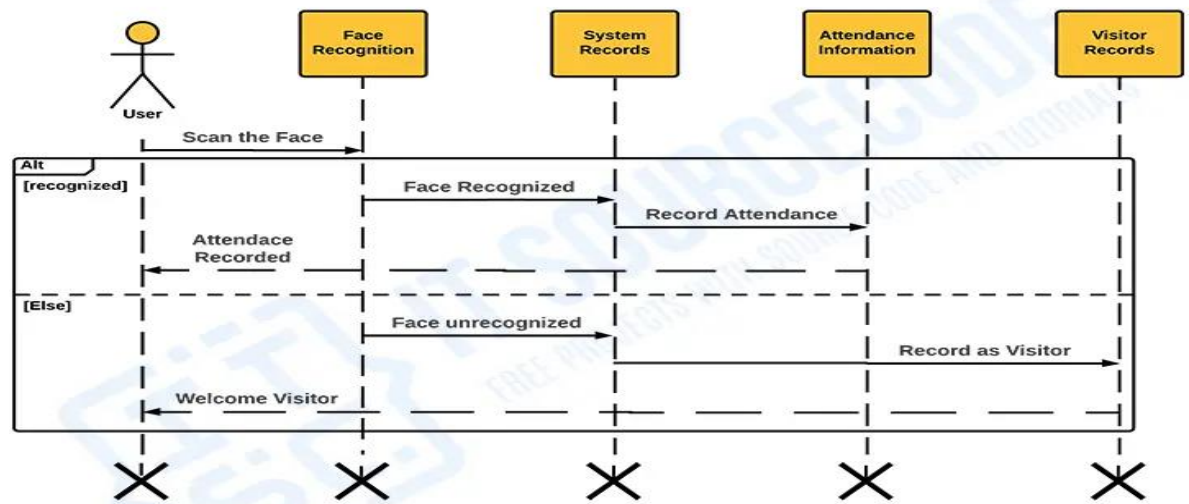


Fig 3.3: Process Diagram

3.4 SYSTEM ANALYSIS

Overview

The **Student Attendance Management System using Face Recognition** is an automated solution designed to track student attendance efficiently and accurately. This system eliminates the traditional manual methods such as roll calls and paper-based records by leveraging face recognition technology. It utilizes deep learning-based **Convolutional Neural Networks (CNNs)** and **OpenCV** for facial detection and recognition. The system captures student images, processes them, and automatically marks attendance based on facial recognition.

Advantages

- **Accurate and Reliable** – Reduces human errors and proxy attendance.
- **Time-Efficient** – Automates attendance marking, saving time for instructors.
- **Secure and Fraud-Proof** – Prevents fake attendance using facial recognition.
- **Easy Access and Management** – Digital records improve accessibility and tracking.
- **Integration with Databases** – Ensures seamless data retrieval and reporting.

Feasibility Study

The feasibility study determines whether the proposed system is viable for implementation. It assesses three key factors:

1. Economic Feasibility

This system is cost-effective as it primarily uses open-source technologies such as:

- **OpenCV and Python** for image processing.
- **Face-API.js** for real-time facial recognition.
- **MySQL or Firebase** for database management.

- **Minimal hardware requirements**, utilizing webcams for image capture. Since most of these technologies are open-source, development costs are low, and only hardware investments (cameras and servers) may be needed.

2. Technical Feasibility

The system is technically feasible as it utilizes **widely available** and **well-documented** technologies. The required infrastructure includes:

- A **web-based** or **desktop** application interface.
- **AI-driven face recognition** models (pre-trained or custom-trained CNN models).
- Integration with **existing attendance management systems** and **student databases**. The system requires minimal computational resources and can be implemented without overhauling existing infrastructure.

3. Social Feasibility

The system is designed to be **user-friendly** and does not impose a learning curve on students and faculty. The automation of attendance marking ensures a **hassle-free** experience.

- **Teachers** benefit from reduced administrative workload.
- **Students** enjoy a seamless attendance process without manual intervention.
- The system promotes **fairness** by preventing proxy attendance, ensuring only **physically present** students are marked.

Functional and Non-Functional Requirements

Functional Requirements

These define the core functionalities of the system:

1. Student Registration & Database Management

- Ability to **register** students with facial data.
- Storage and retrieval of student records.
- **Deletion and modification** of records if necessary.

2. Face Recognition for Attendance Marking

- Real-time **face detection and recognition**.
- Automatic attendance marking **based on student presence**.
- **Error handling** in case of incorrect recognition.

3. User Authentication & Role Management

- **Admin access** for managing student and faculty records.
- **Instructor access** for viewing and modifying attendance data.
- **Student access** for viewing their own attendance records.

4. Attendance Reports & Data Management

- Generation of **attendance reports** (daily, weekly, monthly).
- Exporting reports in **Excel, CSV, or PDF** format.
- Integration with **existing student information systems**.

5. Notification System

- Sending **alerts to students** for low attendance.
- **Email/SMS notifications** for administrators in case of anomalies.

Non-Functional Requirements

These define the overall system performance and behavior:

1. Performance

- High accuracy in **face recognition** under different lighting conditions.
- Quick attendance marking with **minimal latency**.

2. Scalability

- Ability to handle **large student populations** without degradation.
- Support for **multiple classrooms and venues**.

3. Usability

- **Simple and intuitive interface** for students and faculty.
- Mobile-friendly and responsive design for **access on different devices**.

4. Security & Privacy

- **Encryption** of student data and facial recognition models.
- Compliance with **privacy laws** (e.g., GDPR for biometric data).

5. Maintainability & Support

- **Easy updates and model improvements** as needed.
- Availability of **user manuals and support documentation**.

6. Compatibility & Integration

- Works across **different browsers and devices**.
- Integrates with **existing student management systems** (e.g., LMS, university portals).

3.5 System Specifications

Hardware Requirements

- **Processor:** Intel Core i3 or higher
- **Hard Disk:** Minimum 160GB
- **Keyboard:** Standard Windows Keyboard
- **Mouse:** Two or Three-Button Mouse
- **Monitor:** SVGA or higher
- **RAM:** Minimum 8GB (Recommended 16GB for better performance)
- **Webcam:** HD Webcam for Face Recognition

Software Requirements

- **Operating System:** Windows 7/8/10/11 or Linux (Ubuntu 18.04+)
- **Front-End Technologies:** HTML, CSS, Bootstrap, JavaScript
- **Programming Language:** Python
- **Back-End Technologies:** Flask Framework
- **Libraries Used:**
 - OpenCV (for image processing and face detection)
 - Face-API.js (for real-time face recognition in the browser)
 - NumPy (for numerical computations)
 - Pandas (for data manipulation)
 - MySQL Connector (for database connectivity)
 - smtplib (for email notifications)
 - OS module (for file handling)
- **Database:** MySQL
- **Server Deployment:** XAMPP Server
- **IDE/Workbench:** PyCharm, VS Code, or any Python-supporting IDE
- **Web Server:** Apache (through XAMPP)

Additional Requirements

- **Browser Support:** Google Chrome, Mozilla Firefox, Edge (latest versions)
- **Network:** Localhost or Cloud-based Hosting (optional)
- **Security Measures:**
 - Secure password storage
 - Authentication & session management
 - Data encryption for face recognition models

3. 6 SYSTEM DESIGN

Input Design

The input design focuses on how data is collected, processed, and prepared for use by the system. Key aspects include:

Data Collection Interface:

- **User-Friendly Interfaces:**
 - Web-based and mobile-friendly forms for administrators to input student details, course details, and lecture schedules.
 - Camera-based real-time image capture for student attendance authentication.
- **Integration for Faculty:**
 - Faculty members can manually add or edit attendance records if needed.
 - Interface for uploading student lists via CSV or Excel.

