**B.M.S. COLLEGE OF ENGINEERING BENGALURU**

**Autonomous Institute, Affiliated to VTU**

OOMD Mini Project Report

# Smart Attendance System using Face Recognition

*Submitted in partial fulfillment for the award of degree of*

Bachelor of Engineering
in
Computer Science and Engineering

*Submitted by:*

**Bhoomi Suresh Kota - 1BM23CS065**

**Chaithanya Sudhan - 1BM23CS073**

**Gopika Pushparajan - 1BM23CS101**

**Parth Jain - 1BM23CS357**

Department of Computer Science and Engineering
B.M.S. College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560 019
2025-26

**B.M.S. COLLEGE OF ENGINEERING**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

## *DECLARATION*

We, Bhoomi Suresh Kota (1BM23CS065), Chaithanya Sudhan (1BM23CS073), Gopika Pushparajan (1BM23CS10), Parth Jain (1BM23CS357) students of 5th Semester, B.E, Department of Computer Science and Engineering, BMS College of Engineering, Bangalore, hereby declare that, this OOMD Mini Project entitled "Smart Attendance System using Face Recognition" has been carried out in Department of CSE, B.M.S. College of Engineering, Bangalore during the academic semester August 2025- December 2025. I also declare that to the best of our knowledge and belief, the OOMD mini Project report is not from part of any other report by any other students.

**Signature of the Candidate**

Bhoomi Suresh Kota - 1BM23CS065

Chaithanya Sudhan - 1BM23CS073

Gopika Pushparajan - 1BM23CS101

Parth Jain - 1BM23CS357

# B.M.S. COLLEGE OF ENGINEERING

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING


## *CERTIFICATE*

This is to certify that the OOMD Mini Project titled "Smart Attendance System using Face Recognition" has been carried out by **Bhoomi Suresh Kota (1BM23CS065), Chaithanya Sudhan (1BM23CS073), Gopika Pushparajan (1BM23CS10), Parth Jain (1BM23CS357)** during the academic year 2025-2026.

**Signature of the Faculty in Charge**

## Table of Contents

# Chapter 1: Problem Statement

## Smart Attendance System Using Face Recognition

Traditional attendance methods in schools and workplaces are slow, error-prone, and vulnerable to proxy attendance. Manual roll calls waste valuable time, while biometric and ID-card systems require physical contact and can be easily misused.

There is a need for a contactless, real-time, and reliable attendance system that ensures authenticity and reduces administrative effort.

The proposed solution is an AI-powered Smart Attendance System that uses facial recognition to automatically detect, identify, and mark attendance as individuals enter a classroom or workspace. The system captures live video, performs face verification, updates attendance records instantly, and generates digital reports.

The solution must:

- Detect faces in real time using a camera
- Identify registered individuals with high accuracy
- Automatically mark attendance as Present/Absent/Late
- Prevent proxy attendance
- Provide instructors/admins with digital attendance reports

This system will save time, increase accuracy, and enhance transparency, offering a modern attendance alternative suitable for institutions and organizations.

# Chapter 2: Software Requirement Specification

## 1. Introduction

### 1.1 Purpose of this Document:

The purpose of this document is to outline the requirements and specifications for the development of a Smart Attendance System using Face Recognition. It provides a clear understanding of the system's objectives, scope, and deliverables to ensure smooth development and implementation.

### 1.2 Scope of this Document:

This document defines the overall working and primary objectives of the Smart Attendance System. It includes a description of the expected development cost, time, system functionalities, and integration requirements.

### 1.3 Overview:

The Smart Attendance System is a software solution designed to automate the attendance-taking process in classrooms, laboratories, and organizations. The system uses real-time facial recognition to detect and identify individuals, thereby eliminating proxy attendance and reducing manual effort.

## 2. General Description:

The Smart Attendance System will cater to the needs of educational institutions and workplaces, providing features such as face-based attendance, real-time recognition, session management, and attendance reporting. The system will be accessible to administrators, teachers, and authorized staff with varying levels of technical expertise.

## 3. Functional Requirements

### 3.1 Session Management:

- Allow instructors to create class sessions with subject, date, and time.
- Enable instructors to start and stop attendance sessions.
- Maintain session-level attendance summaries.

### 3.2 Face Detection & Recognition:

- Capture live frames through a camera device.
- Detect multiple faces in real time.
- Identify students using pre-stored face data.
- Ensure recognition accuracy under different lighting and environmental conditions.

### 3.3 Attendance Management:

- Automatically mark students as Present/Absent/Late based on recognized faces.
- Prevent proxy attendance through secure biometric verification.
- Allow instructors to review and manually edit attendance if needed.

### 3.4 Student Management:

- Maintain student profiles including ID, name, and facial encodings.
- Support adding, updating, and deleting student records.
- Store face data securely for recognition purposes.

### 3.5 Reporting:

- Generate daily, weekly, and monthly attendance reports.
- Export reports in PDF or CSV formats.
- Provide analytics such as attendance percentages and trends.

## 4. Interface Requirements

### 4.1 User Interface:

- Provide a clean, intuitive, and user-friendly interface for instructors and admins.
- Accessible via web browsers, mobile devices, and desktop systems.
- Show real-time recognition results on screen.

### 4.2 Integration Interfaces:

- Integrate with camera devices for real-time video capture.
- Support integration with institutional management systems (if required).
- Secure APIs for student enrollment and data retrieval.

## 5. Performance Requirements

### 5.1 Response Time:

- The system should detect and recognize faces within 2 seconds of capture.
- Attendance updates should reflect instantly during live sessions.

### 5.2 Scalability:

- Handle a minimum of 500 concurrent recognitions per session.
- Support ongoing sessions in multiple classrooms.

### 5.3 Data Integrity:

- Ensure consistent and accurate attendance updates across all modules.
- Prevent duplicate or conflicting records during high activity.

## 6. Design Constraints

### 6.1 Hardware Limitations:

- The system should work with standard classroom hardware (webcams, IP cameras, computers).
- Image processing must perform efficiently on mid-range devices.

