

## TABLE OF CONTENTS

	Page No.
1. Introduction	4-17
2. Literature Review/Survey	18-19
<ul style="list-style-type: none"><li>• Related works</li><li>• Problem Identification/Motivation</li></ul>	
3. Design/Modelling	20- 23
<ul style="list-style-type: none"><li>• Working principle</li><li>• Block Diagram/Model/Flow chart</li><li>• Methodology</li><li>• List &amp; specifications of components/material</li></ul>	
4. Results	24-27
<ul style="list-style-type: none"><li>• Face Detection and Recognition Output</li><li>• Attendance Marking in Excel</li><li>• Email Confirmation to Students</li></ul>	
5. Conclusion and Future Scope	28- 29
A. Appendix	30
B. References	31

## ABSTRACT

In the modern educational environment, tracking student attendance efficiently and accurately has become a significant challenge. Manual attendance systems are time-consuming, error-prone, and difficult to maintain over long periods. To overcome these limitations, this project introduces a **Face Recognition Attendance System** enhanced with real-time email notifications. The system leverages powerful technologies such as OpenCV for facial recognition, pandas for data handling, and SMTP email services for instant communication, thereby creating a smart, automated, and reliable attendance solution.

The system operates by capturing real-time video streams from a camera, detecting faces using a pre-trained **Haar Cascade Classifier**, and recognizing the detected faces through the **Local Binary Pattern Histogram (LBPH) algorithm**. Each recognized student's attendance is automatically marked as "Present," and an email notification is sent to the respective student confirming their attendance for the day, including details like the subject, teacher's name, and the current date. In case the student is not recognized or not present, the system intelligently marks their status as "Absent" while ensuring no duplicate entries are recorded.

All attendance records are systematically saved in an organized Excel file, segregated by date, providing easy access for future reference. The registered user details, including the student ID, name, and email address, are stored securely in a CSV file, allowing seamless integration with the facial recognition and email notification modules.

Moreover, to ensure reliability and maintainability, environment variables are used to store sensitive email credentials securely. The system also integrates robust exception handling to manage issues such as unrecognized faces, failed email delivery, or corrupt data entries.

This project demonstrates a practical application of computer vision, data science, and automation, addressing a real-world problem with innovative solutions. It holds immense potential for scaling into large educational institutions, corporate training sessions, or any scenario where mass attendance tracking is necessary. By minimizing manual efforts and maximizing efficiency, the Face Recognition Attendance System lays the foundation for a smarter, technologically advanced approach to student management

## 1. INTRODUCTION

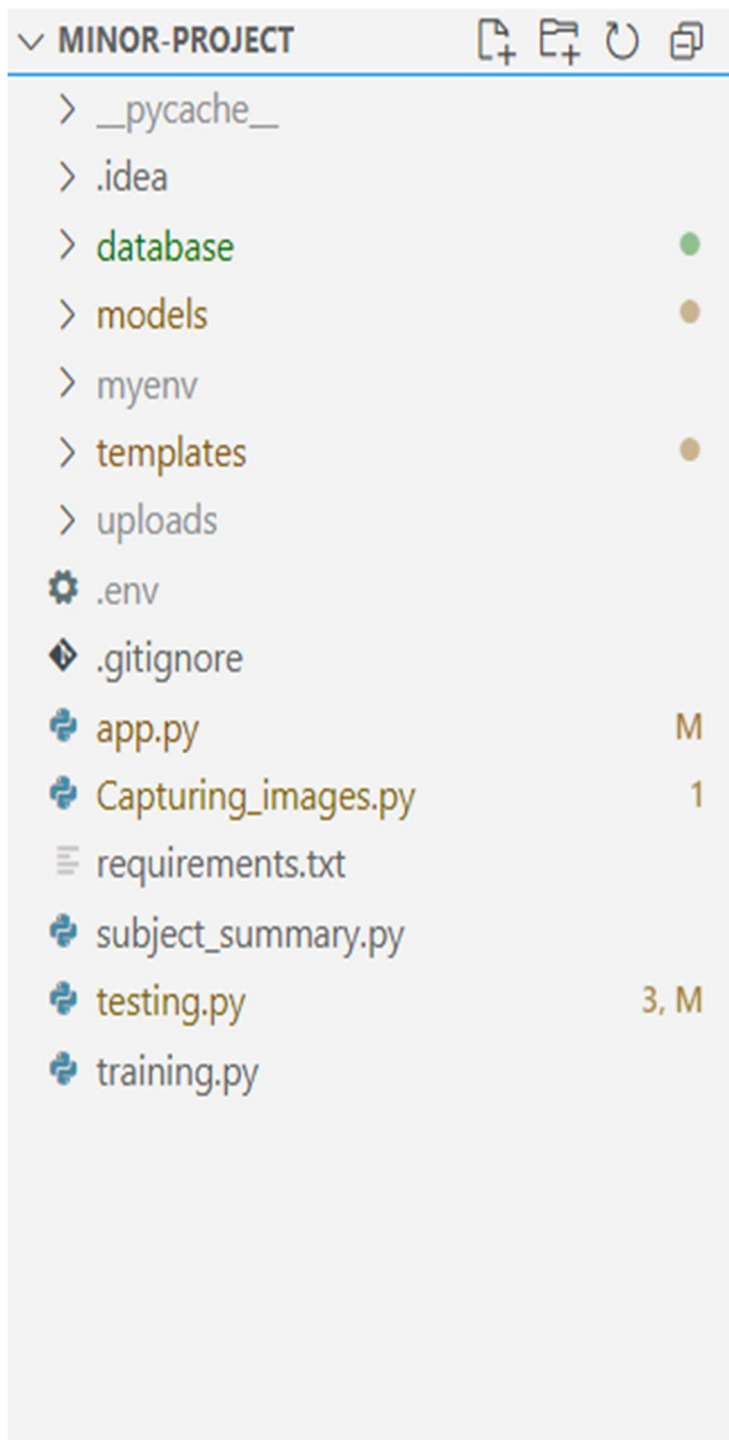
In today's rapidly advancing technological world, automation has become an essential part of every system, including educational institutions. Traditional attendance marking methods like manual roll calls are outdated, time-consuming, and prone to human error. Managing large volumes of attendance data is also challenging when using conventional approaches, often leading to inefficiency, manipulation, and loss of productivity.

To address these challenges, this project introduces a **Face Recognition-based Attendance System with Email Notification**, providing an intelligent, automated, and error-free solution for attendance management. This system uses **computer vision** techniques, **machine learning models**, and **email automation** to streamline the process. It ensures real-time attendance tracking, reduces administrative workload, and improves the overall management of student attendance.