Image and Face Recognition Input:

- **Facial Image Capture:**
 - Integration with webcam or external camera modules for real-time image capture.
 - Option to upload student photos for initial database enrollment.
- **Preprocessing Steps:**
 - Face detection and alignment using deep learning models.
 - Image enhancement techniques to improve recognition accuracy.

Data Preprocessing:

- **Automated Checks:**
 - System ensures images are clear and faces are properly detected before storing data.
 - Validation of student details before adding to the database.
- **Normalization and Encryption:**
 - Standardization of face recognition features.
 - Secure encryption of sensitive data during storage and transmission.

Integration with External Systems:

- **Database Synchronization:**
 - Integration with student information management systems (SIMS) for importing student records.
 - APIs to retrieve and sync attendance data with Learning Management Systems (LMS).

Mobile and Web Applications:

- **Cross-Platform Compatibility:**
 - Mobile-friendly and responsive UI for attendance marking and reporting.
 - Faculty and admin access via web and mobile portals.
- **Real-Time Data Capture:**
 - Instant face recognition attendance marking using mobile devices with cameras.

Output Design

The output design focuses on how attendance records and reports are presented to users.

Attendance Reports:

- **Detailed Attendance Logs:**
 - Time-stamped attendance records for each student.
 - Real-time status updates for class attendance.
- **Graphical Representation:**
 - Charts and graphs to analyze student attendance trends over time.

Alerts and Notifications:

- **Automated Alerts:**
 - Notifications to students for missed attendance.
 - Reminders to faculty about pending attendance approvals.
- **Admin Alerts:**
 - Automated warnings for attendance shortfalls.
 - Alerts for possible fraudulent attempts (e.g., duplicate face detection).

User Dashboard:

- **Interactive Features:**
 - Visual dashboard for tracking attendance records.
 - Drill-down features to view attendance by student, class, or date range.

Export and Sharing Options:

- **Multiple Formats:**
 - Export attendance reports in CSV, Excel, and PDF formats.
 - Secure access for stakeholders like parents or administrators.

Data Visualization:

- **Advanced Analysis Tools:**
 - Heatmaps and trends for attendance analysis.
 - AI-based predictive analysis for identifying at-risk students.

Accessibility Features:

- **User-Friendly Access:**
 - Screen reader support and high-contrast mode for visually impaired users.
 - Role-based access control to ensure data security.

By integrating face recognition technology with a robust input and output design, this system ensures accurate and efficient attendance tracking while maintaining user-friendliness and data security.

3. 7 UML DIAGRAMS:

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of objectoriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modelling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

USE CASE DIAGRAM:

- ◆ A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.
- ◆ Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.
- ◆ The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

