FACE DETECTION USING PYTHON

A Project report submitted in partial fulfillment of the requirements for the Degree of

## Bachelor of Technology

in

## Computer Engineering

Submitted by

**Pratik Sanjay Pawar**

**Yogesh Sanjeev Patil**

**Lalit Kishor Chaudhari**

**Gopal Chandarakant Sindane**



**DEPARTMENT OF COMPUTER ENGINEERING**

S.S.V.P.S.’s B.S. DEORE COLLEGE OF ENGINEERING, DHULE 2024-25

I



lOMoARcPSD|346 384 61

FACE DETECTION USING PYTHON

A Project report submitted in partial fulfillment of the requirements for the Degree of

## Bachelor of Technology

in

## Computer Engineering

Submitted by

**Pratik Sanjay Pawar**

**Yogesh Sanjeev Patil**

**Lalit Kishor Chaudhari**

**Gopal Chandrakant Saindane**

Guided by

## Prof. B. D. Patil



### DEPARTMENT OF COMPUTER ENGINEERING

S.S.V.P.S.’s B.S. DEORE COLLEGE OF ENGINEERING, DHULE 2024-25

## S.S.V.P.S.’s B.S. DEORE COLLEGE OF ENGINEERING, DHULE

**DEPARTMENT OF COMPUTER ENGINEERING**

# CERTIFICATE

This is to certify that the Major Project I entitled “Face Detection Using Python ” has been carried out by

**Pratik Sanjay Pawar**

**Yogesh Sanjeev Patil**

**Lalit Kishor Chaudhari**

**Gopal Chandrakant Saindane**

under my guidance in partial fulfillment of the degree of Bachelor of Technology in Computer Engineering of Dr. Babasaheb Ambedkar Technological University, Lonere during the academic year 2024-25. To the best of my knowledge and belief this work has not been submitted elsewhere for the award of any other degree.

### Date:

**Place:** Dhule

### Guide

Prof. **B. D. Patil**

### Head Principal

Prof. Dr. I. S. Borse Prof. Dr. Hitendra D. Patil

lOMoARcPSD|346 384 61

# ACKNOWLEDGEMENT

It is our proud privilege to express our profound gratitude our **Principal Prof. Dr. H. D. Patil**, **Prof. Dr. I.S Borse** , Head of Department of Computer Engineering and entire management of our College, S.S.V.P.S.’s B. S. Deore College of Engineering, Dhule , and the teachers of the institute for providing us with the opportunity to avail ourselves of the excellent facilities and infrastructure.

The knowledge and values inculcated have proved to be ofimmense help at the very start of our career. Special thanks to the Hon’ble Founder, S.S.V.P.S.’s B. S. Deore College of Engineering, Dhule for having provided us with an excellent infrastructure.

We are grateful to **Prof. B. D. Patil** for their astute guidance, constant encouragement and sincere support for this project work.

We will be failing in duty if we do not acknowledge with grateful thanks to the authors of the references and other literatures referred to in this Major Project. Sincere thanks to all my family members, seniors and friends for their support and assistance throughout the project.

Thanking You,

Pratik Sanjay Pawar

Yogesh Sanjeev Patil

Lalit Kishor Chaudhari

Gopal Chandrakant Saindane

## CONTENTS

**ABSTRACT**……………………………………………….…………………………………i

**LIST OF FIGURES**………………………………………………………………………... ii

**CHAPTER 1 INTRODUCTION…**……………………………………………………...…4

1.1 Introduction…....………………………………………………………………….5

1.2 Objective ………...…………………….…………………………………………6

1.3 Student Attendance System ……………….…………………………………..…6

**CHAPTER 2 LITERATURE REVIEW**…………………………………………………...9

**CHAPTER 3 SYSTEM ADVNTAGES AND DISADVANTAGES**…….…………..........15

3.1 Advantages of face detection ……………….………………………………...….15

3.2 Disadvantages of face detection ….…………………………………………..…..17

3.3 Choosing The Right Face Detection Model ……..………..……...........................20

3.4 Optimization Techniques……………………………………………………....…21

**CHAPTER 4 SYSTEM TECHNOLOGIES**……………………..……………..……….....22

4.1 Python……………………………………………………………………… …….23

4.2 Opencv …………………………………………………………………………....24

4.3 Haar Cascade Classifier …………………………………………..........................25

4.4 Numpy …………………………………………………………………………....27

4.5 Matplotlib ………………………………………………………………..…….....27

**CHAPTER 5 SYSTEM DESIGN** ……………………………………………...……….......22

5.1 Introduction To System Design …………………..………………………......…..23

5.2 Key Steps In The Design Approach ………………………………………..….....24

5.3 Data Flow Diagram ………………………………………………….…...……....25

5.4 System Flow Diagram ………………………………………………....………....27

5.5 Uml Class Diagram ………………………………………….…...........................25

5.6 Explanation of the Sequence Diagram……………….……………………...…....27

**CONCLUSION**………………………………………………………………...….………...30

**REFERENCES**……………………………………………………………..…….…………31

## ABSTRACT

*In colleges, universities, organizations, schools, and offices, taking attendance is one of the most important tasks that must be done on a daily basis. The majority of the time, it is done manually, such as by calling by name or by roll number. The main goal of this project is to create a Face Recognition-based attendance system that will turn this manual process into an automated one. This project meets the requirements for bringing modernization to the way attendance is handled, as well as the criteria for time management. This device is installed in the classroom, where and student's information, such as name, roll number, class, sec, and photographs, is trained. The images are extracted using Open CV. Before the start of the corresponding class, the student can approach the machine, which will begin taking pictures and comparing them to the qualified dataset. Logitech C270 web camera and NVIDIA Jetson Nano Developer kit were used in this project as the camera and processing board. The image is processed as follows: first, faces are identified using a Haarcascade classifier, then faces are recognized using the LBPH (Local Binary Pattern Histogram) Algorithm, histogram data is checked against an established dataset, and the device automatically labels attendance. An Excel sheet is developed, and it is updated every hour with the information from the respective class instructor.*

**Keywords**: Face Detection, Face Recognition, HaarCascade classifier, NVIDIA Jetson Nano

## LIST OF FIGURES:

Figure 2.1 Haar Feature………………………………………………………………………….15

Figure 2.2 Integral of Image……………………………………………………………………..15

Figure 5.3.1 Level 0: Detailed DFD……………………...…………………………….………..16

Figure 5.3.2 Level 1: Detailed DFD……………………...………………….…………………..16

Figure 5.4 SYSTEM FLOW DIAGRAM…………………………………………………….....17

Figure 5.5 UML CLASS DIAGRAM ……………………………….………………………….17

Figure 5.5.1 Sequence Diagram………………………………..………………………..………19

Figure 5.7.1 System Flowchart……………….………………………………………………….20

**CHAPTER 1**

**INTRODUCTION**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

#### INTRODUCTION

Face detection is one of the most important tasks in the field of computer vision and image processing. It involves the identification and localization of human faces within digital images or video streams. The ability to automatically detect faces has wide applications across various fields, including security systems, human-computer interaction, social media platforms, and entertainment, among others. In recent years, advances in computer vision have made face detection more efficient and accurate, enabling real-time systems. A fundamental aspect of face detection is distinguishing human faces from other objects in the image, a task that becomes increasingly challenging in complex environments with varying lighting, orientations, and occlusions. One of the most widely used methods for face detection is through machine learning, specifically using Haar Cascades, a type of object detection algorithm based on feature classification. OpenCV, an open-source computer vision library, provides an efficient implementation of Haar Cascade classifiers, making it easy for developers to incorporate face detection capabilities into their applications. This report demonstrates how to implement a face detection system using Python and OpenCV. The system leverages the Haar Cascade classifier to detect faces in both static images and video streams. The goal is to provide an accessible and practical solution for detecting faces in various real-world applications.

**1.2 OBJECTIVES**

* To develop a Python-based face detection system using OpenCV.
* To demonstrate the use of Haar Cascade classifiers in detecting faces.
* To build a simple, efficient solution for detecting faces in both images and real-time video streams.

By implementing this face detection system, the project aims to showcase the fundamental techniques and tools used in the field of computer vision while also offering a foundation for more advanced facial recognition or emotion detection systems.