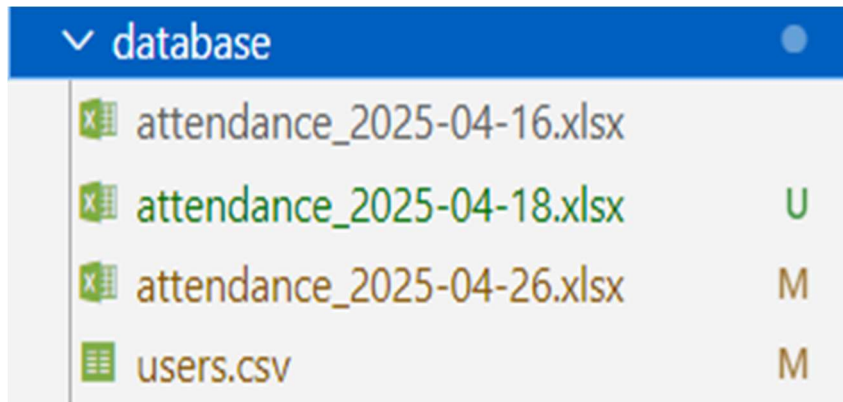
**The project combines several powerful tools and concepts:**

- **Face Detection:** Using **Haar Cascade Classifier** to detect faces from live webcam feeds.
- **Face Recognition:** Implementing the **Local Binary Pattern Histogram (LBPH)** algorithm for recognizing registered faces.
- **Data Management:** Storing user details and attendance records securely using **CSV** and **Excel files**.
- **Email Automation:** Sending real-time email notifications to students once their attendance is marked.
- **Security Measures:** Managing sensitive credentials through **environment variables**.
- **Exception Handling:** Ensuring smooth performance even in cases of missing faces, duplicate recognition, or communication errors.

## Project Structure Explanation:



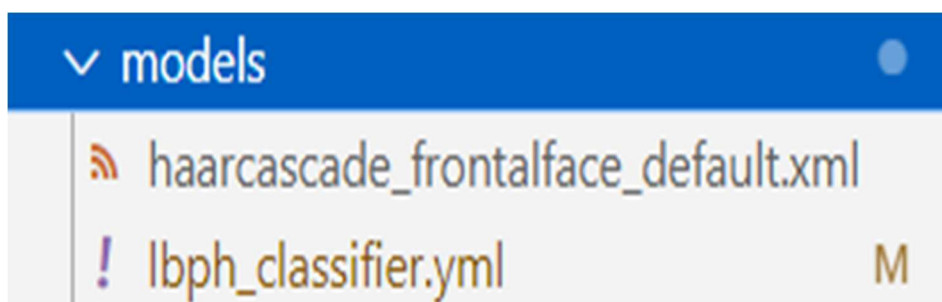
## 1. database/



The database/ directory is used for storing user details and attendance records. It includes:

- **users.csv:** This file stores essential information about each student, such as their ID, name, and email address. These details are used to map face images to specific individuals in the system.
- **attendance\_DATE.xlsx:** This is where attendance logs are stored for each session. The system records the attendance with timestamps, noting which students were present and absent. The filename is dynamic, reflecting the date of the session for easy organization.

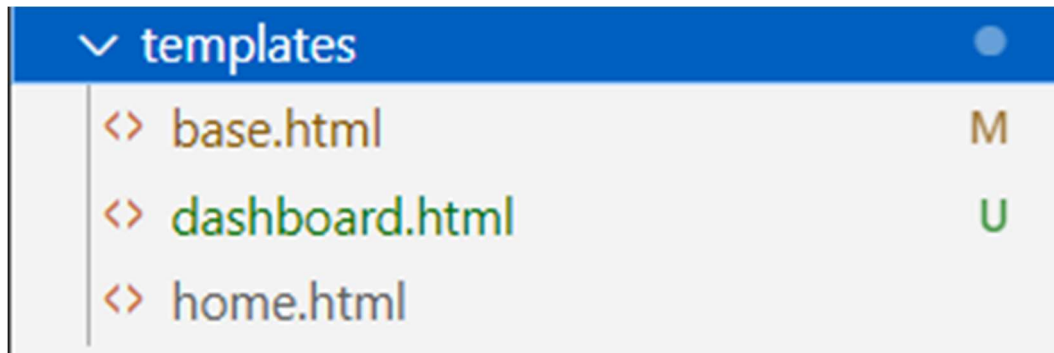
## 2. models/



This folder contains the pre-trained machine learning models required for **face detection** and **recognition**. It includes:

- **haarcascade\_frontalface\_default.xml:** This is the face detection model based on the Haar Cascade algorithm, which is used to identify faces in images captured from the live webcam feed.
- **lbph\_classifier.yml:** This file contains the trained **Local Binary Pattern Histogram (LBPH)** classifier, which is responsible for recognizing students' faces from the stored images.

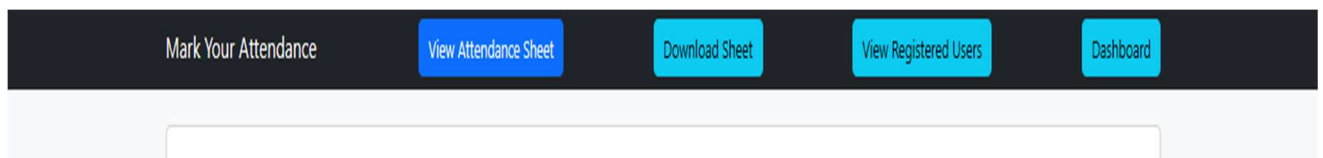
### 3. templates/



The **templates/** folder contains the HTML templates used by the Flask web application to render the pages of the Face Recognition-based Attendance System. These templates structure the web pages and provide the user interface for interacting with the system. Below is an explanation of each of the templates in this folder:

#### 3.1 base.html

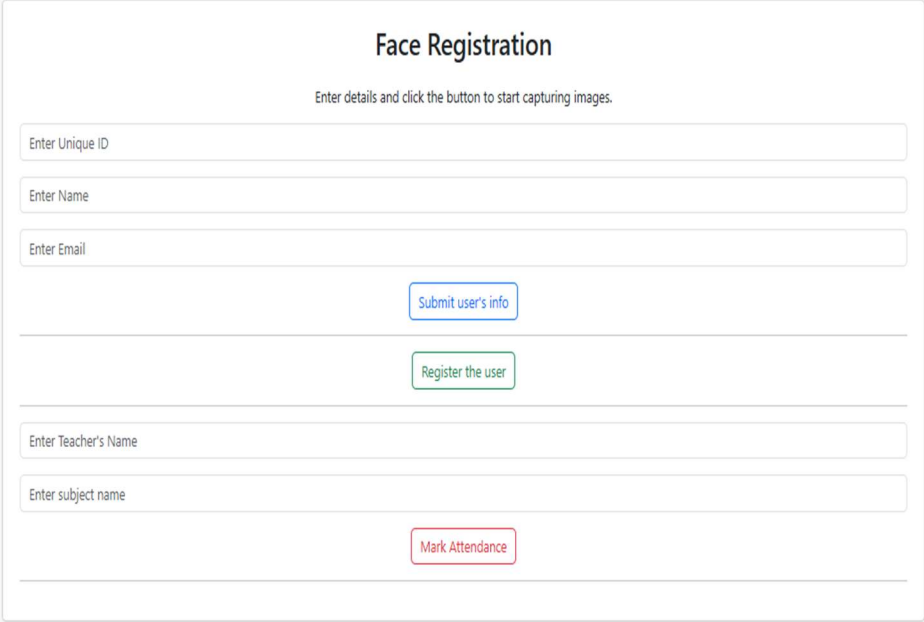
This template defines the base layout of the web pages, including a common navigation bar that is shared across all pages. The navigation bar provides links to various sections of the system. The key elements in this template are:



- **Mark Your Attendance:** Redirects to the home page, where the user can mark their attendance.
- **View Your Attendance:** Provides a link to view the student's own attendance records.
- **Download Your Attendance:** Allows the student to download their attendance record.
- **View Registered Users:** Links to a page where all registered users are listed.
- **Dashboard:** Redirects to the dashboard, displaying attendance logs for all students across all subjects.

The **base.html** template ensures a consistent and easy-to-navigate layout for all pages of the application.

### 3.2 home.html



The screenshot shows a web form titled "Face Registration" with the instruction "Enter details and click the button to start capturing images." The form contains three input fields for "Enter Unique ID", "Enter Name", and "Enter Email". Below these is a blue button labeled "Submit user's info". A horizontal line separates this section from the next, which has a green button labeled "Register the user". Another horizontal line follows, leading to two more input fields: "Enter Teacher's Name" and "Enter subject name". At the bottom of this section is a red button labeled "Mark Attendance".

The home.html template contains the core interaction elements for students and teachers. It includes:

- **Form for Entering Name, Email, and ID:** The student enters their personal details (name, email, and ID), and upon submission, the system begins the process of capturing 65 face images for training.
- **Train Button:** After the images are captured, the "Train" button triggers the process of training the LBPH face recognition model with the captured images.
- **Form for Teacher Name and Subject:** The teacher enters their name and the subject being taught, providing context for the attendance session.
- **Start Recognizing Face:** Once the forms are filled, the "Start Recognizing" button begins real-time face recognition. The camera is activated, and the system identifies students based on the faces detected during the session.

### 3.3 dashboard.html

Mark Your Attendance

View Attendance Sheet

Download Sheet

View Registered Users

Dashboard

## Attendance Dashboard

Date	Day	Teacher	Subject	Student Name	Status
2025-04-16	Wednesday	Meenakshi ma'am	Ai and knowledge representation	Abhi pratap singh	P
2025-04-18	Friday	Meenakshi ma'am	Ai and knowledge representation	Abhi pratap singh	P
2025-04-26	Saturday	Meenakshi ma'am	Ai and knowledge representation	Abhi pratap singh	P

The dashboard.html template displays a detailed view of attendance logs for all students. It provides an organized and comprehensive summary of the attendance records, allowing teachers and students to track attendance over time. Key features of this template include:

- **Attendance Logs:** Displays records of all attendance sessions, sorted by subject and date. This allows users to easily verify attendance.
- **Overview:** A general summary of the attendance status, including the number of classes attended, absences, and any other relevant attendance data.

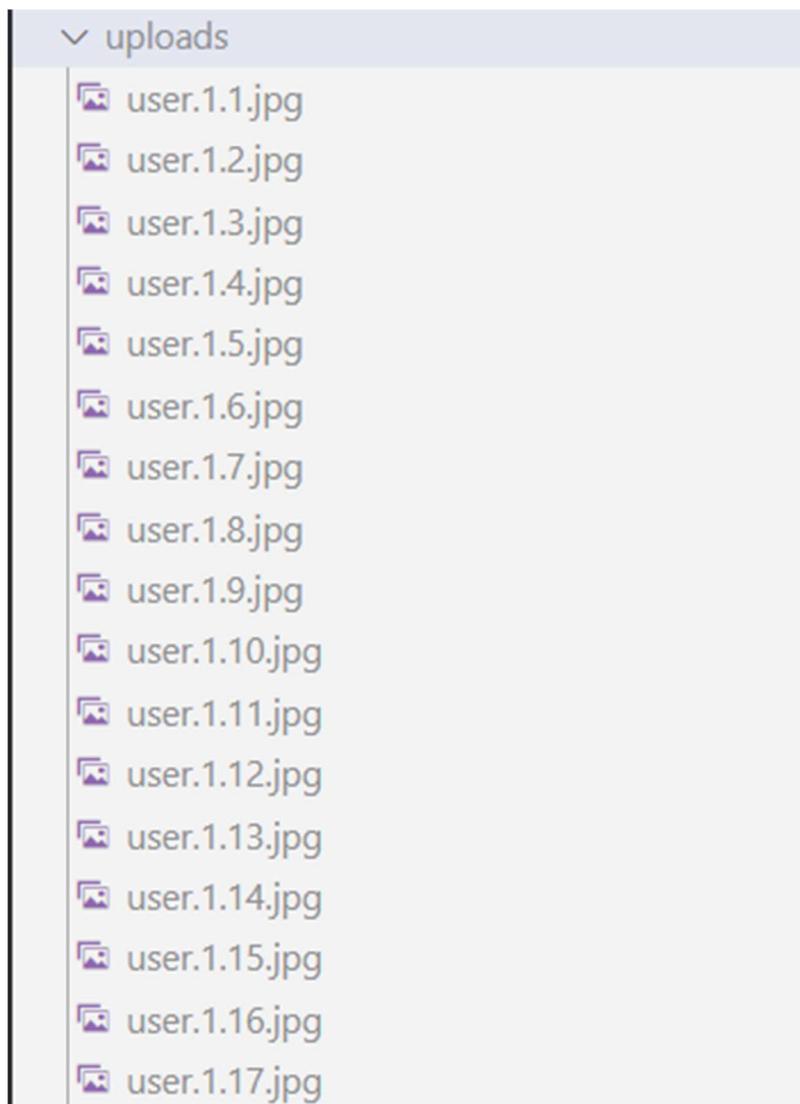
The **dashboard.html** template is designed to provide both students and faculty with an organized view of all attendance records for easy monitoring and reference.