FACE RECOGNITION ATTENDANCE SYSTEM

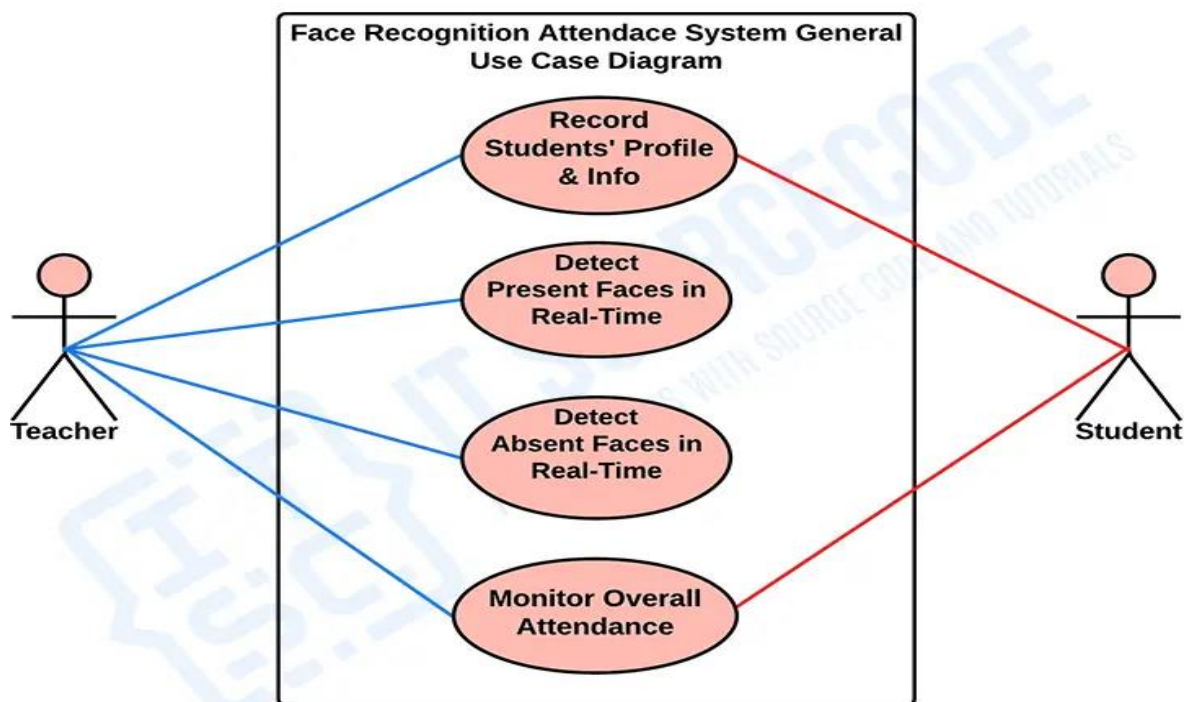


Fig 3.7(a) : Usecase Diagram

CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information

FACE RECOGNITION ATTENDANCE SYSTEM

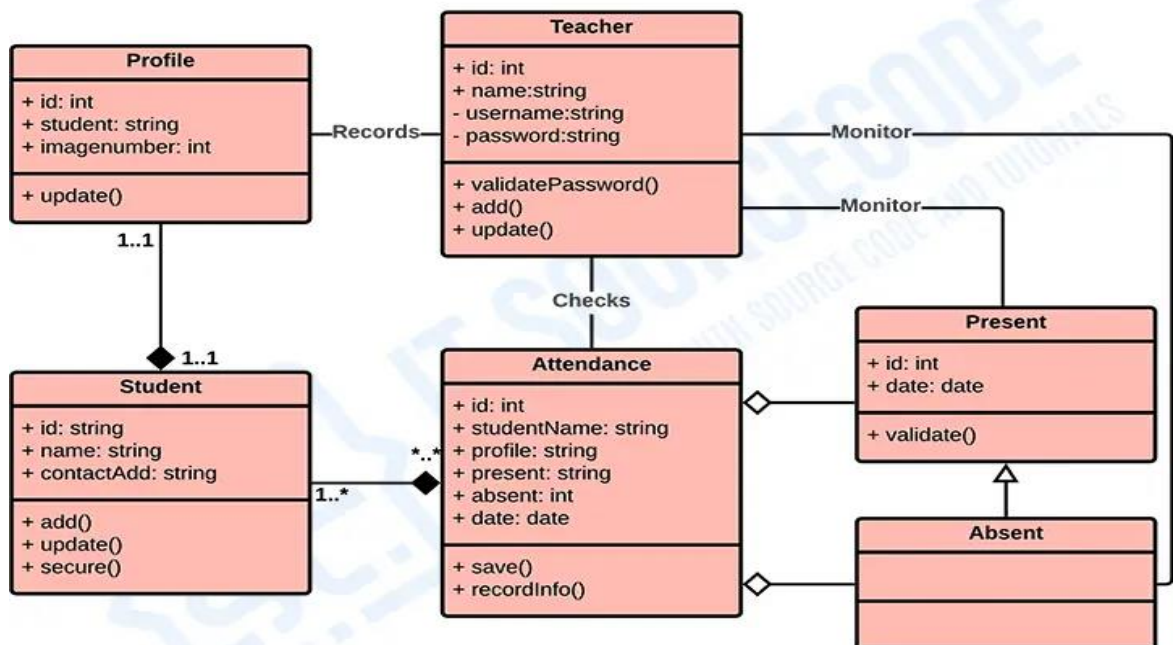


Fig 3.7(b) : Class Diagram

Sequence Diagram:

- ◆ A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order.
- ◆ It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams

FACE RECOGNITION ATTENDANCE SYSTEM

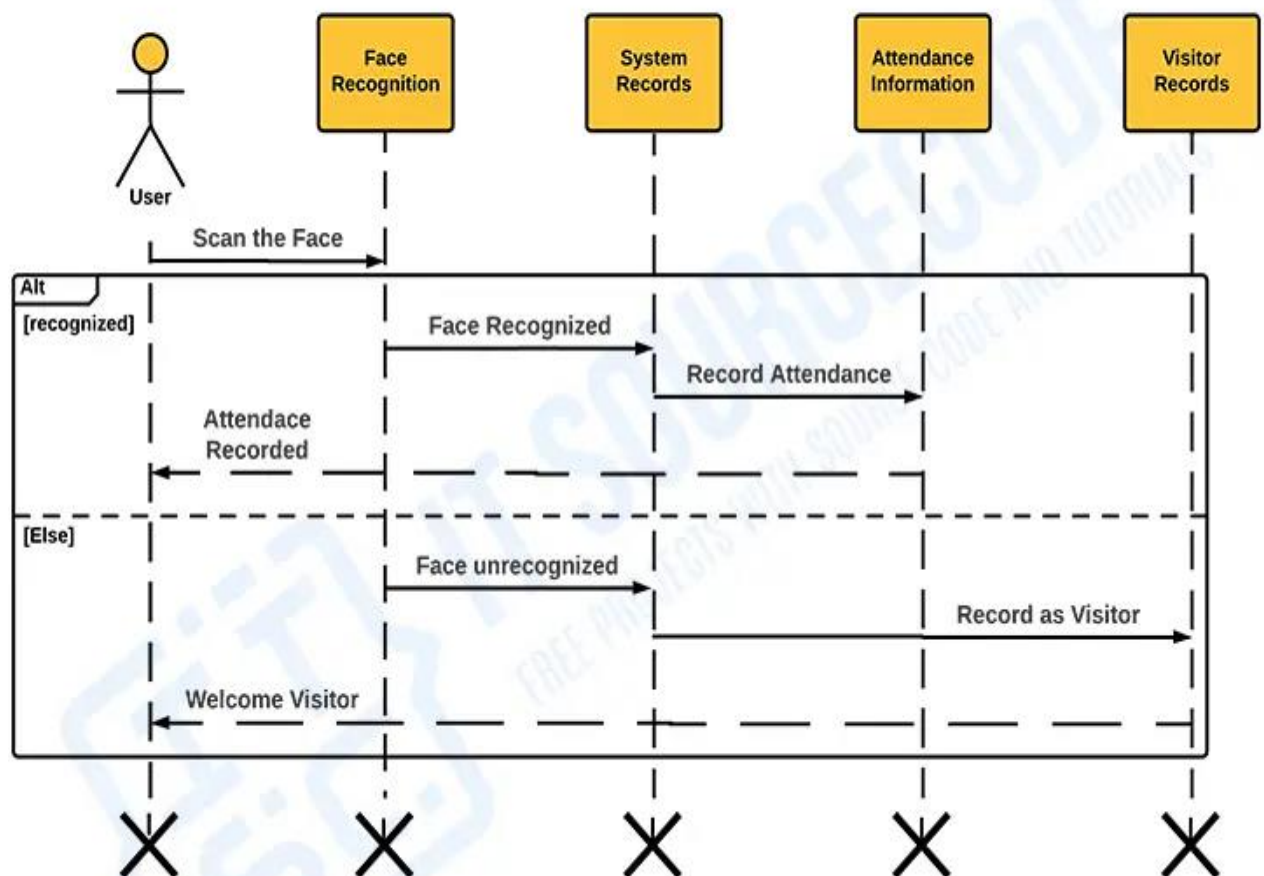


Fig 3.7(c): Sequence Diagram

Deployment Diagram

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware's used to deploy the application.

FACE RECOGNITION ATTENDANCE SYSTEM

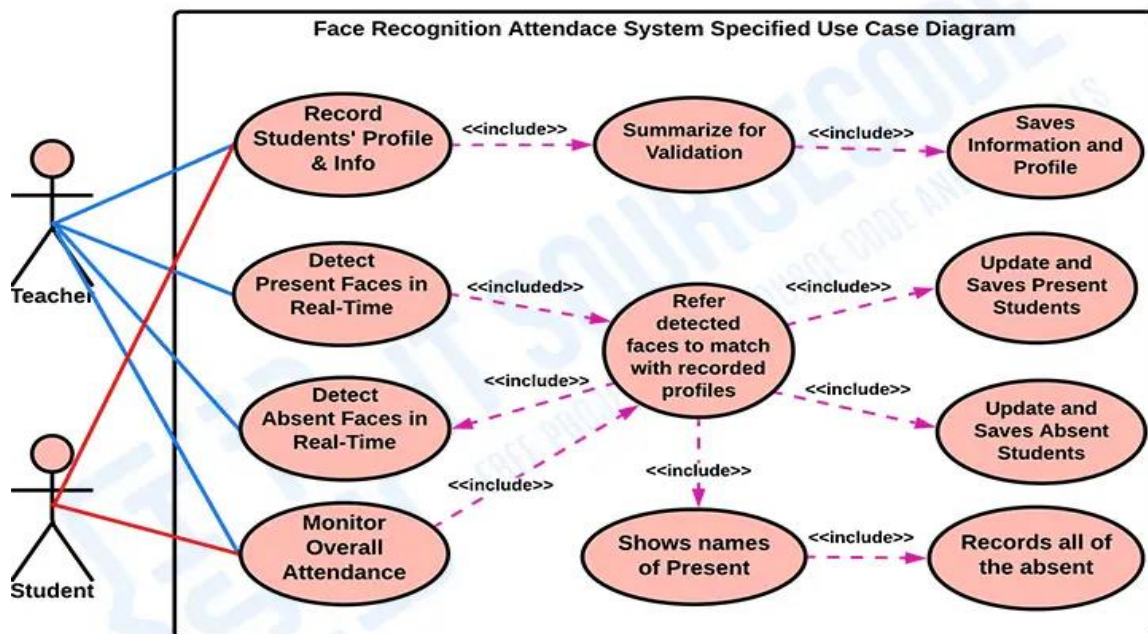


Fig 3.7(d): Deployment Diagram

Component Diagram:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required function is covered by planned development.