### 6.2 Software Dependencies:

- Utilize a high-performance database management system (e.g., MySQL, PostgreSQL).
- Use programming languages/frameworks suitable for AI and web services (e.g., Python, Flask/Django, OpenCV, TensorFlow).

## 7. Non-Functional Attributes

### 7.1 Security:

- Implement strong authentication and authorization for admins, teachers, and system staff.
- Encrypt stored facial data to protect biometric privacy.

### 7.2 Reliability:

- Ensure the system remains operational during active class sessions with minimal downtime.
- Provide fallback options for manual attendance if needed.

### 7.3 Scalability:

Support future expansion to multiple departments, campuses, or institutions.

### 7.4 Portability:

The system should run across different devices and platforms (Windows, macOS, Android browsers).

### 7.5 Usability:

Simple navigation and clear UI layout for instructors to manage sessions quickly.

### 7.6 Reusability:

Adopt modular design patterns to allow future updates such as RFID integration or mobile apps.

### 7.7 Compatibility:

Compatible with common web browsers like Chrome, Firefox, and Safari.

### 7.8 Data Integrity:

Ensure accurate and consistent storage of attendance logs, timestamps, and session metadata.

## 8. Preliminary Schedule and Budget:

The development of the Smart Attendance System is estimated to take 4–6 months, with a projected budget covering project planning, development, testing, deployment, and hardware procurement (cameras). Costs will depend on scale and institution requirements.

# Chapter 3: Class Modeling

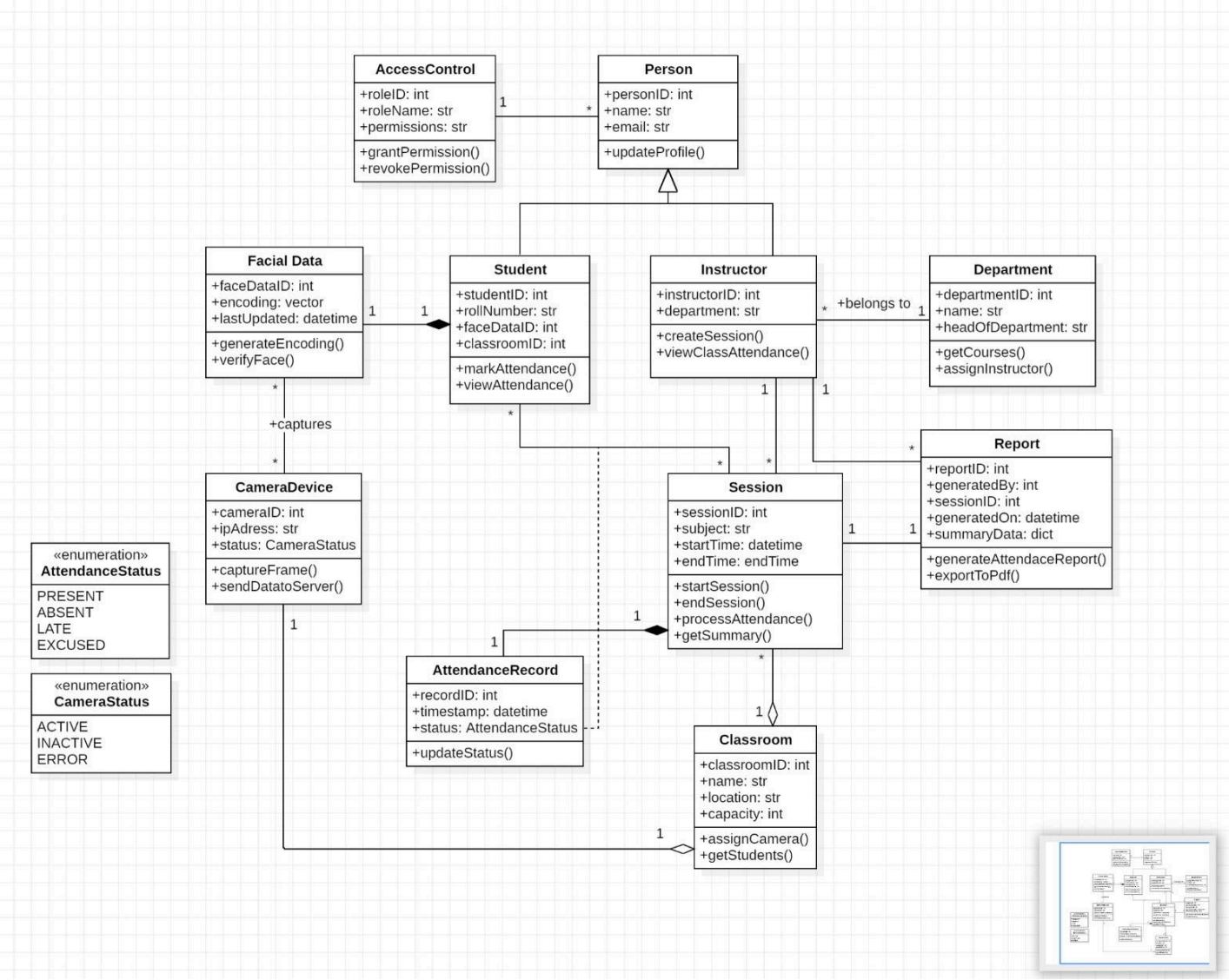


**Fig. 3.1 Class Diagram**

This class diagram represents the structure and interactions of a Smart Attendance System that uses real-time facial recognition to automate student attendance. The model includes all major roles, devices, data objects, and system functionalities required for accurate and secure attendance tracking.

**1. Person (Superclass)**

- **Attributes:** personID, name, email
- **Ops:** updateProfile()
- **Relevance:** Base class for all users; avoids repeating common attributes. Students and instructors inherit from it.

**2. Student (extends Person)**

- **Attributes:** studentID, rollNumber, faceDataID, classroomID
- **Ops:** markAttendance(), viewAttendance()
- **Relevance:** Main entity for attendance. Linked to a classroom and facial data.

**3. Instructor (extends Person)**

- **Attributes:** instructorID, department
- **Ops:** createSession(), viewClassAttendance()
- **Relevance:** Manages attendance sessions and monitors class statistics.