#### 4. uploads/

The **uploads/** folder plays a crucial role in the Face Recognition-based Attendance System, as it stores the captured images of individuals' faces during the enrollment process. These images are essential for training the face recognition model.

**Purpose:** The uploads/ folder is where the face images of each person (student/teacher) are stored during the initial enrollment process. Each person's images are saved with a unique naming convention based on their ID.



**Naming Convention:** The images are saved using the format `user.{person_id}.{image_count}.jpg`, where:

- `{person_id}` is the unique identifier of the person.
- `{image_count}` represents the sequential number for each captured image (from 1 to 65).



**Folder Creation:** If the uploads/ folder does not already exist, the system automatically creates it to store the images. This ensures that the system can dynamically handle different users and their corresponding face images.

### Image Capture Workflow:

- The face detector (Haar Cascade Classifier) detects faces in the live video feed.
- For each detected face, a grayscale image of the face is captured and stored in the uploads/ folder.
- This process continues until **65 images** are captured for each individual, which are later used for training the face recognition model.

There is no hard and fix rule to take only 65 images, we are taking 65 images just for better training of the model, we can take or capture any number of images but we have to keep in mind time taken to capture images and overfitting of the model.

## 5. .env

This file is crucial for **security**. It stores **sensitive credentials** such as the email account username and password required for sending email notifications. By using environment variables, the system ensures that these credentials are never hard-coded, providing an additional layer of security.

## 6. Capturing\_images.py

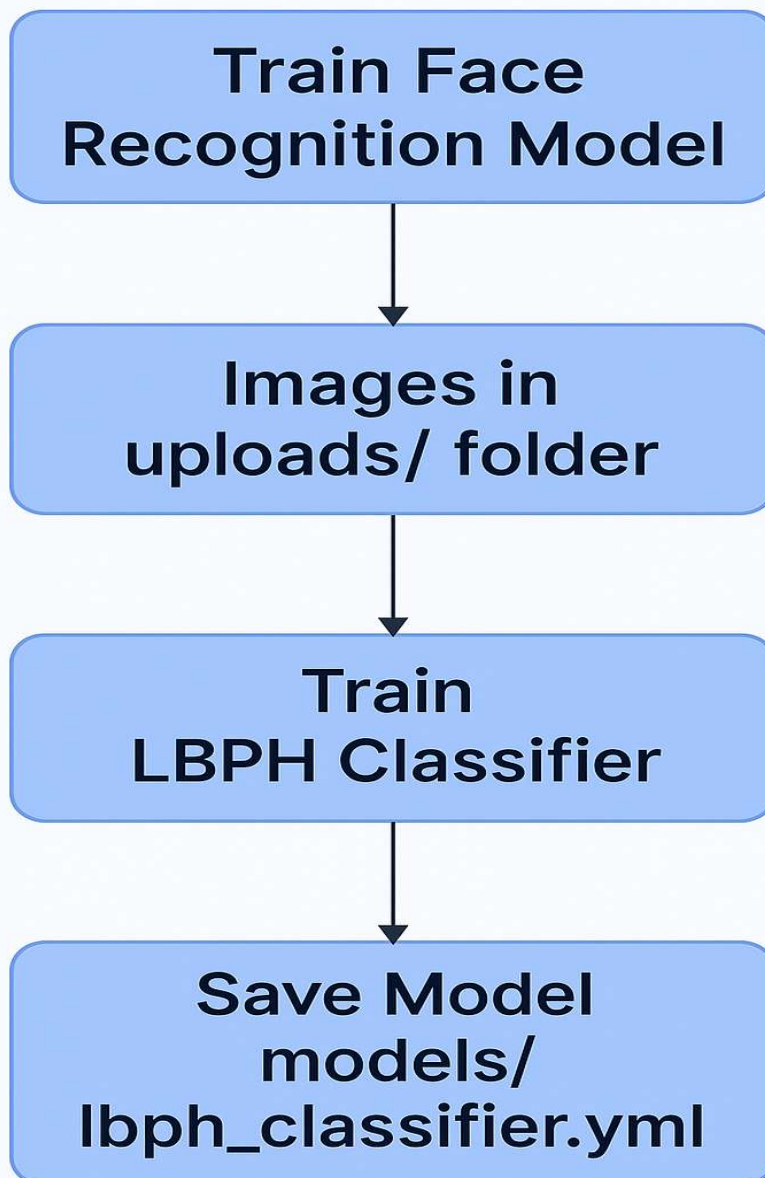
This script is responsible for capturing face images of students using a webcam. It performs the following key tasks:

- Takes **ID**, **Name**, and **Email** as command-line arguments.
- Checks if the provided ID already exists to avoid duplicate entries.
- Captures real-time video from the webcam.
- Detects faces in the video feed using Haar Cascade classifier (haarcascade\_frontalface\_default.xml).
- Extracts the detected faces, converts them to grayscale, and saves **65 face images** per student in the uploads/ folder with filenames like user.ID.count.jpg.
- Saves user details (ID, Name, Email) into the database/users.csv file.
- Displays a live window showing face detection with green rectangles.
- Stops automatically after capturing 65 images or manually by pressing 'q'.

This ensures each student has a sufficient number of labeled face samples for accurate training of the face recognition model.

## 7.. training.py

This file is used to **train the face recognition model**. Using the images captured in the uploads/ folder, it trains the LBPH classifier. Once trained, the model is saved in the models/ folder as lbph\_classifier.yml to be used during real-time testing and attendance marking.



## 8. testing.py - Real-time Testing and Attendance Marking

This script is responsible for **real-time testing** and **attendance marking**. It leverages the **Haar Cascade Classifier** for face detection and **LBPH (Local Binary Pattern Histogram)** classifier for face recognition. Once a student's face is detected and recognized, their **attendance is marked automatically**, and an **email notification** is sent to confirm their presence.