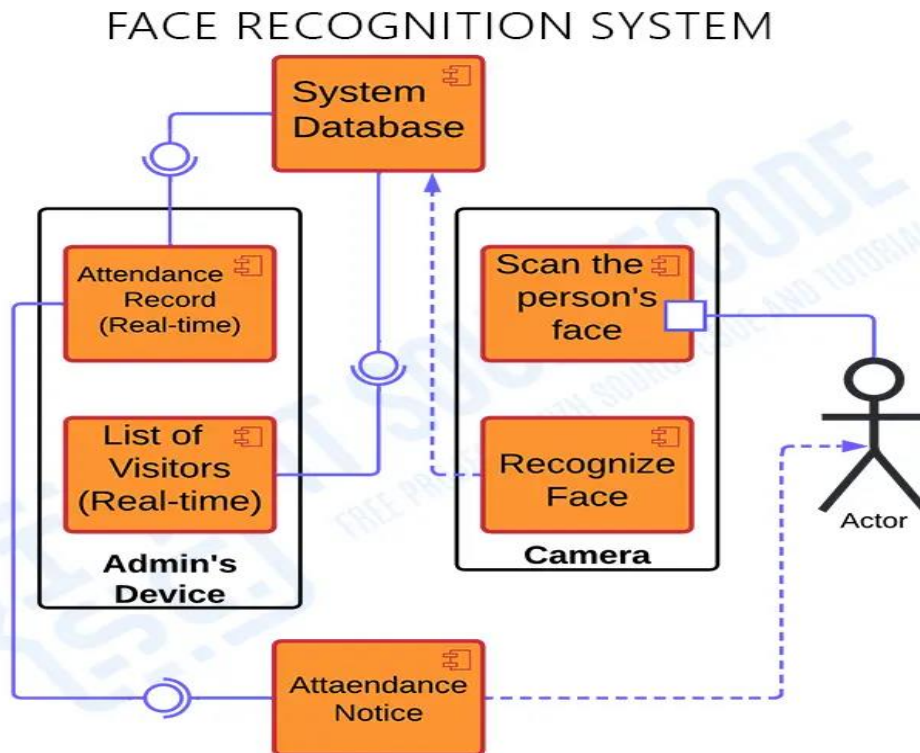


Fig 3.7(e): Component Diagram

Activity Diagram:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

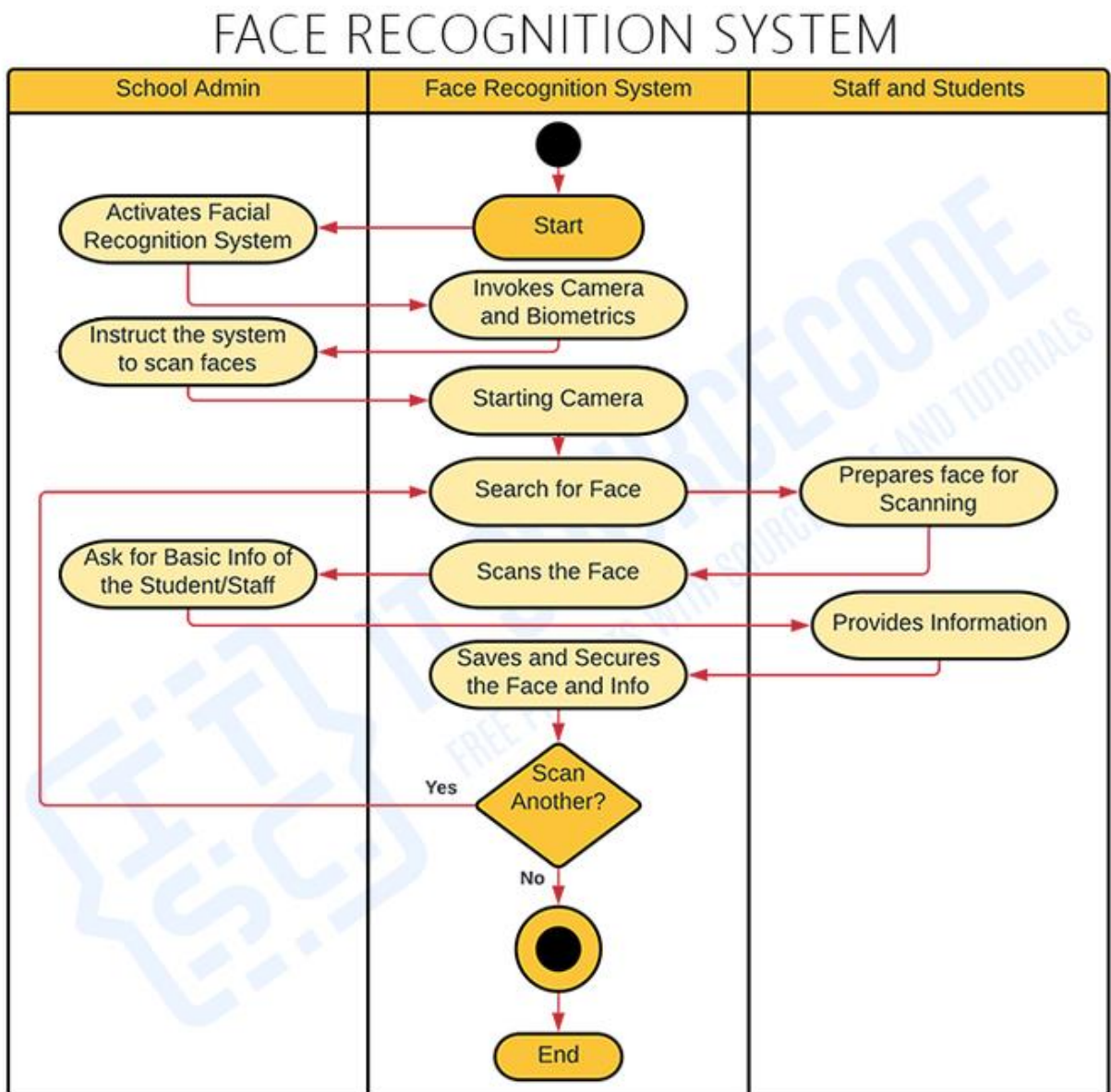


Fig 3.7(f): Activity Diagram

CHAPTER-4

SOFTWARE INSTALLATION

Installing VS Code is pretty straightforward. Here's how you can do it:

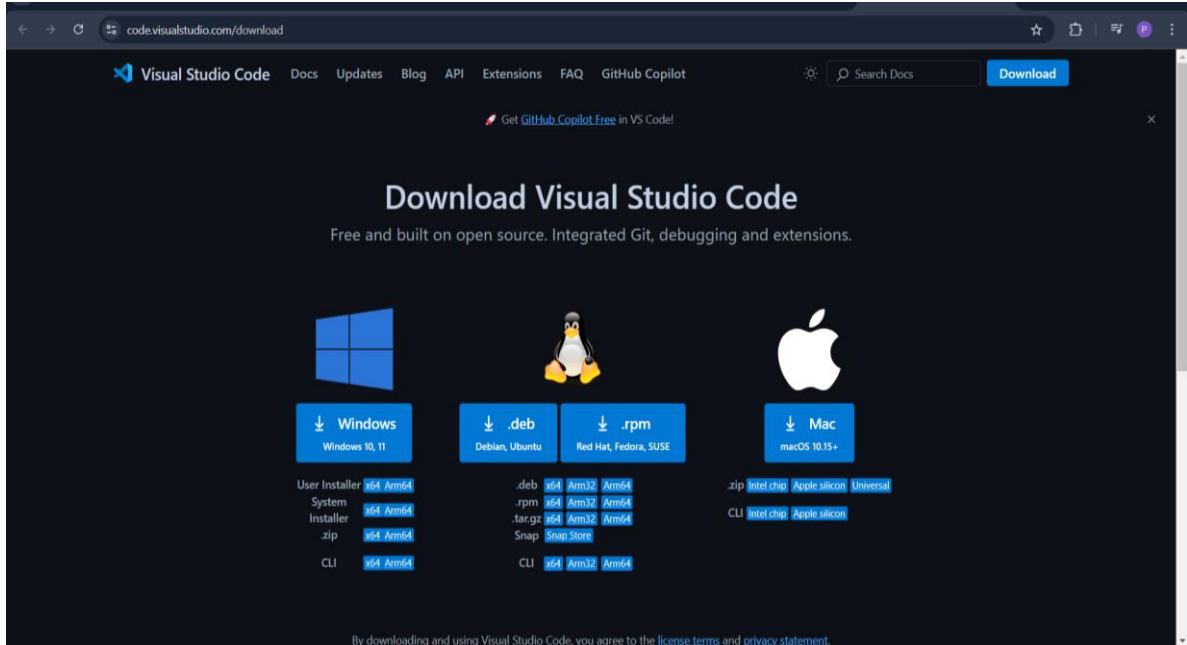


Fig 4.1: Installation of VS Code

For Windows:

1. Go to the official website: <https://code.visualstudio.com/>
2. Click **Download for Windows** (It will download the latest stable version).
3. Open the downloaded .exe file.
4. Follow the installation wizard:
 - o Accept the license agreement.
 - o Choose installation location.
 - o Select options (I recommend enabling "Add to PATH" and "Open with Code" options).
5. Click **Install** and wait for it to finish.
6. Once installed, launch VS Code.

For macOS:

1. Visit: <https://code.visualstudio.com/>
2. Download the macOS version (.zip file).
3. Open the downloaded file and drag Visual Studio Code.app into the Applications folder.
4. Open VS Code from Launchpad or Finder.

(Optional: Set up command-line access by running this in Terminal:

```
sudo ln -s "/Applications/Visual Studio  
Code.app/Contents/Resources/app/bin/code" /usr/local/bin/code)
```

For Linux (Debian/Ubuntu-based distros):

1. Open Terminal and run:

```
css  
CopyEdit  
sudo apt update  
sudo apt install code
```

or manually download the .deb package from [the official site](#) and install it via:

```
css  
CopyEdit  
sudo dpkg -i <filename>.deb
```

4.2 Installing XAMPP:

Installing XAMPP is a straightforward process. Here's a step-by-step guide:

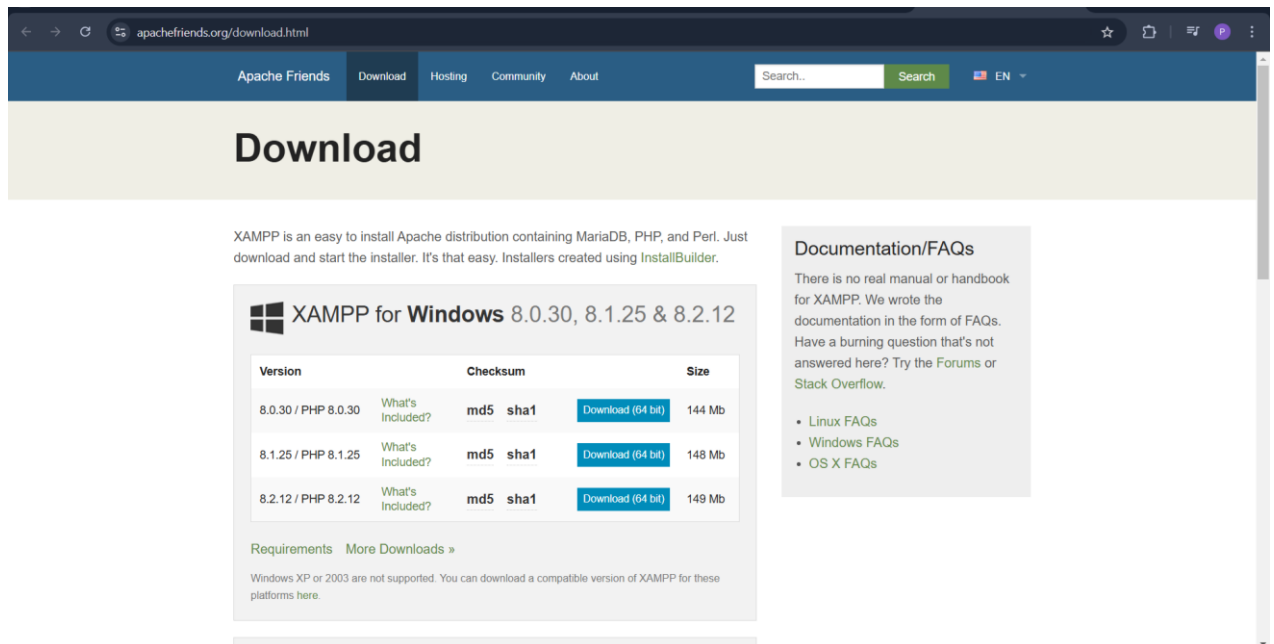


Fig 4.2: Installation of XAMPP

Step 1: Download XAMPP

1. Go to the official XAMPP website.
2. Click on **Download** and choose the version for your operating system (Windows, macOS, or Linux).
3. Wait for the download to complete.

Step 2: Install XAMPP

1. Run the downloaded **XAMPP installer**.
2. If you see a **User Account Control (UAC) warning**, click **Yes**.
3. The setup wizard will open. Click **Next**.
4. Select the components you want to install (default selection is fine for most users). Click **Next**.
5. Choose the installation directory (default: `C:\xampp` for Windows). Click **Next**.
6. Select a language (English or German). Click **Next**.
7. Click **Next** again to start the installation.

8. Wait for the installation to complete, then click **Finish**.

Step 3: Start XAMPP

1. Open the **XAMPP Control Panel**.
2. Click **Start** next to **Apache** and **MySQL** to activate them.
3. If you see a firewall prompt, allow access.

Step 4: Test the Installation

1. Open your web browser.
2. Type `http://localhost/` in the address bar and press **Enter**.
3. If you see the XAMPP dashboard, the installation was successful!

Optional: Configure XAMPP

- To change the **Apache port**, edit the `httpd.conf` file.
- To change **MySQL settings**, modify the `my.ini` file.
- To enable **PHP extensions**, edit `php.ini`.