**4. Department**

- **Attributes:** departmentID, name, headOfDepartment
- **Ops:** getCourses(), assignInstructor()
- **Relevance:** Organizes instructors and controls academic structure.

**5. AccessControl**

- **Attributes:** roleID, roleName, permissions
- **Ops:** grantPermission(), revokePermission()
- **Relevance:** Defines what each role (Admin/Instructor/Student) can do in the system.

**6. FacialData**

- **Attributes:** faceDataID, encoding, lastUpdated
- **Ops:** generateEncoding(), verifyFace()
- **Relevance:** Stores and verifies biometric face encodings for recognition.

**7. CameraDevice**

- **Attributes:** cameraID, ipAddress, status
- **Ops:** captureFrame(), sendDataToServer()
- **Relevance:** The IoT camera that continuously captures frames for real-time attendance.

**8. Session**

- **Attributes:** sessionID, subject, startTime, endTime
- **Ops:** startSession(), endSession(), processAttendance(), getSummary()
- **Relevance:** Represents a single lecture. Links camera input, students, and attendance records.

**9. AttendanceRecord**

- **Attributes:** recordID, timestamp, status
- **Ops:** updateStatus()
- **Relevance:** Stores attendance result for each student in a session.

**10. Classroom**

- **Attributes:** classroomID, name, location, capacity
- **Ops:** assignCamera(), getStudents()
- **Relevance:** Physical room linked to camera and enrolled students.

**11. Report**

- **Attributes:** reportID, generatedBy, sessionID, generatedOn, summaryData
- **Ops:** generateAttendanceReport(), exportToPDF()
- **Relevance:** Creates daily/weekly/monthly attendance summaries for departments and faculty.

**12. Enumerations**

- **AttendanceStatus:** PRESENT, ABSENT, LATE, EXCUSED
- **CameraStatus:** ACTIVE, INACTIVE, ERROR

  Provide standard labels for attendance and camera states.

# Chapter 4: State Modeling

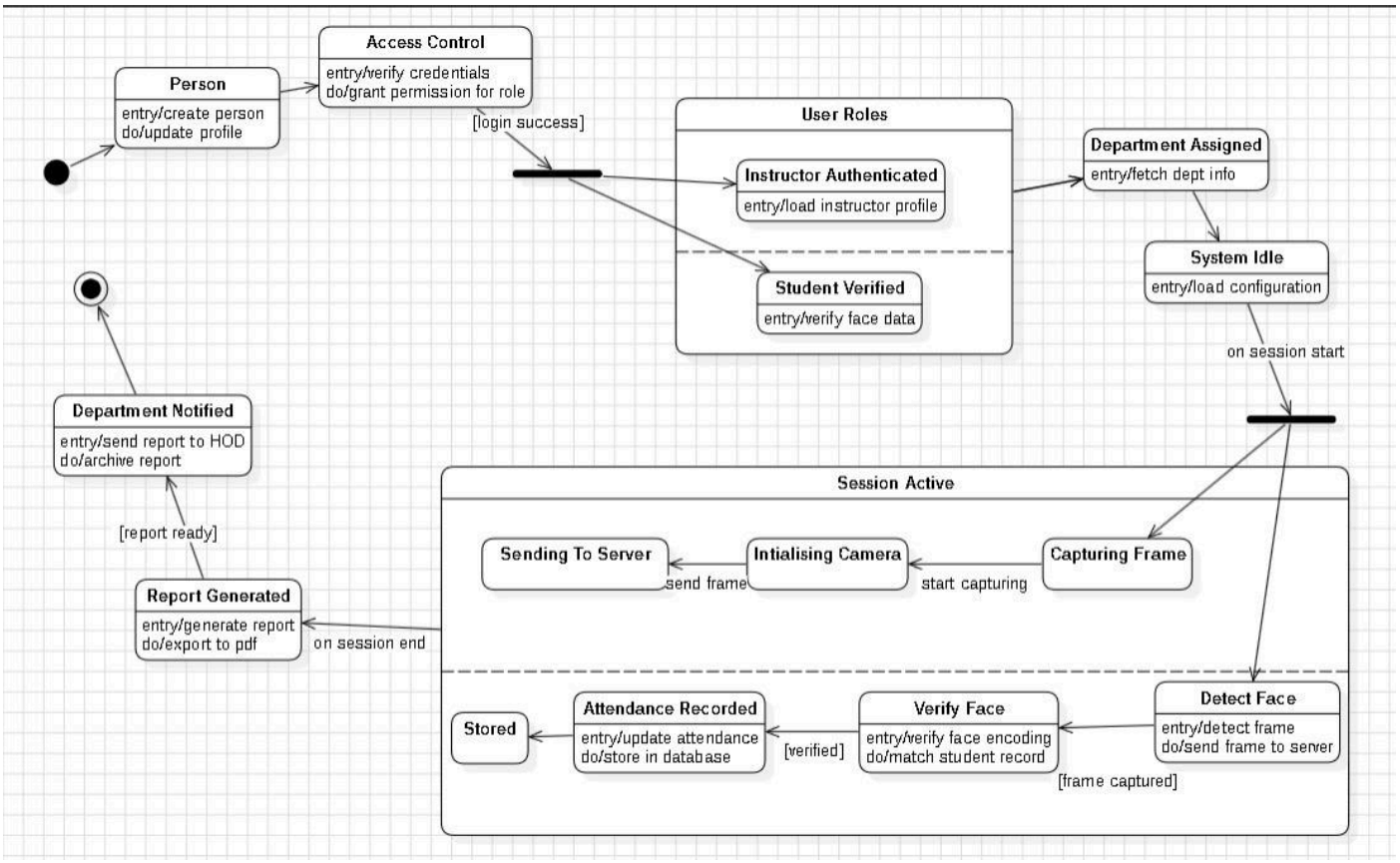


**Fig. 4.1 State Diagram**

This advanced state diagram shows the complete workflow of the Smart Attendance System using Face Recognition - from user login to report delivery. The process begins with Access Control, where user credentials are verified. After login, the system identifies whether the user is an Instructor or Student, loads the appropriate profile, and retrieves the related Department information. It then enters the System Idle state, waiting for the instructor to start a session.