### Here's how it works:

1. **Email Notifications:** When a student's face is identified, an email is sent to the student confirming their attendance. The email contains the date, subject, and teacher's name.
2. **Face Detection and Recognition:** The webcam continuously captures frames. The script detects faces using the Haar Cascade Classifier and recognizes them using the LBPH face recognizer model.
3. **Attendance Marking:** When a face is detected, the script checks if the person has already been marked as present. If not, it adds their name to the attendance list and sends an email confirmation.
4. **Data Handling:**
  - The attendance data is saved in an **Excel file**, with columns for the **Date, Day, Teacher, Subject, Student Name**, and **Status**.
  - If a student's attendance has already been recorded for the day, the system will not mark them again.
5. **Environmental Setup:**
  - The .env file contains email credentials used to send the confirmation email. It loads the environment variables securely.

## Features:

- **Email Confirmation:** After marking attendance for a student, the script sends a **confirmation email** to the student's registered email address.
- **Real-time Display:** It displays the captured video feed with face detection bounding boxes. Once a student is recognized, their name and confidence level are displayed on the screen.
- **Automatic Attendance Marking:** The system automatically marks attendance for recognized students and prevents duplicate markings.
- **Handling Low Confidence:** If the recognition confidence is too low, the student is not marked, and an alert is displayed saying "Face not recognized."
- **Missing Attendance Handling:** At the end of the session, it checks if there were students who were not marked as present, and marks them as absent.

## 9. app.py (Flask App)

The app.py file is the main backend of a Flask application for attendance management. It facilitates user registration, face recognition training, and real-time attendance tracking. The app includes several routes:

- **Home Route (/):** Renders the home page where users can interact with the app.
- **Capture Route (/capture):** Captures user information (ID, name, email) and initiates image capturing for face recognition.
- **Training Route (/train):** Runs the training script to train the face recognition model.
- **Testing Route (/start\_testing):** Starts real-time attendance marking based on the teacher's name and subject.
- **Open and Download Routes (/open\_excel, /download\_attendance):** Allows users to open and download the attendance Excel files.
- **Registered Users Route (/registered\_users):** Opens a CSV file containing registered users.
- **Dashboard Route (/dashboard):** Displays all attendance records in a dashboard.

It uses subprocess to execute external Python scripts for image capturing, training, and testing. The app handles file operations to manage attendance records and user data effectively.

## Flask powered website:

[Mark Your Attendance](#)[View Attendance Sheet](#)[Download Sheet](#)[View Registered Users](#)[Dashboard](#)

### Face Registration

Enter details and click the button to start capturing images.



## **2. REVIEW OF LITERATURE**

### **2.1 Related Works**

Several research studies and projects have been conducted to implement face - recognition-based attendance systems using machine learning and computer vision technologies. These systems have been explored for educational institutions, businesses, and organizations for automatic attendance tracking.

- **Automatic Attendance Systems**

Several institutions have adopted facial recognition technology to automatically mark attendance, reducing human intervention. These systems typically work by capturing facial images of individuals and matching them with a pre-registered database. The attendance is automatically recorded once a match is found.

- **Technological Advancements**

Recent advancements in machine learning and computer vision, particularly in face detection and recognition algorithms (like OpenCV and deep learning techniques), have made these systems more accurate. The integration of these technologies has helped improve the system's ability to identify individuals in different lighting conditions and angles, making the system more robust.

- **Application in Educational Institutions**

Face recognition-based attendance systems are widely used in schools and universities for both students and teachers. They offer a hands-free method of tracking attendance, ensuring more accurate records while saving time for both teachers and students.

## **2.2 Problem Identification/Motivation**

The motivation behind implementing a face recognition-based attendance system stems from the limitations of traditional attendance methods. Some key challenges and motivations for the system include:

### **1. Inefficiency of Manual Attendance Systems**

Traditional attendance systems require teachers to manually call out student names or mark attendance on paper. This process is time-consuming and prone to errors, such as marking students absent when they are present, or vice versa.

### **2. Human Error and Proxy Attendance**

In manual systems, there is always a chance of human error or proxy attendance, where a student marks the attendance for another student. This issue is eliminated in face recognition systems, which automatically detect and verify individuals.

### **3. Need for Automation**

Automation of attendance marking allows teachers to focus more on teaching rather than administrative tasks. By using a face recognition system, institutions can streamline the process, making it faster and more efficient.

### **4. Adoption in Universities**

Face recognition-based attendance systems have been implemented in universities, including my own, where the system is used to mark attendance for teachers and students. This system automatically records attendance based on the face detected and cross-references it with a database of registered users. This implementation has proven effective in ensuring timely and accurate attendance tracking.

### **5. Real-Time Monitoring**

One of the significant advantages of face recognition is real-time tracking. As soon as the teacher or student enters the classroom, their attendance is automatically marked. This removes the need for manual intervention and ensures that attendance is recorded immediately.

### 3. DESIGN AND METHODOLOGY

#### 3.1. Working Principle

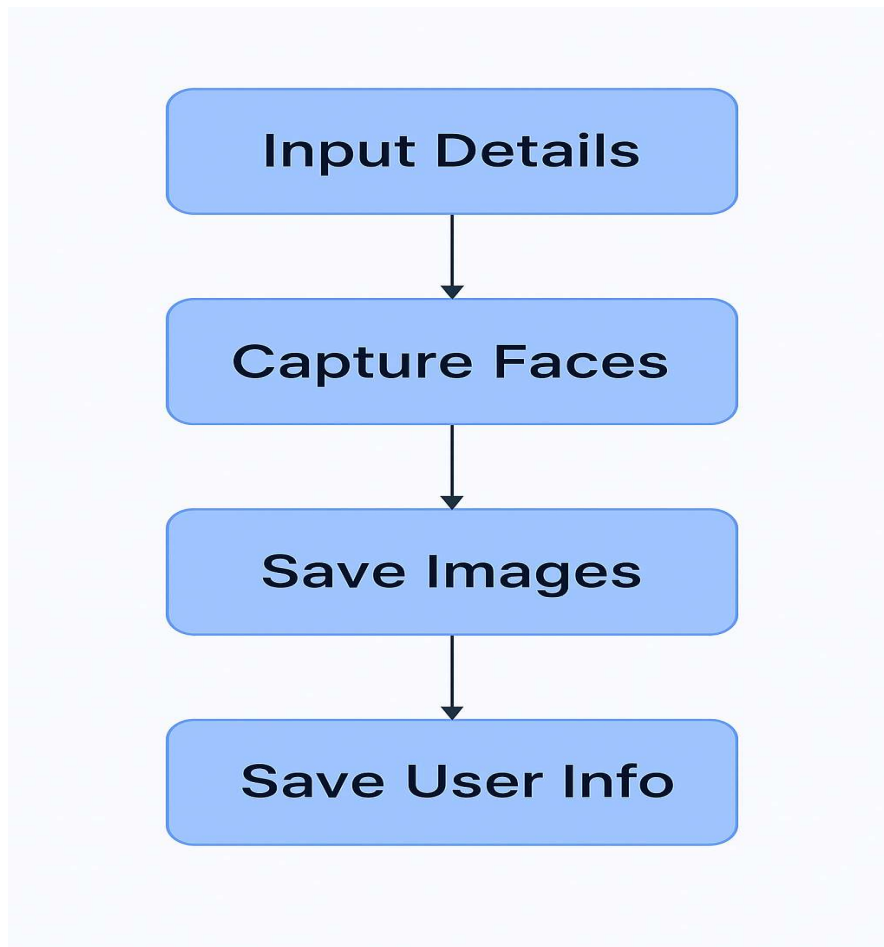
The system operates on the principle of **face detection and recognition** using machine learning models, particularly the **LBPH (Local Binary Pattern Histogram)** classifier.