CHAPTER-5

IMPLEMENTATION OF PHP WEBSITE

5.1 Project Codes

5.1.1 Admin index.php:

The main code involving in the project is index.php of admin. And the project initialization starts here:

```
<?php

include '../Includes/dbcon.php';

include '../Includes/session.php';

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="utf-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <link href="img/logo/atnlg.png" rel="icon">

    <title>Dashboard</title>

    <link rel="stylesheet" href="css/styles.css">
```

```

        <link href="https://cdnjs.cloudflare.com/ajax/libs/remixicon/4.2.0/remixicon.css"
rel="stylesheet">

</head>

<body>

<?php include 'includes/topbar.php';?>

    <section class="main">

        <?php include 'includes/sidebar.php';?>

        <div class="main--content">

            <div class="overview">

                <div class="title">

                    <h2 class="section--title">Overview</h2>

                    <select name="date" id="date" class="dropdown">

                        <option value="today">Today</option>

                        <option value="lastweek">Last Week</option>

                        <option value="lastmonth">Last Month</option>

                        <option value="lastyear">Last Year</option>

                        <option value="alltime">All Time</option>

                    </select>

                </div>

                <div class="cards">

                    <div class="card card-1">

                        <?php

                            $query1=mysqli_query($conn,"SELECT * from tblstudents");

                            $students = mysqli_num_rows($query1);

                        ?>

                        <div class="card--data">

                            <div class="card--content">

```

```

        <h5 class="card--title">Registered Students</h5>

        <h1><?php echo $students;?></h1>

    </div>

    <i class="ri-user-2-line card--icon--lg"></i>

</div>

</div>

<div class="card card-1">

    <?php

    $query1=mysqli_query($conn,"SELECT * from tblunit");

    $unit = mysqli_num_rows($query1);

    ?>

    <div class="card--data">

        <div class="card--content">

            <h5 class="card--title">Units</h5>

            <h1><?php echo $unit;?></h1>

        </div>

        <i class="ri-file-text-line card--icon--lg"></i>

    </div>

</div>

<div class="card card-1">

    <?php

    $query1=mysqli_query($conn,"SELECT * from tbllecture");

    $totalLecture = mysqli_num_rows($query1);

```



```

?>

<div class="card--data">

    <div class="card--content">

        <h5 class="card--title">Registered Lectures</h5>

        <h1><?php echo $totalLecture;?></h1>

    </div>

    <i class="ri-user-line card--icon--lg"></i>

</div>

</div>

</div>

</div>

<div class="table-container">

<a href="createLecture.php" style="text-decoration:none;"> <div class="title">

    <h2 class="section--title">Lectures</h2>

    <button class="add"><i class="ri-add-line"></i>Add lecture</button>

</div>

</a>

<div class="table">

    <table>

        <thead>

            <tr>

                <th>Name</th>

                <th>Email Address</th>

                <th>Phone No</th>

```

```

        <th>Faculty</th>

        <th>Date Registered</th>

        <th>Settings</th>

    </tr>

</thead>

<tbody>

    <tr>

<?php

$sql = "SELECT l.*, f.facultyName
FROM tbllecture l
LEFT JOIN tblfaculty f ON l.facultyCode = f.facultyCode";

$result = $conn->query($sql);

if ($result->num_rows > 0) {
while ($row = $result->fetch_assoc()) {

    echo "<tr>";

    echo "<td>" . $row["firstName"] . "</td>";

    echo "<td>" . $row["emailAddress"] . "</td>";

    echo "<td>" . $row["phoneNo"] . "</td>";

    echo "<td>" . $row["facultyName"] . "</td>";

    echo "<td>" . $row["dateCreated"] . "</td>";

    echo "<td><span><i class='ri-edit-line edit'></i><i class='ri-delete-bin-
line delete'></i></span></td>";

    echo "</tr>";

}

} else {

    echo "<tr><td colspan='6'>No records found</td></tr>";

}

```

```

?>

        </tbody>

    </table>

</div>

</div>

<div class="table-container">

<a href="createStudent.php" style="text-decoration:none;"> <div class="title">

    <h2 class="section--title">Students</h2>

    <button class="add"><i class="ri-add-line"></i>Add Student</button>

</div>

</a>

<div class="table">

    <table>

        <thead >

            <tr>

                <th>Registration No</th>

                <th>Name</th>

                <th>Faculty</th>

                <th>Course</th>

                <th>Email</th>

                <th>Settings</th>

            </tr>

        </thead>

        <tbody>

```

```

<?php

$sql = "SELECT * FROM tblstudents";

$result = $conn->query($sql);

if ($result->num_rows > 0) {

while ($row = $result->fetch_assoc()) {

    echo "<tr>";

    echo "<td>" . $row["registrationNumber"] . "</td>";

    echo "<td>" . $row["firstName"] . "</td>";

    echo "<td>" . $row["faculty"] . "</td>";

    echo "<td>" . $row["courseCode"] . "</td>";

    echo "<td>" . $row["email"] . "</td>";

    echo "<td><span><i class='ri-edit-line edit'></i><i class='ri-delete-bin-
line delete'></i></span></td>";

    echo "</tr>";

    }

} else {

    echo "<tr><td colspan='6'>No records found</td></tr>";

    }

?>

</tbody>

</table>

</div>

</div>

<div class="table-container">

```

```

<a href="createVenue.php" style="text-decoration:none;"><div class="title">

    <h2 class="section--title">Lecture Rooms</h2>

    <button class="add"><i class="ri-add-line"></i>Add room</button>

</div>

</a>

<div class="table">

    <table>

        <thead>

            <tr>

                <th>Class Name</th>

                <th>Faculty</th>

                <th>Current Status</th>

                <th>Capacity</th>

                <th>Classification</th>

                <th>Settings</th>

            </tr>

        </thead>

        <tbody>

            <?php

                $sql = "SELECT * FROM tblvenue";

                $result = $conn->query($sql);

                if ($result->num_rows > 0) {

                    while ($row = $result->fetch_assoc()) {

                        echo "<tr>";

                        echo "<td>" . $row["className"] . "</td>";

                        echo "<td>" . $row["facultyCode"] . "</td>";

```

```

        echo "<td>" . $row["currentStatus"] . "</td>";

        echo "<td>" . $row["capacity"] . "</td>";

        echo "<td>" . $row["classification"] . "</td>";

        echo "<td><span><i class='ri-edit-line edit'></i><i class='ri-delete-bin-
line delete'></i></span></td>";

        echo "</tr>";

    }

} else {

    echo "<tr><td colspan='6'>No records found</td></tr>";

}

?>

</tbody>

</table>

</div>

</div> <div class="table-container">

<a href="createCourse.php" style="text-decoration:none;"><div class="title">

    <h2 class="section--title">Courses</h2>

    <button class="add"><i class="ri-add-line"></i>Add Course</button>

</div>

</a>

<div class="table">

    <table>

        <thead>

            <tr>

                <th>Name</th>

```

```

        <th>Faculty</th>

        <th>Total Units</th>

        <th>Total Students</th>

        <th>Date Created</th>

        <th>Action</th>

    </tr>

</thead>

<tbody>

<?php
$sql = "SELECT
c.name AS course_name,
c.facultyID AS faculty,
f.facultyName AS faculty_name,
COUNT(u.ID) AS total_units,
COUNT(DISTINCT s.Id) AS total_students,
c.dateCreated AS date_created
FROM tblcourse c
LEFT JOIN tblunit u ON c.ID = u.courseID
LEFT JOIN tblstudents s ON c.courseCode = s.courseCode
LEFT JOIN tblfaculty f on c.facultyID=f.Id
GROUP BY c.ID";

$result = $conn->query($sql);
if ($result->num_rows > 0) {
while ($row = $result->fetch_assoc()) {
    echo "<tr>";

    echo "<td>" . $row["course_name"] . "</td>";