When a session begins, the system transitions into the Session Active composite state, which includes several nested substates: Initialising Camera, Capturing Frame, Sending to Server, Detect Face, Verify Face, and Attendance Recorded. These states handle camera setup, frame capture, face detection, identity matching, and automatic attendance marking.

Once verification is successful, the student's attendance is stored. At the end of the session, the system generates the attendance report, exports it as a PDF, and enters the Department Notified state, where the report is sent to the HOD.

**1. Person (Initial State)**

**Entry Actions:** create person profile, update basic information

**Relevance:** This is the starting point where any system user (student or instructor) first exists. Before authentication, the system must have the user's profile.

**2. Access Control**

**Entry/Do Actions:** verify credentials, grant permission based on role

**Event:** login success

**Relevance:** Ensures security, prevents unauthorized access, and determines whether the user is an instructor or a student. It is a mandatory step before performing any system actions.

**3. User Roles (Composite State)**

This state contains two substates depending on the authenticated role.

**3.1 Instructor Authenticated**

**Entry Actions:** load instructor profile

**Relevance:** Loads instructor-specific data such as assigned classroom, subjects, and active sessions. The instructor is responsible for starting and managing attendance sessions.

**3.2 Student Verified**

**Entry Actions:** verify facial data

**Relevance:** Used when a student logs in to view their attendance. Facial verification ensures that only the correct student can access their attendance records.

**4. Department Assigned**

**Entry Actions:** fetch department info of user

**Relevance:** Ensures instructors are linked to the correct department. This enables proper routing of attendance reports to the Head of Department (HOD).

**5. System Idle**

**Entry Actions:** load system configuration (cameras, class info, thresholds, etc.)

**Event:** on session start

**Relevance:** Represents the waiting state before an attendance session begins. Once the instructor starts the session, the system transitions into the active workflow.

**6. Session Active (Main Composite State)**

The core state where attendance is captured, processed, and stored. Includes several nested substates:

**6.1 Initialising Camera**

**Entry Actions:** start capturing, initialize connection with assigned camera

**Relevance:** Ensures the camera is ready and transmitting live video. Without this, face detection cannot begin.

**6.2 Capturing Frame**

**Entry Actions:** continuously capture frames from the camera

**Relevance:** Represents continuous monitoring of the classroom. Each frame is passed forward for detection and analysis.

**6.3 Sending To Server**

**Entry Actions:** send frame to server for processing

**Relevance:** Frames are typically processed on a backend server (CPU/GPU). This step transfers captured data for recognition.

**7. Detect Face**

**Entry Actions:** detect faces in the input frame

**Event:** frame captured

**Relevance:** Identifies whether a face is present in the frame. This is the first major AI step in the recognition pipeline.

**8. Verify Face**

**Entry/Do Actions:** verify face encoding, match encoding with student database

**Event:** verified

**Relevance:** Confirms identity by comparing extracted face encodings with stored student records. This step prevents proxy attendance.

**9. Attendance Recorded**

**Entry/Do Actions:** update student attendance, store record in database

**Relevance:** Final stage of the recognition loop. When a face is verified, an AttendanceRecord is created or updated for the session.

Stored (Pseudo-State) Represents data persistence.

Relevance: Indicates that attendance has been securely saved.

**10. Report Generated**

**Entry/Do Actions:** generate attendance report, export report as PDF

**Event:** on session end

**Relevance:** After the session ends, the system compiles all attendance data into a structured report, ready for official use.

**11. Department Notified (Final State)**

**Entry Actions:** send report to HOD, archive report

**Event:** report ready

**Relevance:** Final step where the generated report is forwarded to the department head for review, documentation, or academic processing.