When a student enters the camera frame:

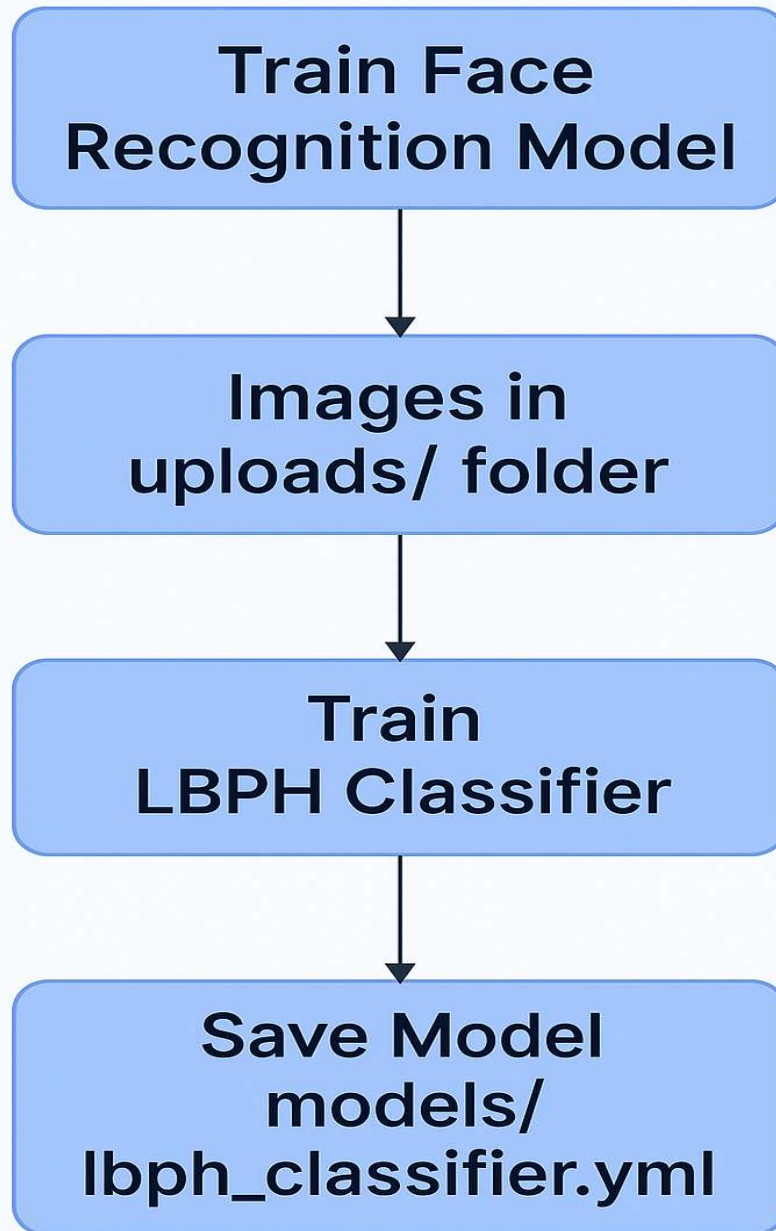
- Their face is detected.
- The detected face is recognized by comparing it against pre-trained images.
- Once recognized, attendance is logged, and a confirmation email is sent automatically.

This ensures **automatic, real-time, contactless attendance** marking with minimal human intervention.

##### 3.1.1 Flow diagram for image capturing:



**After image capturing:**



### 3.2. Methodology

The project methodology is divided into **several key phases**:

- **Phase 1: User Registration**
  - Student details (ID, Name, Email) are captured and stored.
  - Multiple face images are taken to create a dataset.
- **Phase 2: Face Image Processing**
  - Images are resized, converted to grayscale, and saved for consistency.
- **Phase 3: Model Training**
  - The LBPH algorithm is used to train on the captured faces.
  - A .yml file is generated (lbph\_classifier.yml).
- **Phase 4: Real-Time Recognition and Attendance Marking**
  - Live video feed is processed frame by frame.
  - Faces are detected and matched against the trained model.
  - Attendance is marked, and emails are dispatched for confirmation.
- **Phase 5: Storage and Reporting**
  - Attendance records are stored in an Excel file (.xlsx format) for each day.
  - Presence or absence is automatically recorded.

This step-by-step approach ensures system **accuracy, efficiency, and scalability**.