#### 1.3 STUDENT ATTENDANCE SYSTEM

Arun Katara et al. (2017) mentioned disadvantages of RFID (Radio Frequency Identification) card system, fingerprint system and iris recognition system. RFID card system is implemented due to its simplicity. However, the user tends to help their friends to check in as long as they have their friend’s ID card. The fingerprint system is indeed effective but not efficient because it takes time for the verification process so the user has to line up and perform the verification one by one. However for face recognition, the human face is always exposed and contain less information compared to iris. Iris recognition system which contains more detail might invade the privacy of the user. Voice recognition is available, but it is less accurate compared to other methods. Hence, face recognition system is suggested to be implemented in the student attendance system. The system captures real-time video streams using a Logitech C270 webcam, detects faces with the Haar Cascade Classifier, and recognizes students using a trained LBPH model. Recognized students are logged in an SQLite database, with attendance records exported to an Excel sheet hourly for each class. The Flask web interface allows instructors to view attendance, upload new student images for model training, and manage records securely. The system is hosted on the NVIDIA Jetson Nano, ensuring low-latency processing suitable for real-time applications. The choice of face recognition over other methods is justified by its balance of accuracy, efficiency, and user acceptance. While iris recognition may offer higher precision, its invasiveness and cost make it less practical for widespread use in educational institutions. Similarly, RFID and fingerprint systems, while simpler, cannot match the speed and security of face recognition in preventing proxy attendance. The system also addresses ethical concerns by implementing user consent protocols, encrypting facial data, and restricting access to authorized personnel, ensuring compliance with privacy regulations like GDPR. By automating attendance, the system reduces administrative burden, minimizes errors, and enhances classroom efficiency. It also provides a scalable platform for future enhancements, such as integration with institutional management systems, cloud-based storage, or advanced analytics for student engagement. The Face Detection Attendance System represents a significant step toward modernizing educational administration through the application of computer vision and AI technologies.

Chapter 2

**LITERATURE REVIEW**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Face recognition is crucial in daily life in order to identify family, friends or someone we are familiar with. We might not perceive that several steps have actually taken in order to identify human faces. Human intelligence allows us to receive information and interpret the information in the recognition process. We receive information through the image projected into our eyes, by specifically retina in the form of light. Light is a form of electromagnetic waves which are radiated from a source onto an object and projected to human vision. Robinson-Riegler, G., & Robinson-Riegler, B. (2008) mentioned that after visual processing done by the human visual system, we actually classify shape, size, contour and the texture of the object in order to analyze the information. The analyzed information will be compared to other representations of objects or face that exist in our memory to recognize. In fact, it is a hard challenge to build an automated system to have the same capability as a human to recognize faces. However, we need large memory to recognize different faces, for example, in the Universities, there are a lot of students with different race and gender, it is impossible to remember every face of the individual without making mistakes. In order to overcome human limitations, computers with almost limitless memory, high processing speed and power are used in face recognition systems.

The human face is a unique representation of individual identity. Thus, face recognition is defined as a biometric method in which identification of an individual is performed by comparing real-time capture image with stored images in the database of that person (Margaret Rouse, 2012) .Nowadays, face recognition system is prevalent due to its simplicity and awesome performance. For instance, airport protection systems and FBI use face recognition for criminal investigations by tracking suspects, missing children and drug activities (Robert Silk, 2017). Apart from that, Facebook which is a popular social networking website implement face recognition to allow the users to tag their friends in the photo for entertainment purposes (Sidney Fussell, 2018). Furthermore, Intel Company allows the users to use face recognition to get access to their online account (Reichert, C., 2017). Apple allows the users to unlock their mobile phone, iPhone X by using face recognition (deAgonia, M., 2017). The work on face recognition began in 1960. Woody Bledsoe, Helen Chan Wolf and Charles Bisson had introduced a system which required the administrator to locate eyes, ears, nose and mouth from images. The distance and ratios between the located features and the common reference points are then calculated and compared. The studies are further enhanced by Goldstein, Harmon, and Lesk in 1970 by using other features such as hair colour and lip thickness to automate the recognition. In 1988, Kirby and Sirovich first suggested principle component analysis (PCA) to solve face recognition problem. Many studies on face recognition were then conducted continuously until today (Ashley DuVal, 2012). Viola-Jones algorithm which was introduced by P. Viola, M. J. Jones (2001) is the most popular algorithm to localize the face segment from static images or video frame. Basically the concept of Viola-Jones algorithm consists of four parts. The first part is known as Haar feature, second part is where integral image is created, followed by implementation of Adaboost on the third part and lastly cascading process.

Fig.2.1 show Viola-Jones algorithm analyses a given image using Haar features consisting of multiple rectangles (Mekha Joseph et al., 2016).

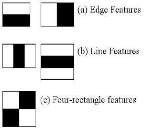
 

Figure 2.1: Haar Feature

In the fig 2.2 shows several types of Haar features. The features perform as window function mapping onto the image. A single value result, which representing each feature can be computed by subtracting the sum of the white rectangle(s) from the sum of the black rectangle(s).

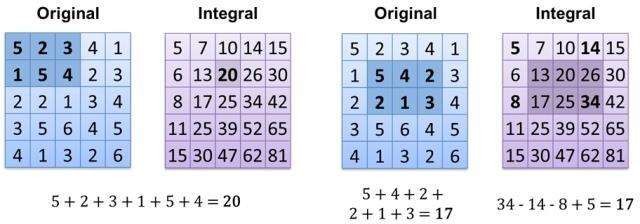


Figure 2.2: Integral of Image

The value of integrating image in a specific location is the sum of pixels on the left and the top of the respective location. In order to illustrate clearly, the value of the integral image at location 1 is the sum of the pixels in rectangle A. The values of integral image at the rest of the locations are cumulative. For instance, the value at location 2 is summation of A and B, (A + B), at location 3 is summation of A and C, (A + C), and at location 4 is summation of all the regions, (A + B + C + D). Therefore, the sum within the D region can be computed with only addition and subtraction of diagonal at location 4 + 1 − (2 + 3) to eliminate rectangles A, B and C.

**2.1 RECENT RESEARCH PAPERS REVIEWED**

1. “Automated Attendance System using Face Recognition” – IEEE (2020)  
   Proposed a system using LBPH with OpenCV; highlighted the balance between accuracy and performance.
2. “Smart Attendance using Real-Time Face Recognition” – IJERT (2021)  
   Focused on real-time implementation using Haar cascade and LBPH, and achieved over 90% recognition accuracy.
3. “Contactless Biometric Attendance System using Face Recognition and AI Chatbots” – Springer (2022)  
   Introduced chatbot integration for help and support in attendance systems—aligned with our Groq API approach.
4. “Comparative Study on Face Recognition Techniques” – IJARCET (2019)  
   Compared Eigenfaces, Fisherfaces, and LBPH, and concluded LBPH is ideal for lightweight real-time solutions.

**2.2 LIMITATIONS IN EXISTING LITERATURE**

Most systems didn’t consider chatbot integration for query management. Deep learning models like CNNs were accurate but too heavy for standard hardware. Few works offered modular and scalable architectures suitable for deployment in colleges. Real-time streaming combined with attendance tracking was rarely addressed in a web-based Flask framework.

**2.3 INNOVATION IN PROPOSED WORK**

1. Real-time face detection and attendance marking using Flask + OpenCV.
2. Integrated Groq LLaMA 3 API chatbot for AI-powered student interaction.
3. CSV-based attendance record with accurate timestamp logging.
4. Modular codebase allowing easy extension (e.g., SQLite or Firebase integration).
5. User-friendly web interface with live video stream and admin controls.

### ****2.4 ROLE OF ARTIFICIAL INTELLIGENCE IN ATTENDANCE SYSTEMS****

Artificial Intelligence (AI) has revolutionized the way biometric attendance systems are implemented. In traditional systems, rule-based logic was used, which lacked adaptability and contextual understanding. With AI integration—especially through machine learning and natural language processing—attendance systems can now offer smarter features like adaptive recognition (improving over time), voice or text-based chatbot assistance, and predictive analytics on student behavior. The integration of Groq’s LLaMA 3 model in this project exemplifies how AI can extend the utility of a face recognition system beyond attendance, making it an interactive and intelligent solution for academic environments.

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 INTRODUCTION**

System analysis is a crucial phase in the software development life cycle that involves a detailed study of the existing system or problem domain to define the system’s requirements and establish the necessary functionalities. In this project, the Face Detection Attendance System is designed to address the inefficiencies and limitations of traditional attendance methods by utilizing computer vision and machine learning technologies.

**3.2 EXISTING SYSTEM**

The conventional attendance system involves manual signing of registers or calling roll numbers, which is time-consuming and prone to errors such as proxy attendance and data manipulation. Biometric systems like fingerprint scanners improve security but require physical contact, making them less ideal in post-pandemic times. Furthermore, these systems lack integration with real-time monitoring and digital records, leading to inefficiencies in larger academic or professional environments.