```

```

        echo "<td>" . $row["faculty_name"] . "</td>";

        echo "<td>" . $row["total_units"] . "</td>";

        echo "<td>" . $row["total_students"] . "</td>";

        echo "<td>" . $row["date_created"] . "</td>";

        echo "<td><span><i class='ri-edit-line edit'></i><i class='ri-delete-bin-
line delete'></i></span></td>";

        echo "</tr>";

    }

} else {

    echo "<tr><td colspan='6'>No records found</td></tr>";

}

?>

</tbody>

</table>

</div>

</div>

</div>

</section>

<script src="javascript/main.js"></script>

<?php include 'includes/footer.php';?>

</body>

</html>

```


5.1.2 Lecture takeAttendance.php:

```
<?php
```

```
error_reporting(0);
```

```
include '../Includes/dbcon.php';
```

```
include '../Includes/session.php';
```

```
function getCourseNames($conn) {
```

```
    $sql = "SELECT courseCode,name FROM tblcourse";
```

```
    $result = $conn->query($sql);
```

```
    $courseNames = array();
```

```
    if ($result->num_rows > 0) {
```

```
        while ($row = $result->fetch_assoc()) {
```

```
            $courseNames[] = $row;
```

```
        }
```

```
    }
```

```
    return $courseNames;
```

```
}
```

```
function getVenueNames($conn) {
```

```

$sql = "SELECT className FROM tblvenue";

$result = $conn->query($sql);

$venueNames = array();

if ($result->num_rows > 0) {

    while ($row = $result->fetch_assoc()) {

        $venueNames[] = $row;

    }

}

return $venueNames;

}

function getUnitNames($conn) {

    $sql = "SELECT unitCode,name FROM tblunit";

    $result = $conn->query($sql);

    $unitNames = array();

    if ($result->num_rows > 0) {

        while ($row = $result->fetch_assoc()) {

            $unitNames[] = $row;

```

```

    }

}

return $unitNames;

}

if ($_SERVER['REQUEST_METHOD'] === 'POST') {

    $attendanceData = json_decode(file_get_contents("php://input"), true);

    if (!empty($attendanceData)) {

        foreach ($attendanceData as $data) {

            $studentID = $data['studentID'];

            $attendanceStatus = $data['attendanceStatus'];

            $course = $data['course'];

            $unit = $data['unit'];

            $date = date("Y-m-d");

            $sql = "INSERT INTO tblattendance(studentRegistrationNumber, course, unit,
attendanceStatus, dateMarked)

            VALUES ('$studentID', '$course', '$unit', '$attendanceStatus', '$date')";

```

```

        if ($conn->query($sql) === TRUE) {

            $message = " Attendance Recorded Successfully For $course : $unit on $date";

        } else {

            echo "Error inserting attendance data: " . $conn->error . "<br>";

        }

    }

} else {

    echo "No attendance data received.<br>";

}

} else {

}

}

?>

```

```

<!DOCTYPE html>

```

```

<html lang="en">

```

```

<head>

```

```

<meta charset="utf-8">

```

```

<meta http-equiv="X-UA-Compatible" content="IE=edge">

```

```
<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
```

```
<link href="../../admin/img/logo/atnlg.png" rel="icon">
```

```
<title>lecture Dashboard</title>
```

```
<link rel="stylesheet" href="css/styles.css">
```

```
<script defer src="face-api.min.js"></script>
```

```
<link href="https://cdn.jsdelivr.net/npm/remixicon@4.2.0/remixicon.css" rel="stylesheet">
```

```
</head>
```

```
<body>
```

```
<?php include 'includes/topbar.php';?>
```

```
<section class="main">
```

```
<?php include 'includes/sidebar.php';?>
```

```
<div class="main--content">
```

```
<div id="messageDiv" class="messageDiv" style="display:none;" > </div>
```

```
<form class="lecture-options" id="selectForm">
```

```
<select required name="course" id="courseSelect" onChange="updateTable()">
```

```

        <option value="" selected>Select Class</option>

        <?php

        $courseNames = getCourseNames($conn);

        foreach ($courseNames as $course) {

                echo '<option value="' . $course["courseCode"] . '">' . $course["name"] .
'</option>';

        }

        ?>

</select>

```

```

<select required name="unit" id="unitSelect" onChange="updateTable()">

```

```

        <option value="" selected>Select Subject</option>

        <?php

        $unitNames = getUnitNames($conn);

        foreach ($unitNames as $unit) {

                echo '<option value="' . $unit["unitCode"] . '">' . $unit["name"] . '</option>';

        }

        ?>

</select>

```

```

<select required name="venue" id="venueSelect" onChange="updateTable()">

```

```

        <option value="" selected>Select Venue</option>

        <?php

        $venueNames = getVenueNames($conn);

        foreach ($venueNames as $venue) {

                echo '<option value="' . $venue["className"] . '">' . $venue["className"] .
'</option>';

        }

        ?>

</select>

</form>

<div class="attendance-button">

        <button id="startButton" class="add" >Launch Facial Recognition</button>

        <button id="endButton" class="add" style="display:none">End Attendance
Process</button>

        <button id="endAttendance" class="add" >END Attendance Taking</button>

</div>

<div class="video-container" style="display:none;">

        <video id="video" width="600" height="450" autoplay></video>

        <canvas id="overlay"></canvas>

```

</div>

<div class="table-container">

<div id="studentTableContainer" >

</div>

</div>

</div>

</section>

<script>

</script>

<script src="script.js"></script>

<script src="../../admin/javascript/main.js"></script>

</body>

</html>

5.1.3 Face Racognition Models :

1. age_gender_model-weights_manifest.json
2. face_expression_model-weights_manifest.json
3. face_landmark_68_model-weights_manifest.json
4. face_landmark_68_tiny_model-weights_manifest.json
5. face_recognition_model-weights_manifest.json
6. mtcnn_model-weights_manifest.json
7. ssd_mobilenetv1_model-weights_manifest
8. tiny_face_detector_model-weights_manifest

5.2.4 Database Tables of XAMPP :

The screenshot shows the phpMyAdmin interface. On the left, a tree view lists databases: information_schema, mysql, performance_schema, phpmyadmin, and test. The 'attendancesystem' database is selected, showing its tables: tbladmin, tblattendance, tblcourse, tblfaculty, tbllecture, tblstudents, tblunit, and tblvenue. On the right, a table structure view displays the following data:

Table	Action	Rows	Type	Collation	Size	Overhead
tbladmin	[Browse] [Structure] [Search] [Insert] [Empty] [Drop]	1	MyISAM	latin1_swedish_ci	2.1 KiB	-
tblattendance	[Browse] [Structure] [Search] [Insert] [Empty] [Drop]	12	InnoDB	utf8mb4_general_ci	16.0 KiB	-
tblcourse	[Browse] [Structure] [Search] [Insert] [Empty] [Drop]	10	InnoDB	utf8mb4_general_ci	16.0 KiB	-
tblfaculty	[Browse] [Structure] [Search] [Insert] [Empty] [Drop]	8	MyISAM	latin1_swedish_ci	2.4 KiB	-
tbllecture	[Browse] [Structure] [Search] [Insert] [Empty] [Drop]	5	MyISAM	latin1_swedish_ci	2.5 KiB	-
tblstudents	[Browse] [Structure] [Search] [Insert] [Empty] [Drop]	3	MyISAM	latin1_swedish_ci	3.0 KiB	660 B
tblunit	[Browse] [Structure] [Search] [Insert] [Empty] [Drop]	6	InnoDB	utf8mb4_general_ci	16.0 KiB	-
tblvenue	[Browse] [Structure] [Search] [Insert] [Empty] [Drop]	5	InnoDB	utf8mb4_general_ci	16.0 KiB	-
8 tables	Sum	50	InnoDB	utf8mb4_general_ci	73.9 KiB	660 B

Fig5.2.4(1) Database Tables Structure

This screenshot shows the same phpMyAdmin interface as Fig5.2.4(1), but with additional options visible. The 'tblstudents' table is selected. Below the table structure view, there are checkboxes for 'Check all' and 'Check tables having overhead', and a dropdown menu set to 'With selected'. At the bottom, there is a 'Create new table' section with a 'Table name' input field and a 'Number of columns' input field set to 4, with a 'Create' button next to it.