### 3.3. List & Specifications of Components / Material

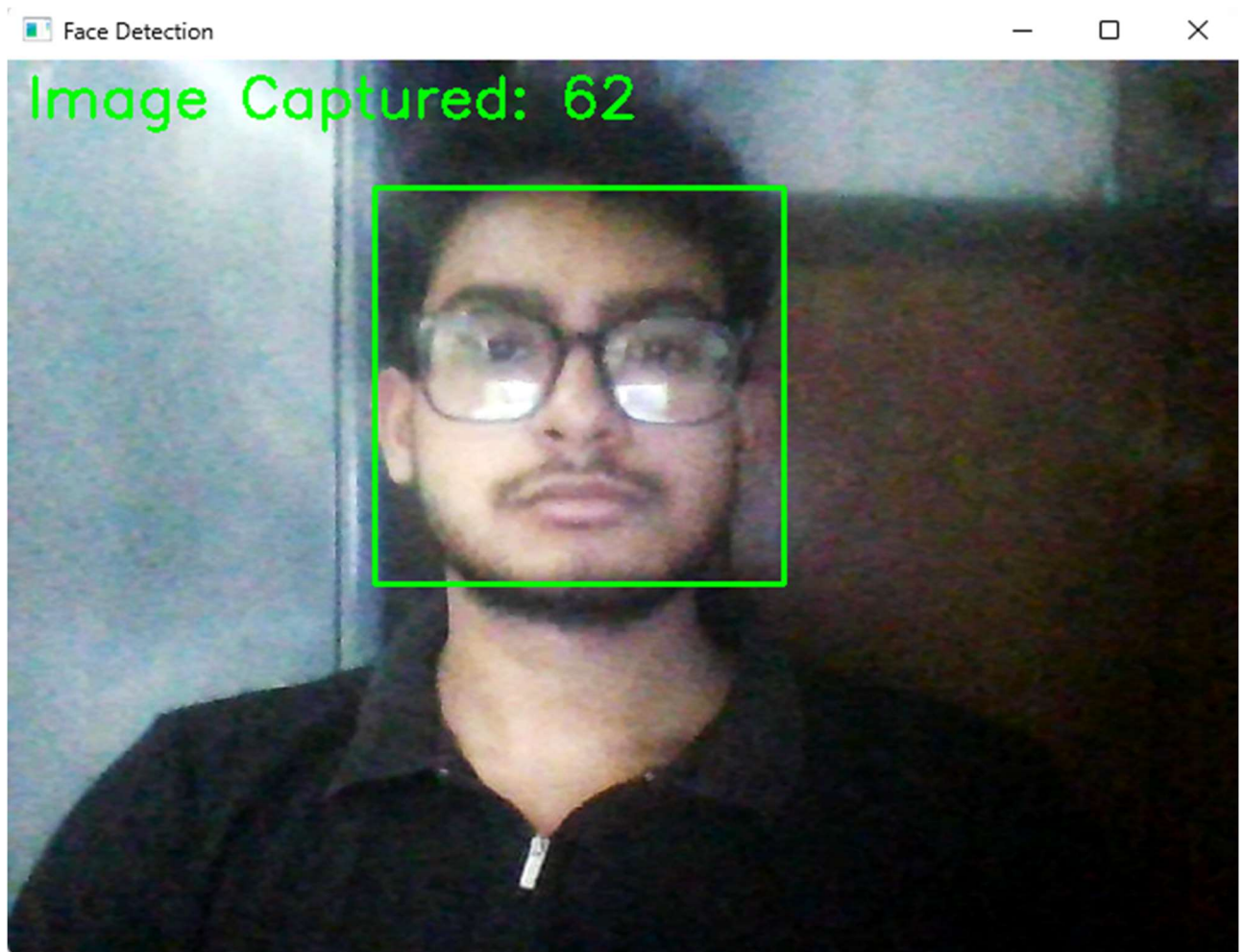
Component / Material	Specifications
Webcam	Minimum 720p resolution recommended
Computer/Laptop	Minimum i3 Processor, 8GB RAM
Python	Version 3.7 or higher
OpenCV Library	For face detection and recognition
Pandas Library	For data manipulation
Yagmail Library	For email sending
dotenv Library	For securely loading email credentials
SMTP Server (Gmail, Outlook etc.)	For email services
Local Storage	Folders: uploads/, models/, database/
Excel (.xlsx) support	Via pandas and openpyxl libraries

## 4. Experimental Results

In this section, we present the outputs obtained after the implementation of the system.

It includes snapshots, performance metrics, and observations from the testing phase.

### 4.1 Face Detection and Image Capturing



The system successfully detected faces in real-time and captured multiple samples for each user.

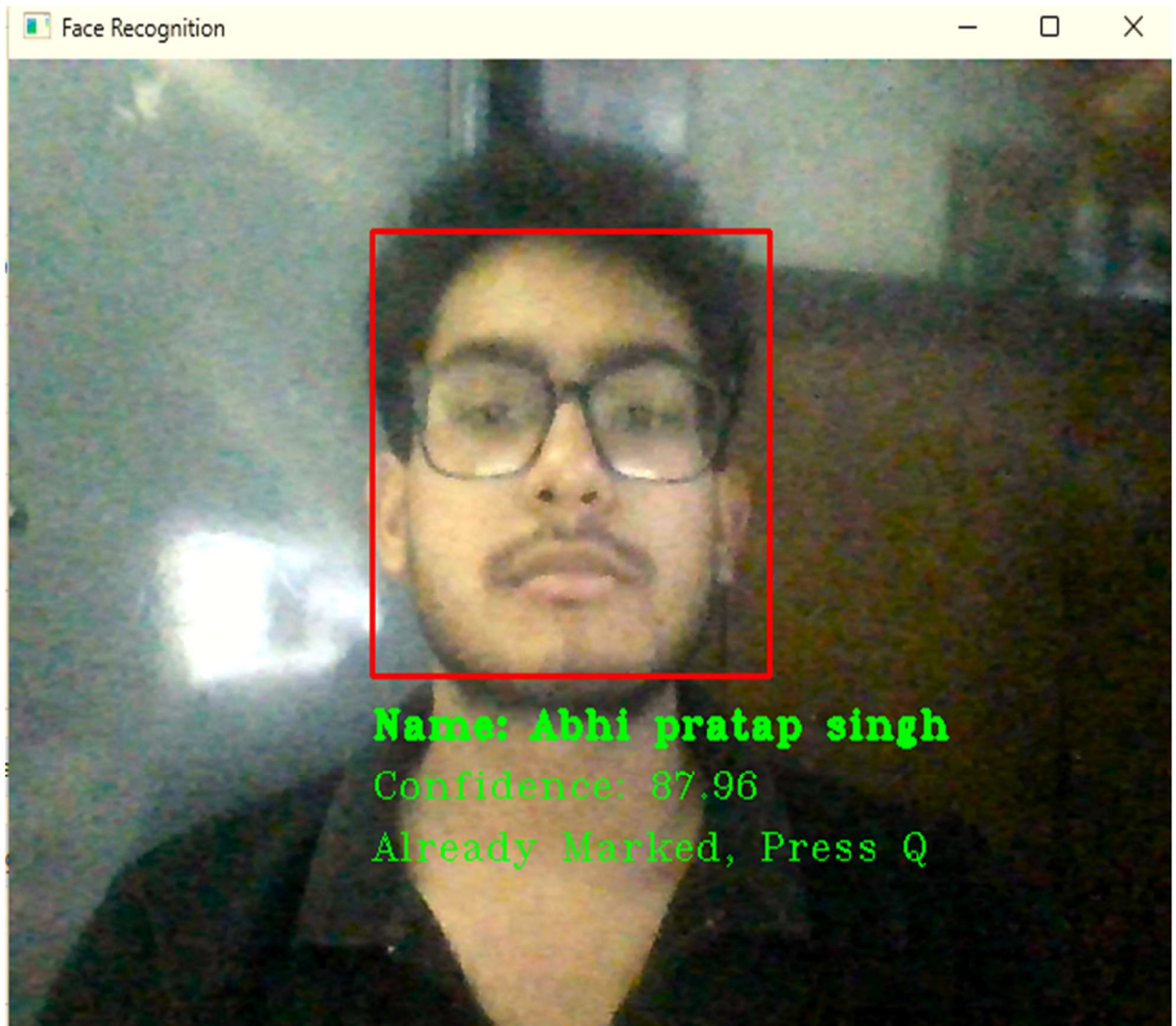
## 4.2 Training Phase Output

```
! lbph_classifier.yml M ×
models > ! lbph_classifier.yml
1  %YAML:1.0
2  ---
3  opencv_lbphfaces:
4    threshold: 1.7976931348623157e+308
5    radius: 1
6    neighbors: 8
7    grid_x: 8
8    grid_y: 8
9    histograms:
10     - !!opencv-matrix
11       rows: 1
12       cols: 16384
13       dt: f
14       data: [ 0.0486111119, 0.0069444445, 0., 0., 0.0069444445, 0.,
15              0., 0., 0.013888889, 0., 0., 0., 0.0069444445, 0., 0., 0.,
16              0.027777778, 0., 0., 0., 0., 0., 0., 0.0069444445, 0.,
17              0., 0., 0.020833334, 0., 0.0069444445, 0.0069444445, 0., 0.,
18              0., 0., 0., 0., 0., 0., 0.0069444445, 0., 0., 0., 0.,
19              0., 0., 0.0069444445, 0., 0.0069444445, 0., 0.0069444445,
20              0., 0., 0., 0.0972222239, 0.0069444445, 0., 0.,
21              0.0416666679, 0., 0.013888889, 0.027777778, 0.013888889, 0.,
22              0., 0., 0.0069444445, 0.0069444445, 0., 0., 0., 0., 0., 0.,
23              0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
24              0., 0.0069444445, 0., 0., 0., 0.0069444445, 0., 0., 0., 0.,
25              0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.013888889,
26              0.013888889, 0., 0., 0., 0., 0., 0., 0.138888896, 0., 0.]
```