**3.3 PROPOSED SYSTEM**

The proposed system leverages face detection and recognition to automate attendance marking. Using a webcam, the system captures live video, detects faces using Haar Cascade classifiers, and recognizes registered students through a trained LBPH (Local Binary Pattern Histogram) model. Once a face is identified, the system marks attendance with a timestamp and updates a structured attendance record. This system eliminates proxy attendance, reduces human effort, and enhances accuracy and security.

**3.4 FEASIBILITY STUDY**

* Technical Feasibility: The system is built using Python, OpenCV, Flask, and other widely-supported libraries, ensuring ease of development and compatibility across platforms.
* Operational Feasibility: The user interface is simple and intuitive, allowing faculty or administrators to manage training, view attendance, and operate the system with minimal training.
* Economic Feasibility: The system uses open-source tools and existing hardware (a webcam-enabled computer), minimizing implementation costs.

**3.5 REQUIREMENTS ANALYSIS**

* **Functional Requirements**:
  1. User can capture and train student images.
  2. System should detect and recognize faces from live video.
  3. Attendance should be recorded with date and time.
  4. Admin can view, update, or export attendance logs.
* **Non-Functional Requirements**:
  1. The system must process frames in near real-time.
  2. It should ensure data privacy and store information securely.
  3. System uptime and reliability must be high during operational hours.

**3.6 SYSTEM CONSTRAINTS**

Lighting conditions and camera quality can affect accuracy.Recognition rate may vary depending on training data quality and face orientation.The system must run on a device with sufficient processing power and camera access.

**3.7 ASSUMPTIONS AND DEPENDENCIES**

All users (students) will be registered with clear facial images before attendance tracking.The webcam and system software will function reliably during use .Internet connection is not mandatory unless cloud integration is required.

Chapter 4

**SYSTEM TECHNOLOGIES**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**4.1 OPENCV**

OpenCV is an open-source computer vision library that provides various functions for real-time image processing. It includes the Haar Cascade classifier, a popular algorithm for face detection.

**4.1.1. KEY FEATURES:**

* Image Processing: Basic operations such as resizing, blurring, edge detection, and image thresholding.
* Video Processing: Real-time video capture, frame extraction, and display.
* Face Detection: OpenCV includes several pre-trained face detection models like Haar Cascades and LBP (Local Binary Patterns).
  1. **WHY OPENCV FOR FACE DETECTION?**
* Haar Cascade Classifier: OpenCV includes Haar cascade classifiers for face detection, which is a lightweight and efficient method suitable for real-time applications.
* Fast and Lightweight: OpenCV is optimized for performance and can handle real-time video processing on standard hardware.

**4.3 HAAR CASCADE CLASSIFIER:**

A machine learning-based approach used by OpenCV for object detection. It uses a series of positive and negative images to train a classifier that can detect objects in new images. OpenCV provides pre-trained Haar Cascades for face detection.The Haar Cascade Classifier is one of the most commonly used methods for object detection in computer vision, particularly for face detection. It is based on the concept of Haar-like features, which are used to represent the object to be detected. The cascade classifier was popularized by Paul Viola and Michael Jones in their landmark paper, *"Rapid Object Detection using a Boosted Cascade of Simple Features"* (2001), and has since become a widely-used approach in real-time applications due to its balance between accuracy and speed. In this section, we will explore the underlying principles of the Haar Cascade Classifier, how it works, and its application in face detection.

**4.3.1 KEY COMPONENTS OF HAAR CASCADE CLASSIFIER**

* Haar-like Features: These are simple rectangular features (based on the Haar wavelet transform) that capture the differences in intensity between different regions of an image.
* Integral Image: A technique used to quickly calculate the sum of pixel intensities in a given region of an image.
* AdaBoost (Adaptive Boosting): A machine learning technique that combines weak classifiers to create a strong classifier. In the context of Haar Cascades, this is used to combine multiple Haar-like features to detect an object.
* Cascade of Classifiers: A series of increasingly complex classifiers applied in sequence to quickly discard non-object regions and focus computational resources on the most promising areas

**4.3.2. HAAR-LIKE FEATURES**

Haar-like features are simple rectangular features used to represent image characteristics. They are similar to the Haar wavelet transform and are calculated by subtracting the sum of pixel values in one rectangle from the sum of pixel values in another rectangle.

Types of Haar-like Features

There are several types of Haar-like features, including:

* Edge Features: Captures intensity differences between adjacent regions (vertical or horizontal edges).
* Line Features: Measures intensity differences between two horizontal or vertical lines.
* Four-Rectangle Features: Captures intensity differences in four adjacent rectangular regions (used for more complex patterns).

These features are calculated for every possible location and scale within an image to detect patterns

**4.4 NUMPY**

NumPy (short for Numerical Python) is a powerful library in Python that provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays. It is one of the core libraries in the Python ecosystem for scientific computing and data analysis. NumPy is especially important in fields like data science, machine learning, computer vision, and engineering, where large datasets are manipulated and processed efficiently. A core library in Python used for handling arrays and performing matrix operations. NumPy will help in handling image data as arrays.

**4.4.1. KEY FEATURES OF NUMPY**

* Multi-dimensional Arrays (ndarray): The core data structure in NumPy is the ndarray (N-dimensional array), which allows for the storage and manipulation of homogeneous data (all elements must have the same data type).

These arrays are more efficient than Python’s built-in lists for numerical operations, as they are implemented in C and offer more compact storage.

* Mathematical Operations: NumPy provides a wide range of mathematical functions that can be applied to arrays. These include basic operations like addition, subtraction, multiplication, division, and more complex ones like matrix multiplication, linear algebra, and statistical operations.

Vectorized operations: Unlike Python lists, NumPy arrays support element-wise operations without requiring explicit loops. This allows for fast computation.

* Broadcasting: Broadcasting is a powerful feature of NumPy that allows operations to be performed on arrays of different shapes. NumPy automatically adjusts the shapes of arrays to make them compatible for element-wise operations.
* Linear Algebra: NumPy provides functions for performing operations like matrix multiplication, determinant calculation, eigenvalues, and solving systems of linear equations.
* Random Number Generation: NumPy includes the numpy.random module, which allows the generation of random numbers from various probability distributions.
* Integration with Other Libraries: NumPy serves as the foundation for many other libraries in the Python ecosystem, such as Pandas (for data analysis), Matplotlib (for data visualization), and SciPy (for scientific and technical computing). Other machine learning libraries like TensorFlow and PyTorch are also built on top of NumPy arrays.

**4.5 MATPLOTLIB**

Matplotlib is a comprehensive data visualization library in Python used for creating static, animated, and interactive plots. It is built on NumPy and is widely used for visualizing trends, distributions, and image data. Although not mandatory for the face detection system to function, Matplotlib plays a supportive role in understanding patterns in attendance records and displaying image-related data during development or debugging. Its ability to plot both numerical and image data makes it a versatile tool in computer vision projects.

**4.5.1 KEY FEATURES OF MATPLOTLIB**

• Data Visualization:

Supports line graphs, bar charts, scatter plots, histograms, and more for interpreting numeric data.  
• Image Display:

Allows rendering of image data (e.g., training samples, detected faces) using imshow() for debugging and verification.  
• Customization:

Offers control over plot styling, labels, legends, colors, grid lines, and figure size for presentation-ready outputs.  
• Attendance Trend Visualization:

Can be used to plot attendance trends over time (e.g., number of students present per day), helping in reporting and analytics.  
• Real-time Plotting Support:

Useful during live attendance capture or training sessions to visualize recognition success rates or model training feedback in real time.

.

Chapter 5

**REQUIREMENTS**

**5.1 OVERVIEW**

System requirements define the necessary hardware and software components required to design, develop, and execute the Face Detection Attendance System efficiently. This project is a Flask-based web application that integrates computer vision using OpenCV, face recognition using the LBPH algorithm, and also incorporates a chatbot feature using Groq’s API for LLaMA 3. The requirements are categorized into hardware and software components for development and deployment environments.

**5.2 SOFTWARE REQUIREMENTS**

* Operating System: Windows 10/11, macOS, or Linux (Ubuntu recommended for Python development).
* Python Version: Python 3.x (specific version used during development).
* Required Libraries:
  1. OpenCV (for face detection, video processing)
  2. dlib (for face recognition, if used)
  3. face\_recognition (if used, builds on dlib)
  4. NumPy (for numerical operations)
  5. Pillow (for image manipulation)
  6. Any GUI framework libraries (e.g., PyQt5, Tkinter, Kivy).