# Chapter 5: Interaction Modeling

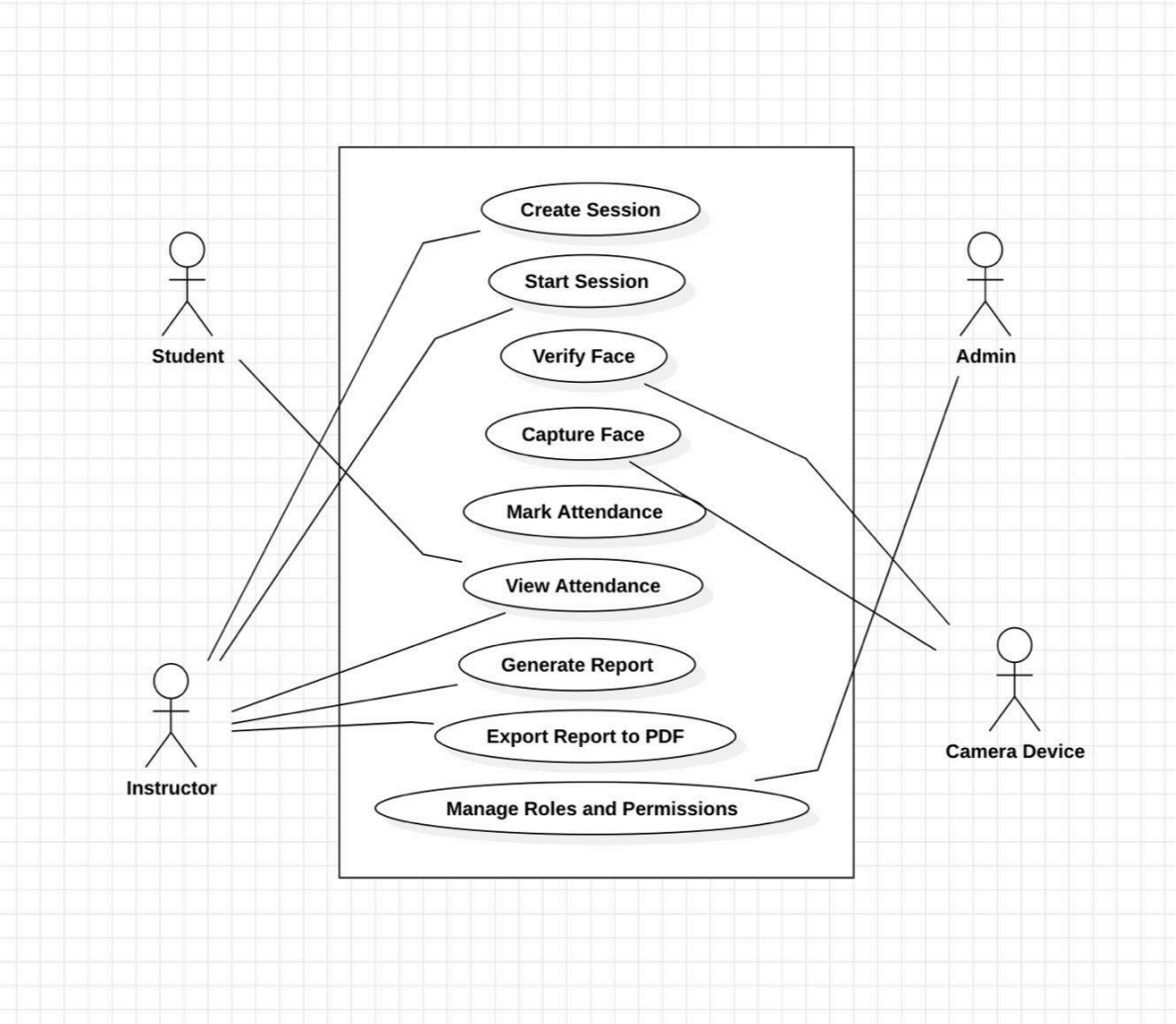


**Fig. 5.1 Use Case diagram**

The use case diagram shows all major interactions between the user roles and the Smart Attendance System. It visualizes what functionality the system must support for each actor.

**Actors & Their Relevance**

- **Instructor** – Starts sessions, captures attendance, generates reports. Central operator of the system.
- **Student** – Gets verified through face recognition and can view their attendance.
- **Camera Device** – Automatically captures frames and sends them for processing. Acts as an IoT sensor.
- **Admin** – Manages roles, permissions, and configuration settings.

**Main Use Cases & Relevance**

- **Create Session / Start Session** – Instructor begins a class attendance cycle.
- **Capture Face** – Camera detects students entering class; essential for automation.
- **Verify Face** – Face encodings compared with stored records to prevent proxy attendance
- **Mark Attendance** – System stores presence/absence based on recognition results.
- **View Attendance** – Students and instructors view attendance status.
- **Generate Report** – End-of-session report summarizing attendance.
- **Export to PDF** – Creates an official copy for academic documentation.
- **Manage Roles & Permissions** – Ensures secure, controlled system access.

This diagram clearly shows **who interacts with what**, emphasizing role-based access and automation.

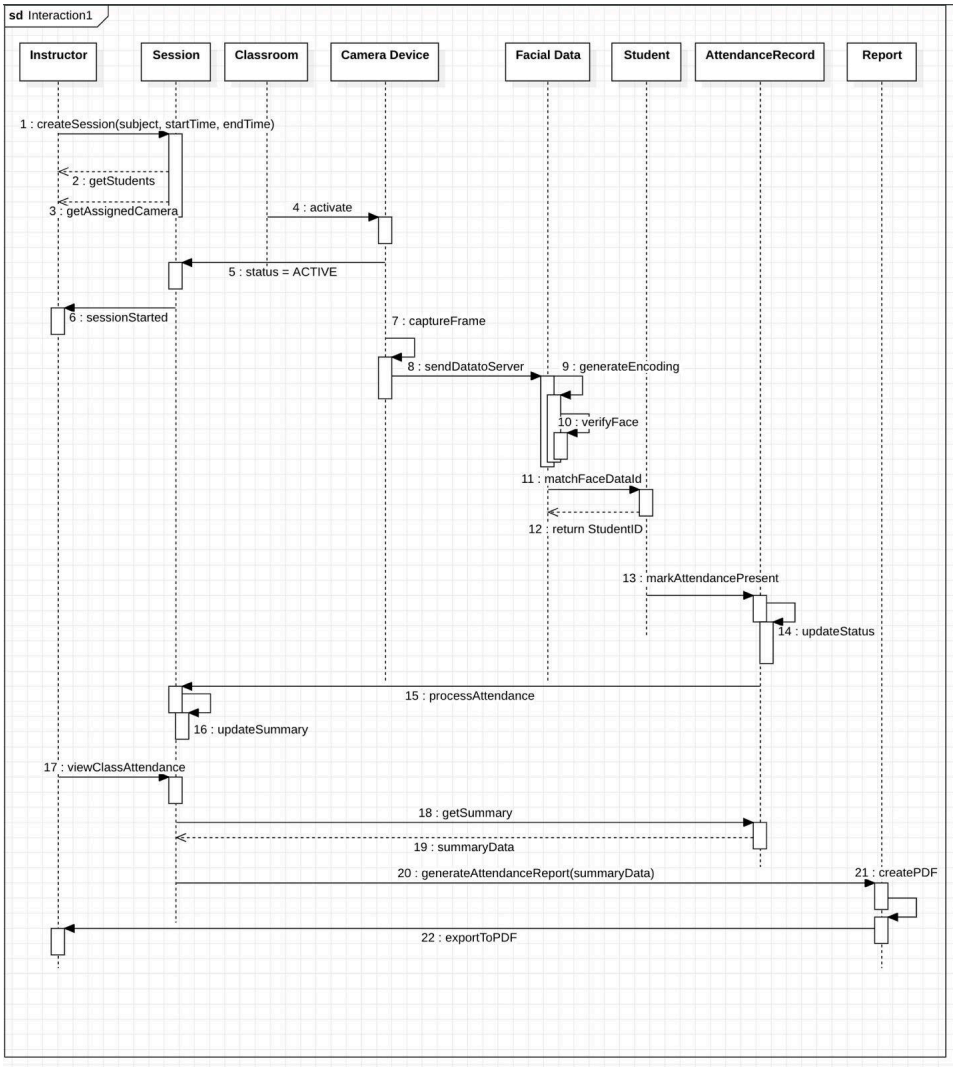


**Fig. 5.2 Sequence Diagram**

This sequence diagram details the step-by-step flow during a live attendance session. It shows objects, messages, and the exact order in which they occur.