- The captured images were processed using the LBPH algorithm.
- The trained model file lbph\_classifier.yml was generated and saved for future use.



### 4.3 Attendance Marking and Email Notification



- Upon successful recognition, attendance was marked automatically.

A	B	C	D	E	F	G	H	I	J	K
Date	Day	Teacher	Subject	Student Name	Status					
2025-04-27	Sunday	Meenakshi ma'am	AI and knowledge representation	Abhi pratap singh	P					

## Attendance Marked Successfully

External

Inbox



set.batch.2022@gm... 9:07 pm  
to me ▾



Hello Abhi pratap singh,

Your attendance has been successfully marked.

- Date: 2025-04-27
- Teacher: Meenakshi ma'am
- Subject: Attendance Marked Successfully
- Status: Present

Regards,  
Face Recognition Attendance System

- Simultaneously, an email was dispatched to the concerned authority with the attendance record.

## 5. Conclusion and Future Scope

### Conclusion

The project successfully demonstrates an **automated attendance system** based on **facial recognition technology**.

By using computer vision and machine learning, the system captures student images, trains a recognition model, and marks attendance accurately in real-time.

It significantly reduces manual work, minimizes errors, and improves the overall efficiency of attendance management.

Additionally, features like **automatic email notifications** make record-keeping easier and more reliable.

This project offers a **practical, scalable, and user-friendly** solution that can be deployed in schools, colleges, offices, and other organizations where attendance tracking is essential.

## Future Scope

While the system performs effectively in its current state, there are several avenues for further enhancement and expansion:

- **Integration of Deep Learning Models:**  
To improve face recognition accuracy even under varying lighting conditions, facial expressions, or occlusions, advanced deep learning models like **Convolutional Neural Networks (CNNs)** or pre-trained architectures like **FaceNet** and **VGG-Face** can be incorporated.
- **Incorporating SMS Alerts:**  
Along with email notifications, **SMS-based alerts** can be integrated to provide real-time attendance updates to students, parents, or administrators, thereby increasing system responsiveness and usability.
- **Attendance Visualization Dashboards:**  
Developing interactive **dashboards** for real-time monitoring of attendance records, daily summaries, and trend analytics would empower administrators with better data insights and quick decision-making.
- **Cloud Integration:**  
Hosting the system on the **cloud** would allow remote access, centralized attendance management across multiple branches, and secure data backups.
- **Security Enhancements:**  
To prevent spoofing attacks (like using photos/videos for fake attendance), future versions can implement **liveness detection** to ensure the presence of a live person during authentication.
- **Mobile Application Development:**  
Creating a dedicated **mobile app** would enable users to receive notifications, view attendance records, and perform administrative actions remotely, increasing overall system flexibility.

Through these future enhancements, the system can become even more **robust, intelligent, and adaptable** to the evolving needs of modern institutions and organizations.

## Appendix

- **Python Version:** 3.12.6
- **Libraries Used:** OpenCV, NumPy, pandas, yagmail, flask, os, datetime, webbrowser, csv, dotenv, sys.
- **Bootstrap** was used for html templates.
- **Hardware Used:** Webcam, PC/Desktop
- **Operating System:** Windows

GitHub Repository:

[https://github.com/Abhipratapsingh123/Face\\_recognition\\_project](https://github.com/Abhipratapsingh123/Face_recognition_project)

The repository contains complete source code, and commit history of the project.

## **REFERENCES**

- (1) Jones Granatyr "Computer Vision Masterclass" , Udemy,  
<https://www.udemy.com/course/computer-vision-masterclass/>, Accessed on  
8<sup>th</sup> February 2025.
- (2) Jose Portilla "Python anf Flask Bootcamp" , Udemy,  
<https://www.udemy.com/course/python-and-flask-bootcamp-create-websites-using-flask/>, Accessed on 25<sup>th</sup> March 2025.
- (3) Ardit Sulce "Automate Everything With Python" , Udemy,  
<https://www.udemy.com/course/python-and-flask-bootcamp-create-websites-using-flask/>, Accessed on 6<sup>th</sup> August 2024.