**5.3 HARDWARE REQUIREMENTS**

1. Camera: High-resolution webcam (e.g., 720p or 1080p) for live video capture.
2. Processor: Quad-core CPU or higher (e.g., Intel Core i5/i7, AMD Ryzen 5/7) for efficient face detection and recognition.
3. RAM: 8GB or more for smooth operation, especially with larger face databases.
4. 4.Storage: SSD recommended for faster data access and system responsiveness

**5.4 FUNCTIONAL REQUIREMENTS**

Student Management: Add, train, and manage student face data. Face Detection: Detect faces using webcam feed in real-time. Face Recognition: Match detected faces with trained data using LBPH algorithm. Attendance Marking: Automatically record attendance with name, date, and time. Attendance Sheet: Save attendance logs in CSV format. Chatbot Interaction: Enable users to ask queries via LLaMA 3-based chatbot. Web Interface: Simple and responsive UI to access system functions via browser. Live Video Feed: Real-time video stream for visual attendance confirmation.

**5.5 NON-FUNCTIONAL REQUIREMENTS**

Performance: The system must respond to face detection and recognition in real-time (within 1–2 seconds). Scalability: Able to support an increasing number of student face records with minimal degradation. Security: Only authorized users can access the admin panel and chatbot services. Usability: Clean and intuitive interface for faculty and students. Portability: The system should work on any machine with Python and required libraries. Reliability: Should function continuously without failure during lecture hours.

**5.6 ASSUMPTIONS**

All users will be registered with at least 5 facial images for training accuracy. Webcam and system permissions will be enabled during usage. Groq API key and internet will be available for chatbot features. Face recognition is conducted under moderate lighting conditions for optimal accuracy.

**5.7 CONSTRAINTS**

System performance may vary based on lighting, camera quality, and processing power. Recognition accuracy decreases with poor training images or rapid face movements. Chatbot features depend on stable internet and API access from Groq.

Chapter 6

**SYSTEM DESIGN**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**6.1 INTRODUCTION TO SYSTEM DESIGN**

The System Design chapter serves as a comprehensive overview of how the face detection system is architected, from high-level concepts to the breakdown of individual components. The primary aim is to ensure clarity on how different parts of the system interact, how face detection will be achieved, and the design decisions taken to meet the project’s goals. This section introduces the fundamental design choices, objectives, and the flow of data through the system, offering readers a structured look at the entire face detection pipeline, starting from image acquisition to face detection and output generation. Overview of the Design Approach The design approach for the face detection system is based on leveraging computer vision and machine learning techniques to detect human faces in images or video streams. Python, a versatile programming language, has been chosen due to its extensive libraries such as OpenCV and Dlib, which provide state-of-the-art algorithms for real-time face detection. The design follows a modular structure, where each component plays a specific role in achieving the desired output. This approach ensures flexibility, maintainability, and scalability. The system is intended to be easily extendable to incorporate additional features like facial recognition, emotion detection, or multi-face detection.

**6.2 KEY STEPS IN THE DESIGN APPROACH**

The system design provides a visual representation of the sequential steps and processes in the facial recognition system. It highlights how data flows through various stages, ensuring clarity and efficiency. Key elements of the flowchart include: Initialization: The process begins with loading essential libraries such as OpenCV and Flask, along with pre-trained models. Image Capture: The system captures real-time images or video feeds from the camera for further processing. Preprocessing: Captured images are normalized for lighting and resized for consistency. Face Detection: OpenCV is used to identify faces within the preprocessed images.

**6.2.1 IMAGE INPUT AND PREPROCESSING**

Input data is captured through either images or video. The system preprocesses this input to prepare it for detection (e.g., resizing, grayscale conversion). Face Detection Algorithm: The core component, where face detection occurs. The system uses machine learning or traditional computer vision algorithms, such as Haar Cascade or HOG + SVM, to identify faces.

**6.2.2 POST-PROCESSING AND OUTPUT**

Detected faces are highlighted with bounding boxes, and additional information (if applicable) such as confidence scores or labels is displayed. User Interaction: In case of real-time applications, the system displays detection results on the screen or stores the output (e.g., saving images with detected faces).

**6.3 DATA FLOW DIAGRAM**

Fig. 6.3.1 ,fig.6.3.2 shows a Data Flow Diagram (DFD) is a great way to visually represent the flow of data through your face detection system. It helps to illustrate how input data is processed by the system to produce the desired output. Here's an outline of a Level 0 DFD and Level 1 DFD for your face detection system:

* Level 0: DFD Diagram This level provides a very high-level overview of the system, showing the primary inputs, processes, and outputs.

|  |
| --- |
| [User] (Face Detection System) [Detected Faces (Output)]        [Camera/Video Source] |

Fig .6.3.1 Level 0 : Detailed DFD

* Level 1: Detailed DFD This level breaks down the "Face Detection System" process into smaller sub-processes, giving more detail on how the data flows and is processed within the system. This represents the overarching system or main class that orchestrates the entire face detection process. The FaceDetector takes the preprocessed image (grayscale, resized) and applies its face detection algorithm to identify the locations of faces within the image. The output of this step would typically be a list of coordinates or regions of interest (ROIs) representing the detected faces.

[Camera/Video Source] [Face Detection Model]

[User] (Image Acquisition) [Image Store] (Preprocessing)

[Faces Detected] (Face Detection Algorithm) [Preprocessed Image]

(Output Generation) (Post-Processing) [Processed Output]

Fig. 6.3.2 Level 1: Detailed DFD

**6.4. SYSTEM FLOW DIAGRAM**

Here's 6.4 fig shows a overview of the system architecture flow for real-time face detection This diagram illustrates a real-time face detection system, starting with Video Capture (camera/video). The captured frames undergo Preprocessing (frame conversion, e.g., grayscale, resize) before being fed into a Face Detection Model that identifies faces and provides bounding box coordinates. Finally, Postprocessing adds annotations like drawing bounding boxes and performing facial landmark detection, and applies filtering, with the ultimate Display/Save Results shown to the user or stored

Video Capture Preprocessing Face Detection

(Camera/Video) (Frame Conversion) (Face Detection Model)

Postprocessing (Draw Bounding Boxes, Display/Save Results

(Face annotations, Facial Landmark Detection)

Filtering)

Fig. 6.4 : SYSTEM FLOW DIAGRAM

**6.5 UML CLASS DIAGRAM**

We can break down the classes involved as follows: Face Detection System: A class that serves as the main interface for face detection. Methods: load image(), process image(), detect faces(), draw faces(). Image Processor: A utility class to handle image processing tasks like converting to grayscale or resizing. Methods: convert to grayscale(), resize image(), blur image(). Face Detector: This class will contain the logic for detecting faces using a detection model like Haar cascades. Methods: load cascade model(), detect(). Bounding Box Drawer: A utility class for drawing bounding boxes around detected faces. Methods: draw bounding boxes().

### Here's fig. 6.5 shows how you can represent the system with a UML diagramFace Detection System: The main class that integrates all components and uses them in the face detection pipeline. It calls methods from the other classes to process the image and detect faces. Image Processor: Handles image preprocessing tasks such as converting to grayscale, resizing, or blurring. Face Detector: Implements the logic for detecting faces using a model (like Haar cascades or HOG-based detectors). Bounding Box Drawer: Responsible for highlighting the detected faces in the image by drawing bounding boxes.

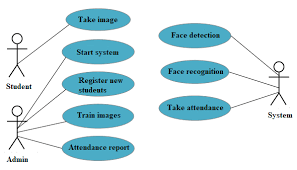


Fig 6.5 :UML DIAGRAM

### 6.6 SEQUENCE DIAGRAM (FOR DETECTING FACES)

The 6.6 fig :Sequence Diagram (for detecting faces) shows the sequence of actions for detecting faces from an image. This diagram outlines a Face Detection System workflow. It begins by loading an image, which is then preprocessed by an ImageProcessor (converted to grayscale, resized). The preprocessed image is passed to a FaceDetector to identify faces. Finally, a BoundingBoxDrawer visualizes the detected faces by drawing bounding boxes on the image.

Step 1: FaceDetectionSystem loads the input image.

Step 2-3: ImageProcessor processes the image (converts to grayscale, resizes, etc.).

Step 4: FaceDetector detects faces in the processed image.

Step 5: BoundingBoxDrawer draws bounding boxes around the detected faces.

Step 6: The processed image with detected faces is outputted.

FaceDetectionSystem

1. load\_image(image)

ImageProcessor

|

| 2. convert\_to\_grayscale()

| 3. resize\_image()

FaceDetector

|

| 4. detect\_faces()

BoundingBoxDrawer

|

| 5. draw\_bounding\_boxes()

Output

Fig. 6.6 :Sequence Diagram (for detecting faces)

**6.7 SYSTEM FLOWCHART**

The system flowchart fig 6.7 provides a visual representation of the sequential steps and processes in the facial recognition system. It highlights how data flows through various stages, ensuring clarity and efficiency. Key elements of the flowchart include: Initialization: The process begins with loading essential libraries such as OpenCV and Flask, along with pre-trained models.