**Key Lifelines & Relevance**

- Instructor – Initiates session.
- Session – Controls the workflow of attendance.
- Classroom – Provides student list and camera assignment.
- Camera Device – Continuously captures video.
- Facial Data – Encodes and verifies faces.
- Student – Identified using face match.
- AttendanceRecord – Stores the presence/absence.
- Report – Generates PDF summary.

**Sequence Flow Summary**

1. Instructor creates a session (createSession).
2. The classroom provides student data and a camera.
3. Camera activates and begins capturing frames.
4. For each frame:
    - System detects face
    - Generates encoding
    - Matches encoding with Student
    - Marks attendance
5. After session:
    - Summary generated
    - Report produced and exported

**Relevance**

This diagram represents the real-time automation pipeline and shows how multiple components collaborate to eliminate manual attendance.

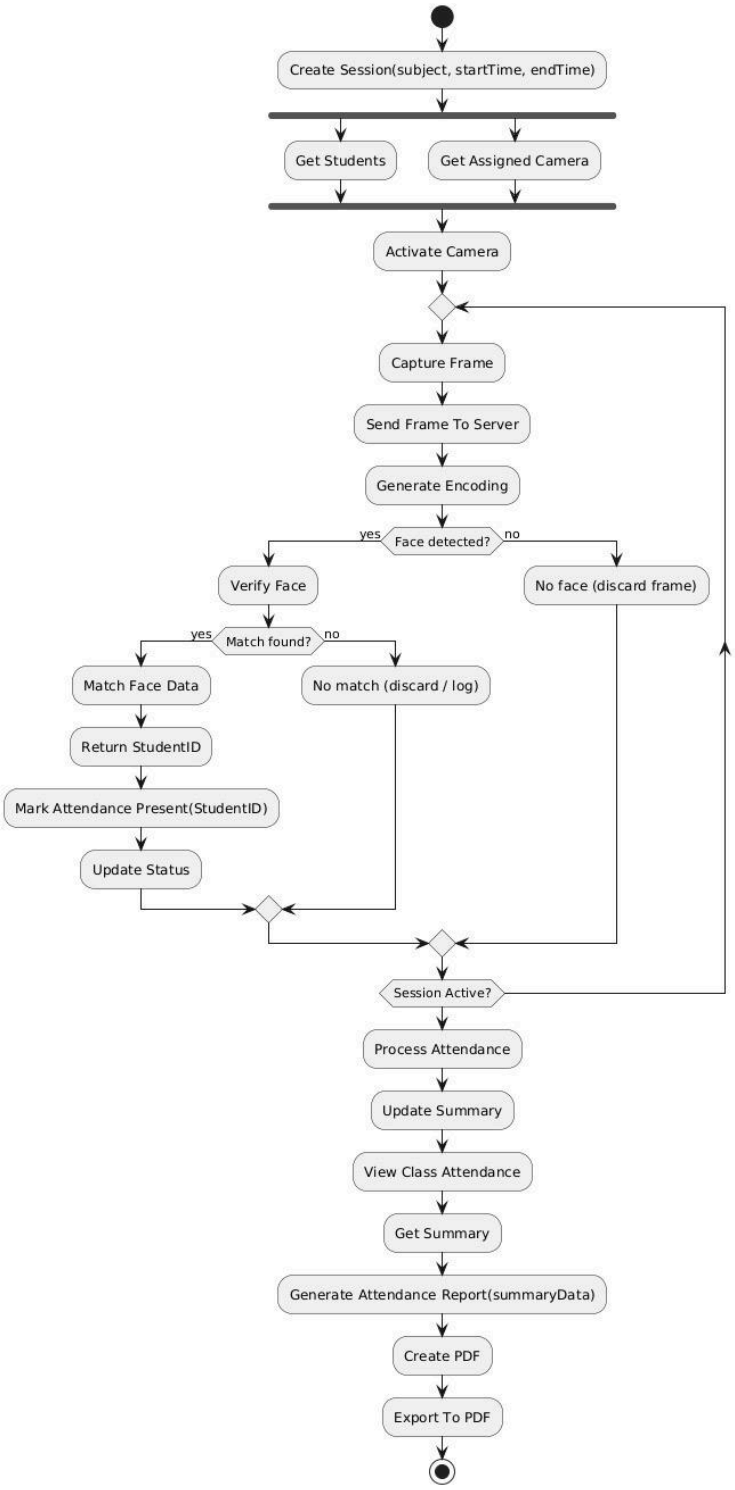


**Fig. 5.3 Activity diagram**

This diagram explains the workflow logic for attendance capture in a simple, decision-based flow.

**Activities & Relevance**

- Create Session – Initializes class details.
- Get Students / Get Assigned Camera – Prepares necessary data.
- Activate Camera – Starts the video feed.
- Capture Frame → Send to Server – Core sensing workflow.
- Generate Encoding → Verify Face – AI/ML processing.
- Match Found?
    - Yes → Mark Attendance
    - No → Discard Frame
- Process Attendance → Update Summary – Session-level processing.
- Generate Report → Export to PDF – Administrative conclusion.

**Relevance**

Activity diagrams highlight logical decision paths such as:

- Face detected?
- Match found?
- Session active?

This shows the system's operational intelligence and exception handling (e.g., no face, mismatch).