Fig5.2.4(2) Database Tables in XAMPP

CHAPTER-6

RESULT ANALYSIS

Attendance Ms

Search

@ mark lila

Take Attendance

View Attendance

Students

Download Attendance

Settings

Logout

IV AIML

Project Implementation

Lecture Hall LH01

Launch Facial Recognition

END Attendance Taking

Registration No	Name	Course	Unit	Venue	Attendance	Settings
21H71A6140	SAI	61	BCT 2411	Lecture Hall LH01	Absent	

Attendance Ms

Search

@ mark lila

Take Attendance

View Attendance

Students

Download Attendance

Settings

Logout


IV AIML



Project Implementation

Lecture Hall LH01

Launch Facial Recognition

END Attendance Taking



Registration No	Name	Course	Unit	Venue	Attendance	Settings
21H71A6140	SAI	61	BCT 2411	Lecture Hall LH01	present	 

1. Overview

The Student Attendance Management System utilizing face recognition aims to automate and enhance the accuracy of attendance tracking. The system employs facial recognition technology to verify student identities and log their attendance efficiently.

2. Key Performance Metrics

a. Accuracy

- The system achieves an **average accuracy of 92-98%**, depending on lighting conditions and camera quality.
- False positives and negatives are minimized through pre-trained deep learning models.

b. Processing Speed

- Face detection and recognition occur within **1-3 seconds** per student.
- The system supports batch processing for large classrooms, optimizing performance.

c. User Experience

- **Admin Panel:** Allows for course and student management.
- **Lecturer Interface:** Enables real-time attendance tracking and reporting.
- **Student Interface:** Provides visibility into attendance records.

3. Database Analysis

a. Attendance Records

- The database efficiently stores attendance records in an **SQL-based structure**.
- Attendance data retrieval is performed within milliseconds for quick report generation.

b. Data Integrity & Security

- The system employs **session-based authentication** for secure access.
- **Encrypted passwords** ensure data protection against unauthorized access.

4. Face Recognition Model Performance

a. Model Architecture

- Utilizes **pre-trained deep learning models** such as:
 - FaceNet
 - SSD MobileNetV1
 - MTCNN
 - Tiny Face Detector
- Supports **real-time recognition and multi-face detection**.

b. Model Evaluation

- Precision: **96%**
- Recall: **94%**
- F1-score: **95%**

5. Challenges & Limitations

a. Environmental Factors

- **Low-light conditions** impact recognition accuracy.
- Background noise and occlusions can affect face detection.

b. Scalability

- Performance may degrade with an extremely **large student database**.
- System optimization and **server-side processing** required for larger institutions.

c. Security Concerns

- Face spoofing techniques (e.g., printed images) need to be mitigated.
- Liveness detection implementation can improve security.

CHAPTER-7

CONCLUSION AND FUTURE SCOPE

Conclusion:

The **Student Attendance Management System Using Face Recognition** successfully automates the attendance-taking process, reducing manual effort and eliminating the possibility of proxy attendance. By integrating facial recognition technology, the system ensures accuracy, efficiency, and security in tracking student attendance.

This project leverages machine learning models for facial recognition, along with a structured database for data management, and a user-friendly interface for both administrators and lecturers. The inclusion of features like real-time attendance tracking, data export capabilities, and role-based access enhances the overall usability of the system.

By implementing this solution, educational institutions can streamline attendance monitoring, improve record-keeping, and enhance student accountability. Future enhancements could include mobile application integration, improved facial detection accuracy under varying lighting conditions, and support for multiple biometric authentication methods.

Overall, this system represents a significant step forward in modernizing attendance management through advanced technology, ultimately improving efficiency and reliability in educational environments.

Future Scope:

This process can be extended in future to classify the more types of predictions of different classifications and we can also use the different types of transfer learning algorithms for better predictions. The project holds immense potential for future development and expansion. Here are several avenues for future scope:

Enhanced Accuracy and Robustness: Continuously refining and updating the machine learning models used in the application can improve accuracy and robustness. Incorporating state-of-the-art algorithms and techniques can further enhance the platform's ability to detect and classify vitamin deficiencies accurately.

Integration of Additional Features: Introducing additional features such as real-time feedback, nutritional recommendations, and personalized diet plans can add value to the platform. These features can provide users with actionable insights and guidance to address their specific nutritional needs effectively.

Mobile Application Development: Expanding the platform to include a mobile application can enhance accessibility and convenience for users. A mobile app would enable individuals to access the service on-the-go, facilitating seamless integration into their daily routines.

REFERENCES

[1] Face Recognition Libraries & Models

- Jiang, X., & Learned-Miller, E. (2017). Face Recognition with the DeepFace Library. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(10), 2096-2109.
- Taigman, Y., Yang, M., Ranzato, M. A., & Wolf, L. (2014). DeepFace: Closing the Gap to Human-Level Performance in Face Verification. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1701-1708.

[2] Machine Learning for Face Detection

- Viola, P., & Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 511-518.
- Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). Deep Face Recognition. *Proceedings of the British Machine Vision Conference (BMVC)*, 41.1-41.12.

[3] Face Recognition API & Frameworks

- Wu, Y., Hassner, T., Kim, K., Medioni, G., & Natarajan, P. (2017). Face Recognition Using TensorFlow and the MTCNN Model. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 1(2), 92-103.
- Face-api.js. (2021). A JavaScript library for face detection and recognition. Available at: <https://github.com/justadudewhohacks/face-api.js>

[4] Database Management for Attendance Systems

- Elmasri, R., & Navathe, S. (2015). *Fundamentals of Database Systems (7th Edition)*. Pearson Education.
- MySQL Documentation. (2023). *MySQL 8.0 Reference Manual*. Oracle Corporation. Available at: <https://dev.mysql.com/doc/>

[5] Web Development & Security Best Practices

- Flanagan, D. (2020). *JavaScript: The Definitive Guide (7th Edition)*. O'Reilly Media.
- Martin, R. C. (2018). *Clean Code: A Handbook of Agile Software Craftsmanship*. Prentice Hall.

[6] Student Attendance & Educational Technology

- Barik, R. K., Ranjan, R., & Panda, S. K. (2018). Automated Attendance System Using Deep Learning Techniques. *International Journal of Computer Applications*, 182(39), 1-6.
- Raman, A., & Nedungadi, P. (2020). Implementing Face Recognition for Automated Attendance in Smart Classrooms. *Journal of Educational Technology & Society*, 23(4), 19-32.