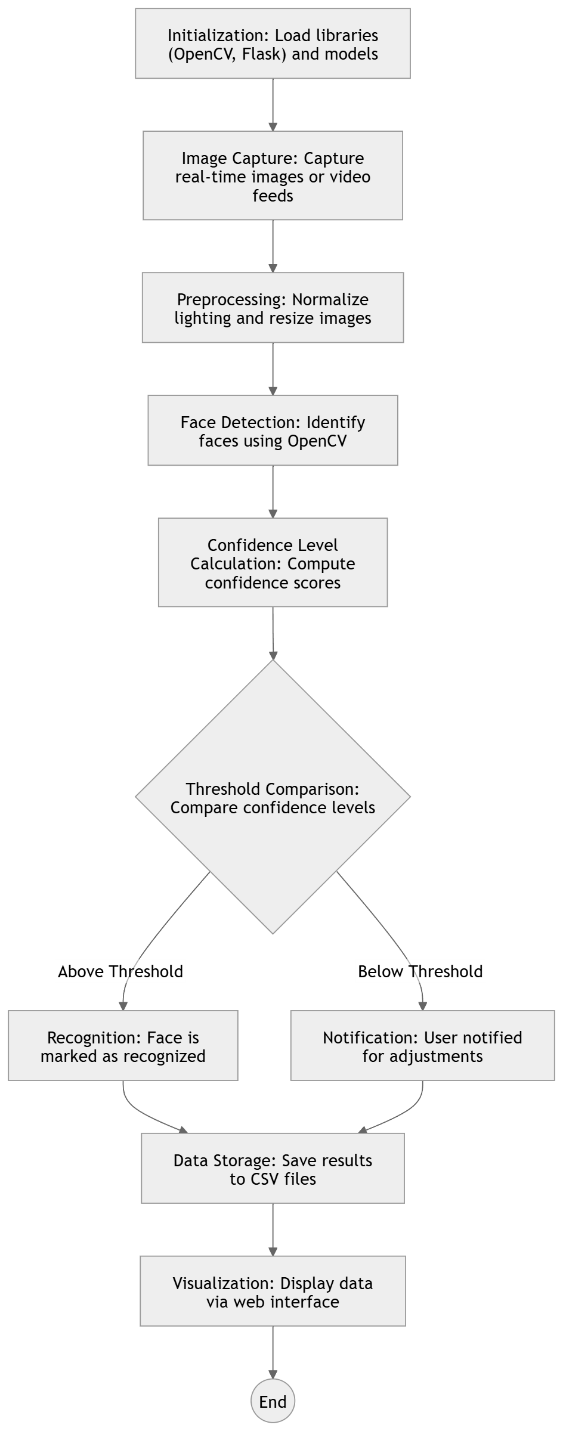


Fig. 6.7 FlowChart

Chapter 7

**IMPLEMENTATION**

**7.1 HOME PAGE**

Screen short 7.1 showsthe homepage has a clean, modern design with a clear hierarchy of information. The prominent title, descriptive tagline, and call-to-action button guide the user. The visual demonstration of face detection through the overlay effectively communicates the application's functionality at a glance. The navigation bar suggests a comprehensive website with various features beyond just the detection itself, such as enrollment, camera access, and a gallery**.**

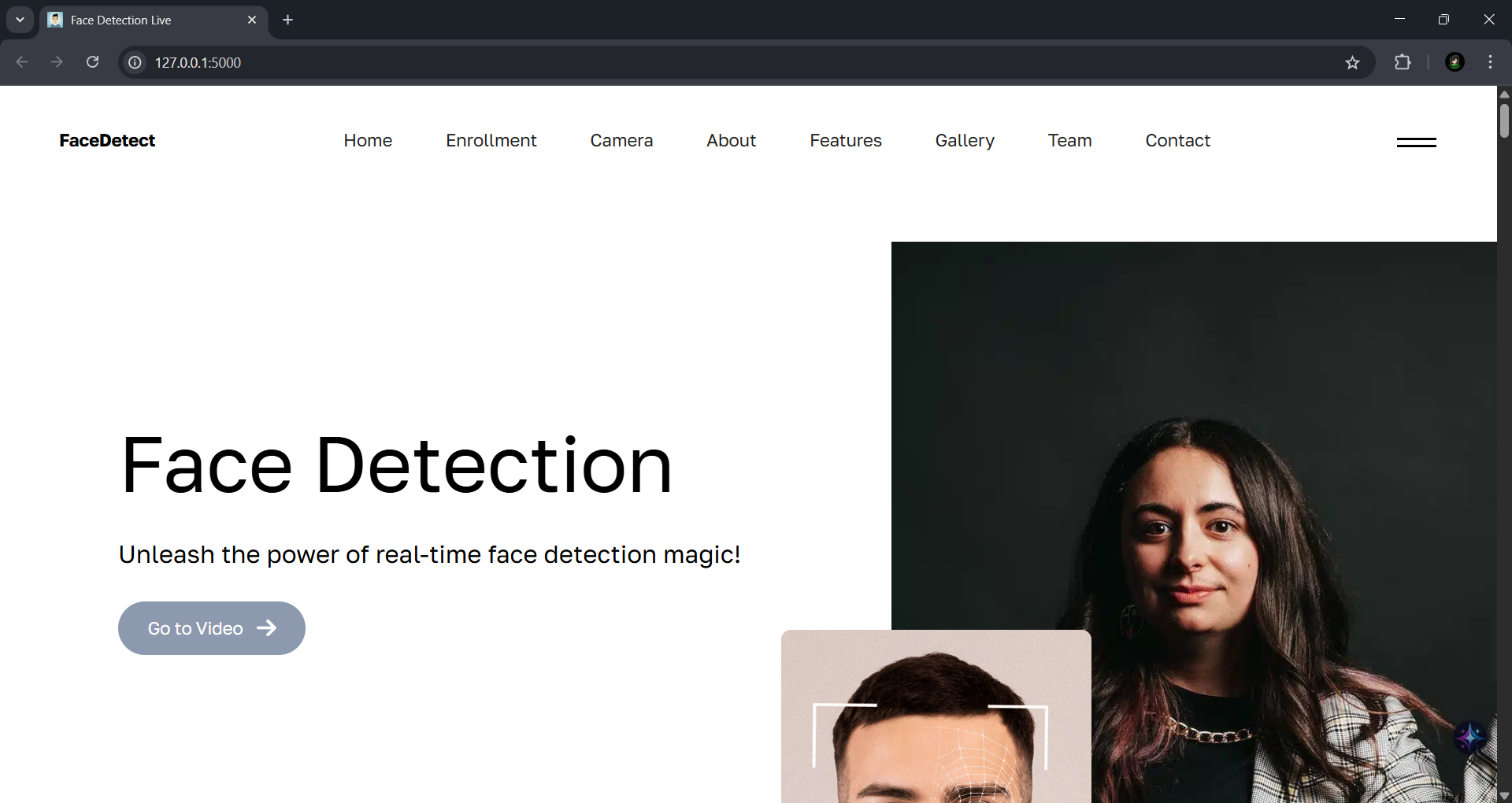


fig 7.1home page

**7.2 ENROLLMENT PAGE**

Fig. 7.2 show the page is designed for attendance management, offering two distinct methods: automatic attendance via live face detection and manual attendance entry. The layout is clean and straightforward, using distinct white cards for each section to improve readability. The green buttons indicate primary actions. This setup is typical for an attendance system where automated processes are preferred but manual override or entry is also necessary