# Chapter 6: UI Design with Screenshots

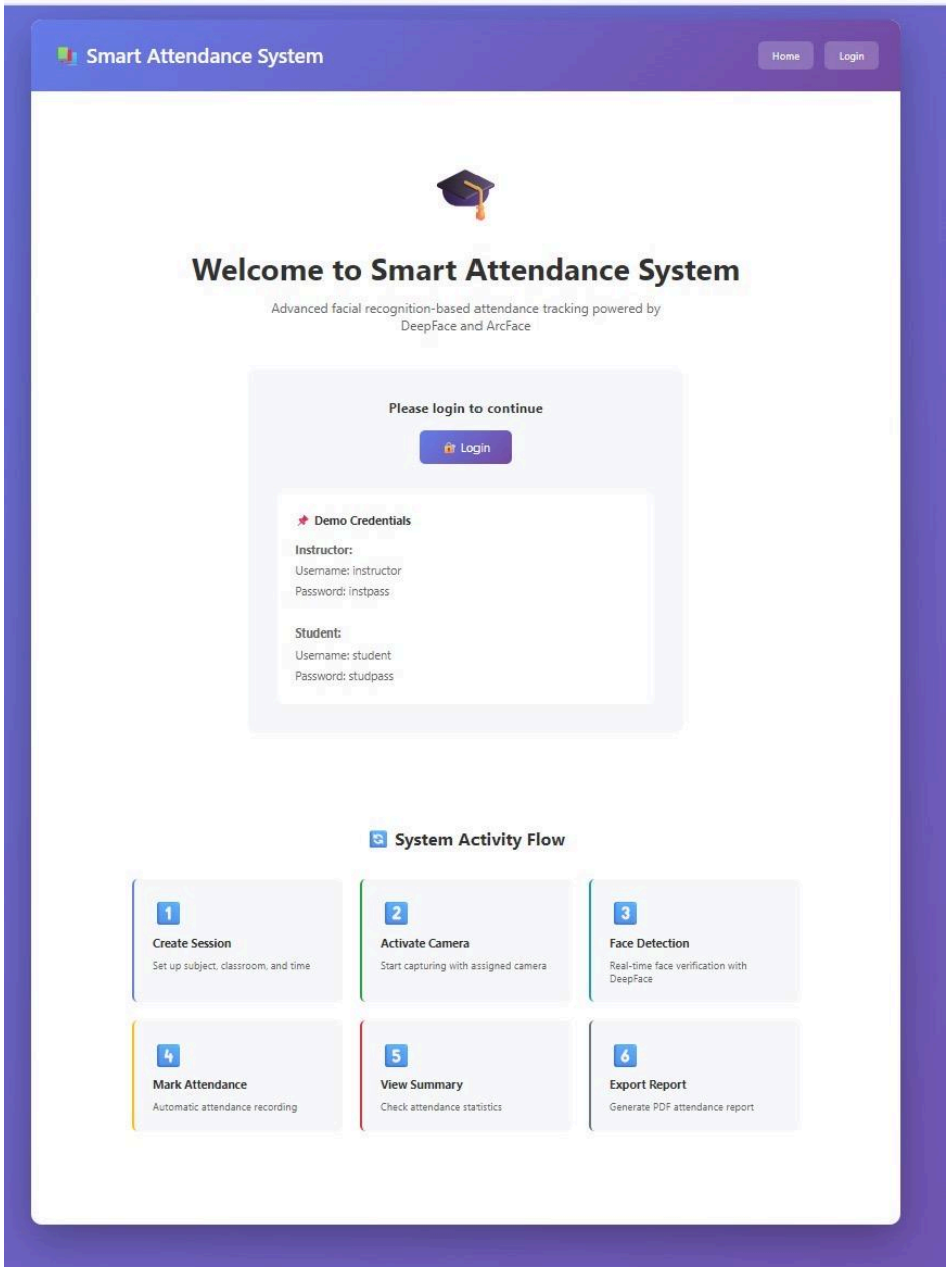


**Fig 6.1 Landing Page**

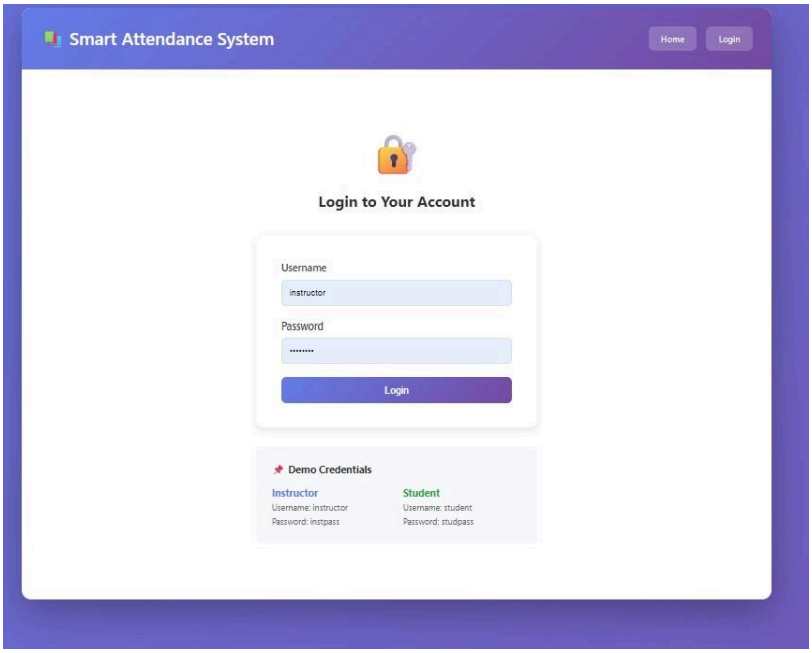


**Fig 6.2: Login Page**

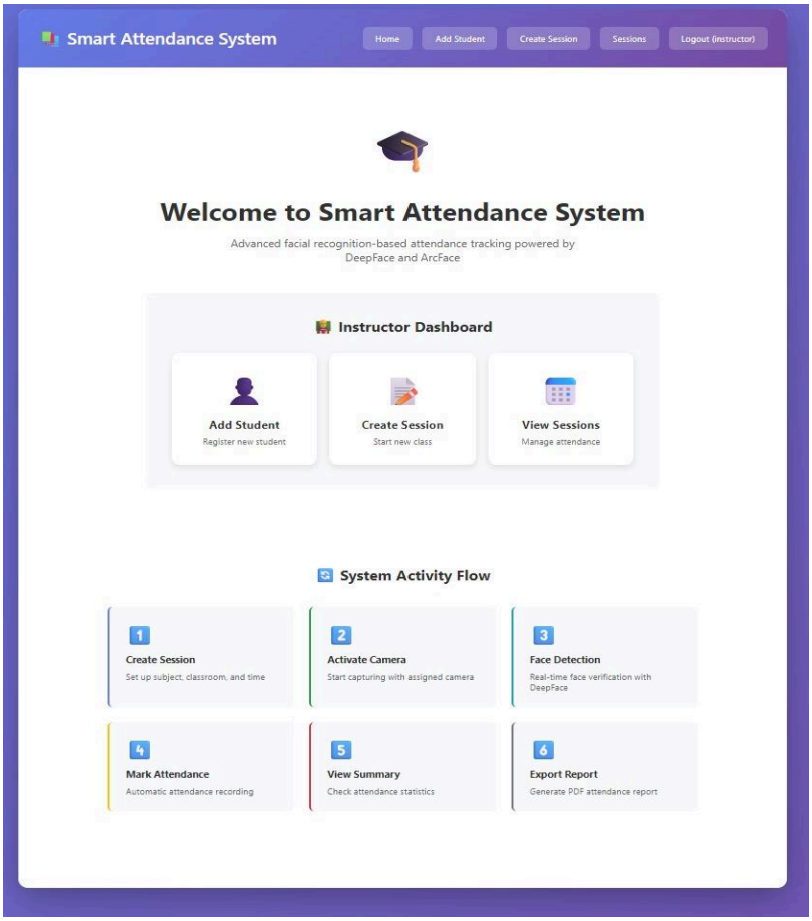