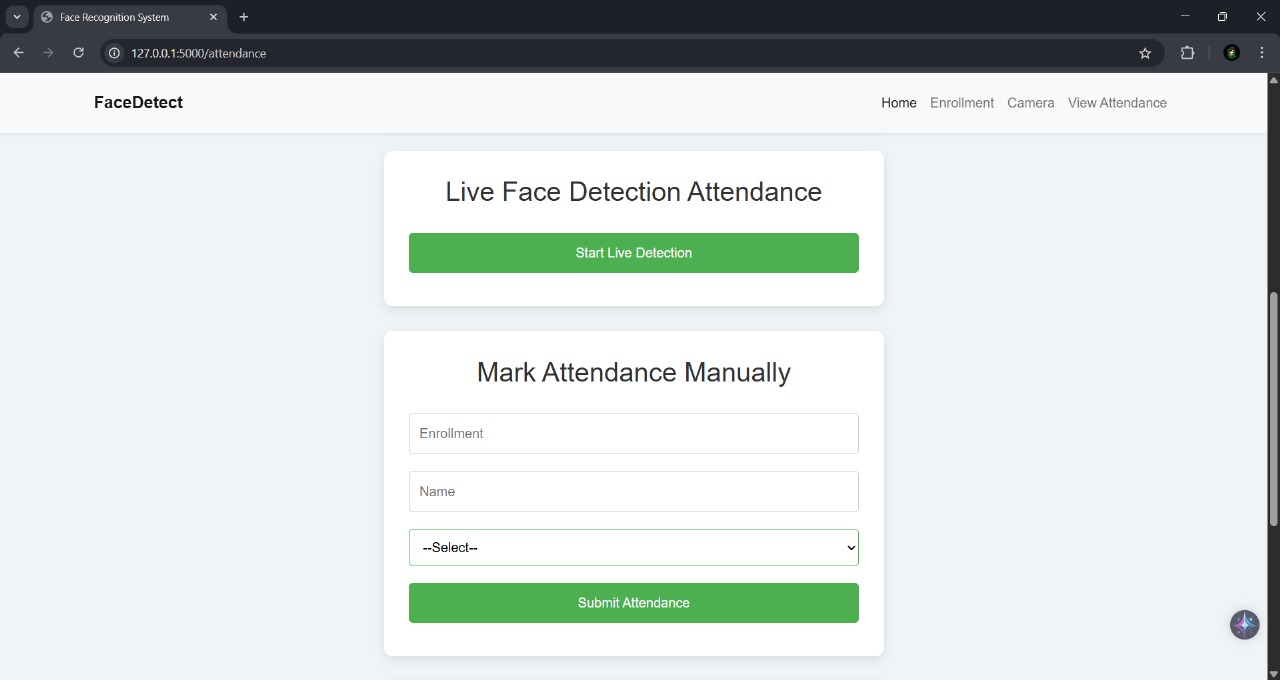


Fig. 7.2 Enrollment page

**7.3 LIVE FACE DECTATION ATTENDANCE PAGE**

Fig. 7.3 Live Face Dectation Attendancepage serves as the operational interface for the face recognition attendance system. It clearly shows the live video stream, control buttons for managing the video, and real-time overlays indicating detected and recognized faces, along with their confidence scores. The distinct coloring of bounding boxes (green for recognized, red for unknown) provides immediate visual feedback on the recognition status, making it intuitive for monitoring attendance. The timestamp further enhances its utility for attendance tracking.

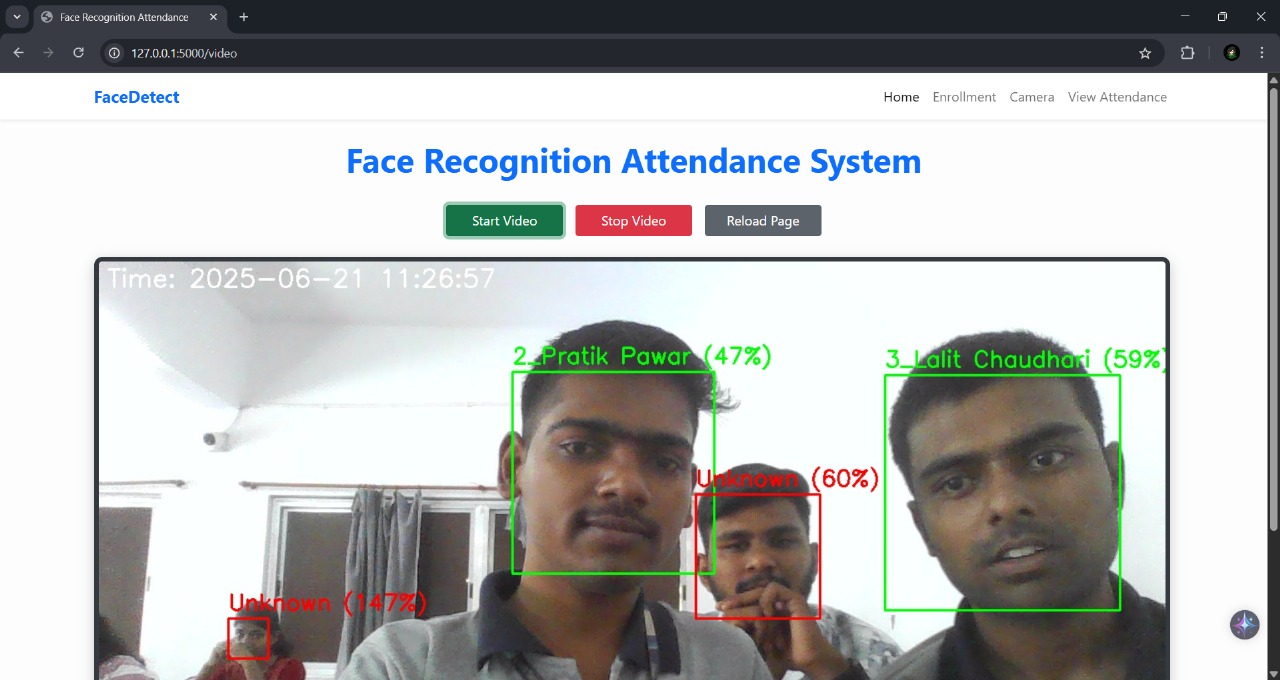


Fig. 7.3 Live Face Dectation Attendancepage

**7.4 MANUALLY ATTENDANCE PAGE**

Fig. 7.4 Manually AttendanceThis web page is designed for managing the enrollment and training aspects of a face recognition system. It allows for the capture of new individual's images along with their details (Enrollment Number, Student Name) and provides a dedicated function to train or retrain the underlying face recognition model using the collected data. The prominent success message in the pop-up confirms that a training operation has just been completed, signifying that the system is ready to use the newly learned facial data for recognition tasks