**Fig 6.3 Dashboard**

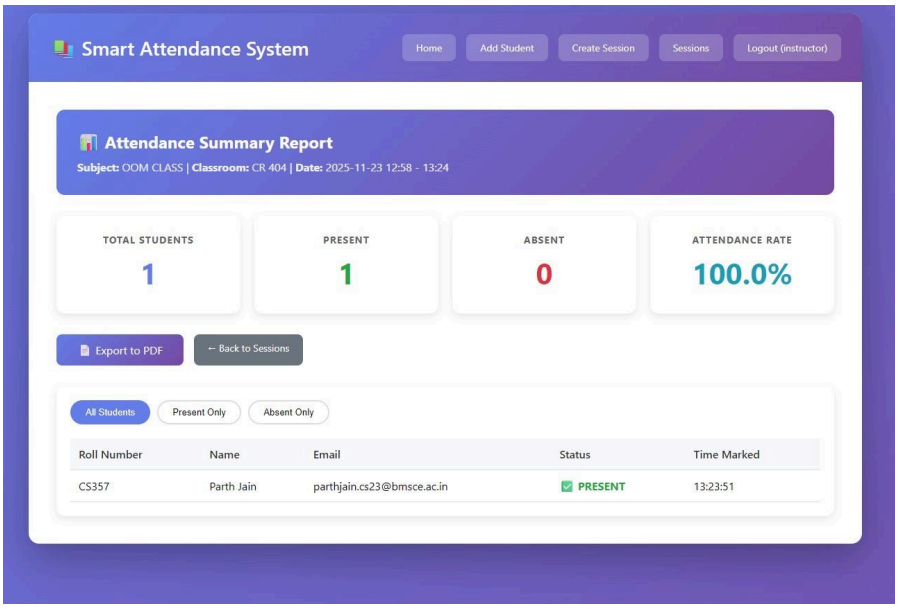


**Fig 6.3 Attendance Report**

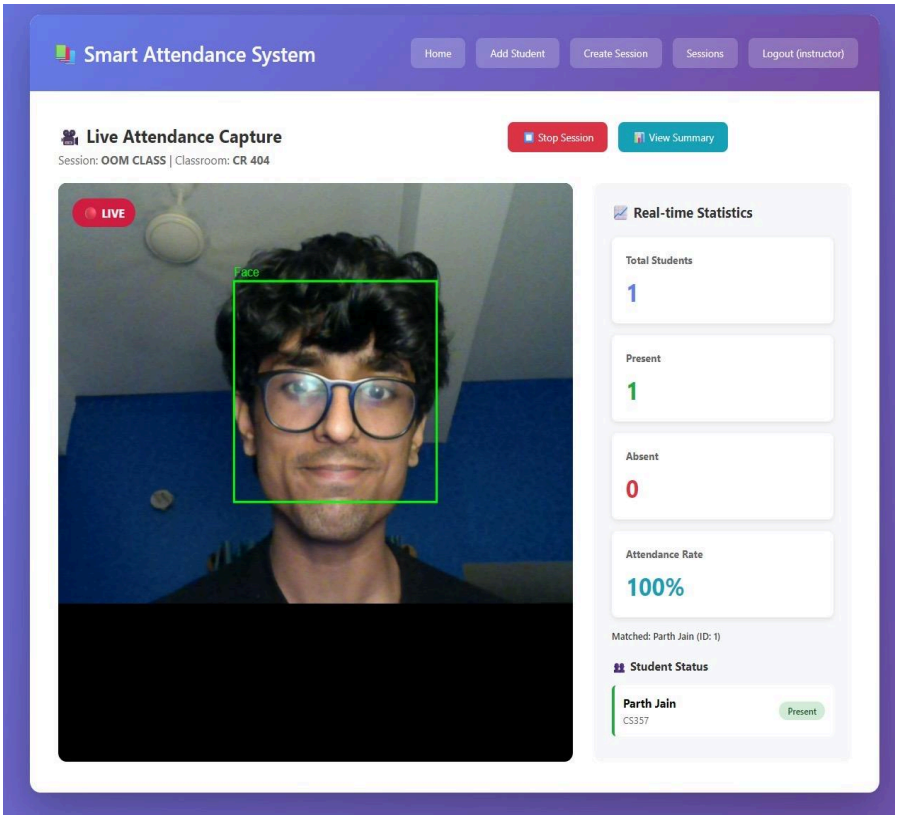


**Fig 6.4 Facial Recognition**