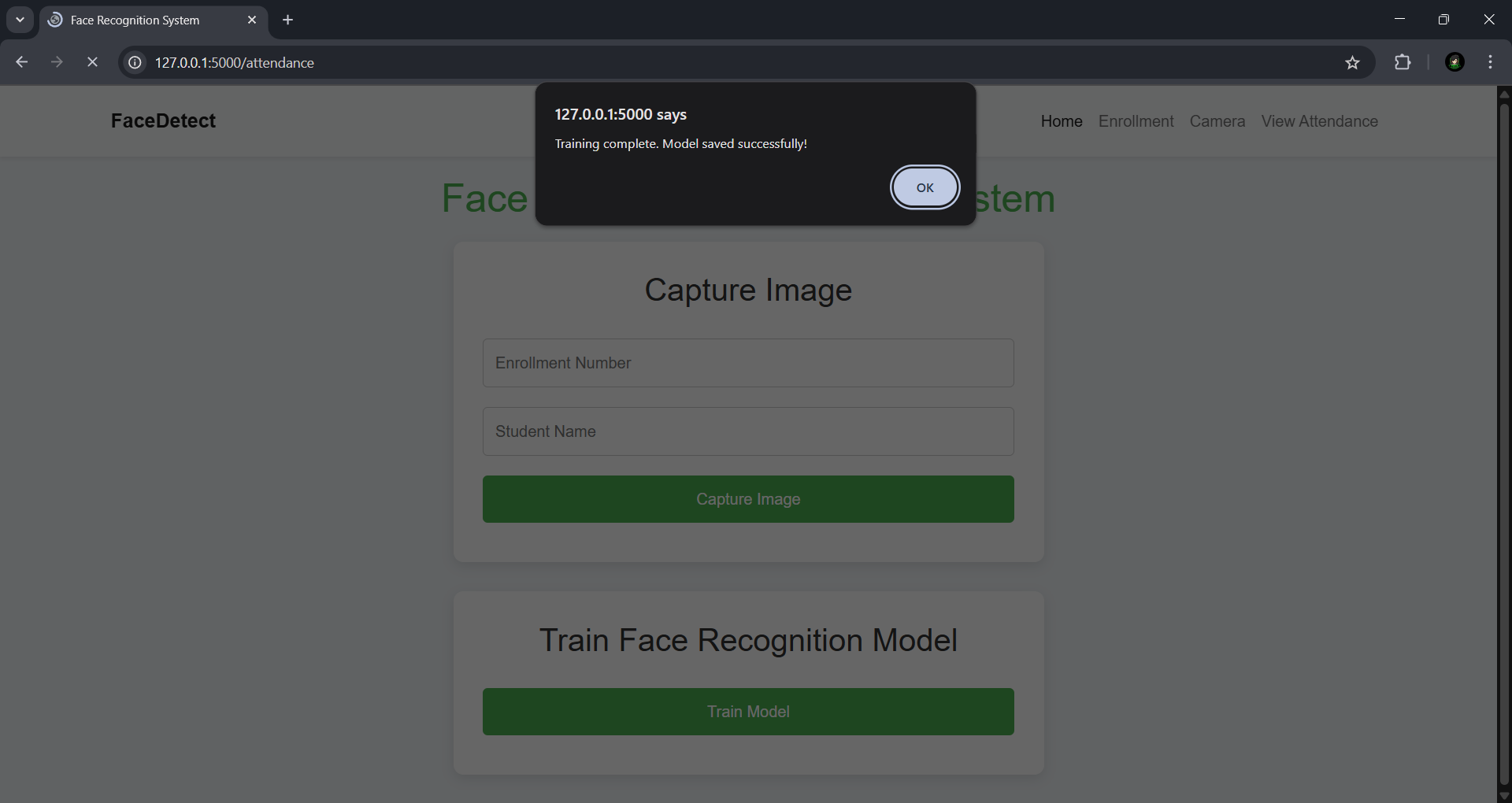
****

Fig. 7.4 Manually Attendance

**7.5 VIEW ATTENDANCE PAGE**

Fig.7.5 View Attendance web page provides a clear and organized view of attendance records captured by the face recognition system. The table format makes it easy to read and review who was present, their enrollment details, and the exact time of their attendance. The inclusion of a "Delete Attendance" option in the navigation suggests that these records can also be managed (i.e., deleted) by an authorized user. The "Back to Home" button ensures easy navigation within the application.

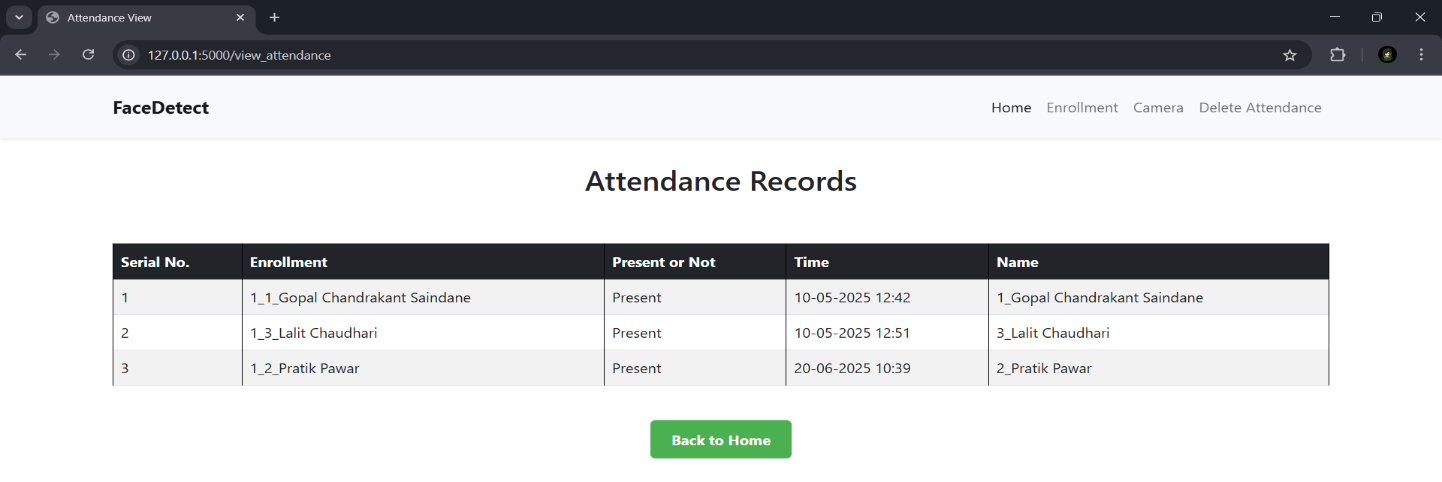


Fig.7.5 View Attendance web page

**7.6 DELETE ATTENDANCE PAGE**

Fig.7.6 Delete Attendance web page serves as an administrative interface for managing enrolled students within the face recognition system. It provides a clear list of all currently enrolled individuals with their key details. The "Delete" action button for each student is a significant feature, allowing for the removal of student data, which is essential for data management and privacy within such a system. The page is clean, functional, and user-friendly for managing the student roster.

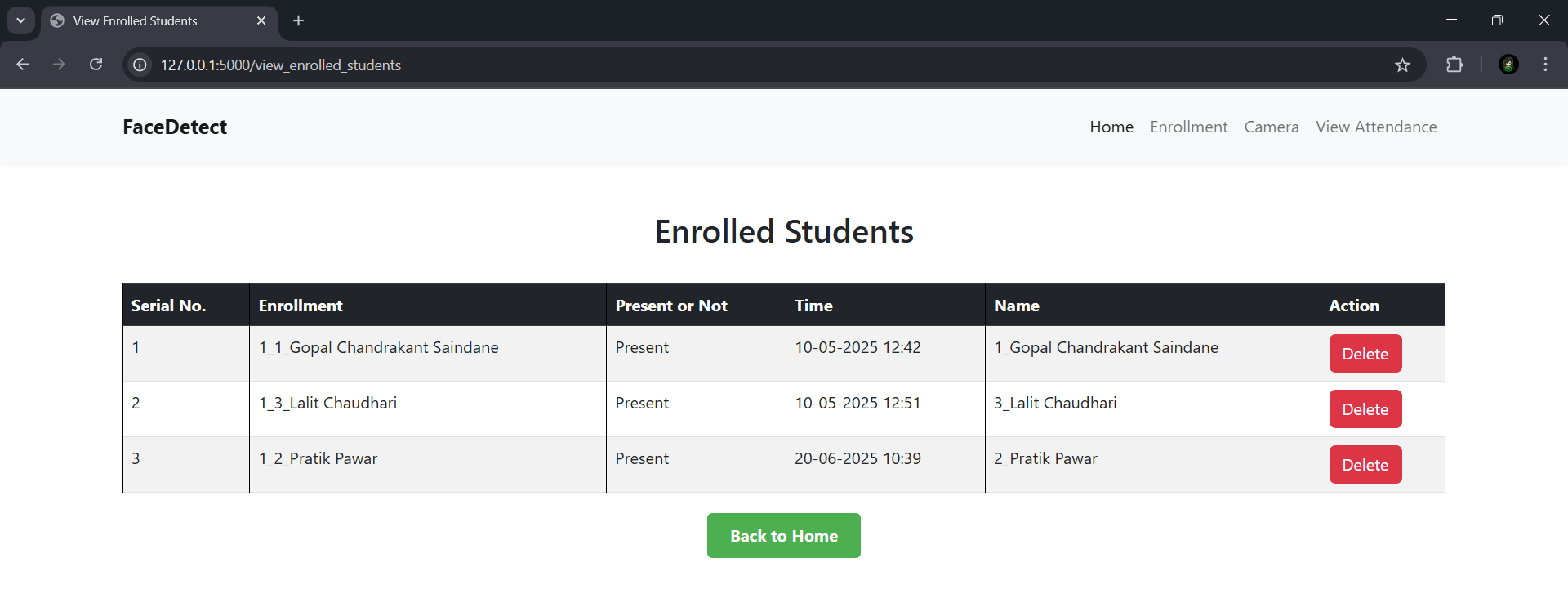


Fig.7.6 Delete Attendance web page

Chapter 8

**TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**8.1 TYPES OF TESTS**

A well-defined test plan is essential for systematic and effective testing. Our strategy will encompass several levels of testing, moving from individual components to the integrated system.

**8.1.1 UNIT TESTING**

Objective: To test individual modules or components of the system in isolation to ensure they function correctly according to their design specifications.

**8.1.2 SYSTEM TESTING**

Objective: To evaluate the complete, integrated system to verify that it meets all specified functional and non-functional requirements from an end-user perspective.

* 1. **MODULE TESTING**

**8.2.1FACE DETECTION MODULE:**

Testing the accuracy and speed of the face detection algorithm (e.g., Haar cascades, dlib, OpenCV's DNN module).

* + Test Cases:
    1. Detecting faces in various lighting conditions (bright, dim, natural, artificial).
    2. Detecting faces at different angles and orientations (frontal, profile, tilted).
    3. Detecting multiple faces in a single frame.
    4. Detecting faces with partial obstructions (glasses, masks – if applicable to your design, otherwise this would be a negative test).
    5. Detecting faces of different ages, genders, and ethnicities.
    6. Negative test cases: Images without faces, images with non-human faces (e.g., statues, drawings).

**8.2.2 DATABASE INTERACTION MODULE:**

Testing the functions related to storing and retrieving user data, attendance records, and face encodings.

* + Test Cases*:*
    1. Successful user registration and data storage.
    2. Accurate retrieval of user profiles.
    3. Correct recording and retrieval of attendance timestamps.
    4. Handling of duplicate entries or missing data.
    5. **TIME AND DATE MODULE:**

Verifying the accuracy of timestamping attendance records.

* + Test Cases:
    1. Correctly capturing current date and time.
    2. Handling of different time zones (if applicable).

**8.3 INTEGRATION TESTING**

Objective: To test the interactions between different modules of the system to ensure they work together seamlessly as a cohesive unit.

Scope:

**8.3.1 ENROLLMENT WORKFLOW:**

Testing the process from capturing a user's face to storing their data and face encodings in the database.

* + Test Cases:
    1. Successful end-to-end enrollment of a new user.
    2. Error handling during enrollment (e.g., poor image quality, existing user ID).
    3. **ATTENDANCE MARKING WORKFLOW:**

Testing the complete process from live video feed to face detection, recognition, and attendance recording.

* + Test Cases:
    1. A recognized user successfully marking attendance.
    2. An unrecognized user being prompted to enroll or denied entry.
    3. Multiple users being processed consecutively.
    4. System behavior when a recognized user attempts to mark attendance multiple times within a short period (e.g., preventing duplicate entries).
    5. System responsiveness to a continuous stream of faces.

**8.4 FUNCTIONAL TESTING:**

* 1. User Registration: Ensuring new users can be registered accurately.
  2. Attendance Marking: Verifying that users can successfully mark their attendance through face recognition.
  3. Administrator Features: Testing admin functionalities such as adding/removing users, viewing attendance logs, generating reports.
  4. Error Handling: Testing how the system responds to invalid inputs, network issues, or other unexpected events.
  5. Security Testing:
     1. Liveness Detection (if implemented): Testing the system's ability to differentiate between a live person and a photo/video (e.g., using anti-spoofing techniques).
     2. Data Security: Verifying that sensitive user data (face encodings, personal information) is stored securely (e.g., encrypted).
     3. Access Control: Ensuring only authorized personnel can access administrator functionalities.

**8.5 NON-FUNCTIONAL TESTING**:

* + Performance Testing:
    1. Response Time: Measuring the time taken for face detection and recognition.
    2. Throughput: Assessing the number of users the system can process per minute.
    3. Scalability: Evaluating how the system performs with an increasing number of registered users and concurrent attendance markings.
    4. Resource Utilization: Monitoring CPU, memory, and disk usage during operation.
  + Usability Testing:
    1. User Interface (UI) and User Experience (UX): Assessing the ease of use for both users marking attendance and administrators managing the system.
    2. Clarity of Feedback: Ensuring clear messages are provided to the user (e.g., "Attendance Recorded," "Face Not Recognized").
  + Reliability Testing:
    1. Stability: Running the system for extended periods to detect crashes or memory leaks.
    2. Recovery: Testing the system's ability to recover gracefully from failures (e.g., power outage, application restart).
* Tools/Methods:

1. User Acceptance Testing (UAT) with actual end-users.
2. Performance testing tools (e.g., custom Python scripts for load simulation, system monitoring tools).
3. Security auditing tools and penetration testing (basic level for a student project).

Chapter 9

**ADVANTAGES , DISADVANTAGES AND APPLICATION**

**9.1ADVANTAGES OF LIVE ATTENDANCE USING FACE DETECTION**

This is a major advantage, especially in a post-pandemic world. Users don't need to touch shared surfaces (like fingerprint scanners or keypads), reducing the spread of germs.

1. Increased Accuracy and Reduced Fraud (Buddy Punching):

Face detection makes "buddy punching" (where one person clocks in for another) virtually impossible. Since each face is unique, it ensures that only the registered individual can mark their attendance. This leads to more accurate attendance records.

1. Efficiency and Time-Saving:

Automated Process: Eliminates manual record-keeping, saving significant administrative time and effort. Faster Clock-in/out: Individuals can clock in/out quickly by simply looking at the camera, reducing queues, especially in large organizations or educational institutions.

1. Enhanced Security and Access Control:

Beyond just attendance, these systems can be integrated with access control to ensure only authorized personnel enter specific areas.

1. Scalability and Adaptability:

Can efficiently handle varying volumes of attendance data and adapt to different environments (e.g., offices, schools, construction sites).

1. Real-time Monitoring and Reporting:

Provides immediate updates on who is present, allowing for real-time tracking and the generation of quick, accurate attendance reports for payroll or other administrative tasks.

1. Cost-Effectiveness (in some cases):

While initial setup might have hardware costs, the long-term savings from reduced manual effort, increased accuracy, and potentially using existing camera infrastructure can make it cost-effective. Python's open-source libraries (like OpenCV) further reduce software costs.

1. User-Friendly Interface:

For the end-user, it's generally a very intuitive and simple process – just stand in front of the camera.

1. Integration Potential:

Can be easily integrated with other systems like HR management systems (HRMS) or payroll software for a seamless workflow.

1. Reduced Paperwork:

Eliminates the need for physical attendance sheets or cards.

**9.2 DISADVANTAGES OF LIVE ATTENDANCE USING FACE DETECTION**

1. Privacy Concerns:

This is one of the most significant drawbacks. Collecting and storing biometric data (facial images) raises serious privacy issues. There are concerns about how this data is stored, secured, and used, and the potential for misuse or breaches.

1. Spoofing Vulnerability:

Sophisticated spoofing attacks using high-quality photos, videos, or even 3D masks can potentially trick the system if anti-spoofing measures are not robustly implemented.

1. Technical Expertise for Development and Maintenance:

Developing and maintaining such a system in Python requires expertise in computer vision, machine learning, and software development. Debugging issues related to model performance, hardware integration, or real-time processing can be complex.

**9.3 APPLICATIONS**

1. Educational Institution**s :**

Used in schools, colleges, and universities to automatically record student attendance. Reduces administrative workload for faculty and eliminates proxy attendance. Real-time logs help in generating monthly attendance reports and performance tracking.

1. Corporate Offices :

Can replace biometric fingerprint scanners in offices for employee attendance. Supports hygienic, touch-free logging especially relevant in the post-pandemic era. Can be linked to HR systems for automated payroll and attendance tracking.

1. Laboratories and Research Centers :

Tracks researcher or student entry and exit times automatically. Provides logs that are useful for lab access monitoring and project hour tracking.

1. Coaching Centers and Training Institutes:

Easily deployed in private institutes with minimal setup. Helps maintain punctuality and prevents misuse of attendance logs.

1. Government and Secure Facilities :

Can be customized to monitor employee attendance in sensitive or restricted-access government offices. Integrated AI chatbot offers real-time interaction, enhancing usability and support.

1. Gyms and Fitness Centers:

Member Check-in: Automate member entry, ensuring only registered members can access the facilities, streamlining the check-in process and reducing administrative burden.

1. Public Transportation Hubs (e.g., Bus Stations, Metro Stations) :

Staff Attendance: Track the presence of operational staff, ticket collectors, security personnel, etc., at various points within the transport network. (Not for passenger tracking due to privacy).

1. Event Security and Access Control:

Staff and Volunteer Attendance: Manage the check-in and check-out of event staff and volunteers at large-scale events (festivals, conferences, sports events). Restricted Zone Access: Control access to backstage areas, VIP lounges, or press rooms.

1. Residential Communities (Gated Communities, Apartment Complexes):

Resident Entry/Exit: For residents who opt-in, it can provide a quick, hands-free entry/exit experience at gates or building entrances, enhancing convenience and security.Security Guard Patrol Tracking: Ensure security personnel are patrolling specific areas by using face recognition at checkpoints.

1. Childcare Centers and Daycares :

Child Drop-off/Pick-up Verification: Ensure that only authorized parents or guardians are picking up children, adding an extra layer of security. Staff Attendance: Track attendance of caregivers and teachers.

1. Librarian/Staff Attendance:

Manage the presence of library staff. Patron Check-in (for registered users): For academic or specialized libraries, registered users could use it for faster check-in, tracking library usage patterns.

1. Museums and Art Galleries :

Staff Attendance: For employees working at the museum/gallery. Special Exhibitions Access: Control access to exclusive exhibits for ticket holders or VIPs (with prior enrollment and consent).

1. Warehousingand Logistics:

Worker Attendance: Track the presence of warehouse staff, forklift operators, and loading/unloading teams. High-ValueStorage Access: Control access to specific high-security storage areas where valuable goods are kept.

1. Restaurants and Cafes:

Staff Attendance: Automate the clock-in/out for kitchen staff, waiters, and baristas. Inventory Area Access: Restrict access to inventory or cash-handling areas to authorized employees only.

1. Retail Security and Employee Monitoring (with clear policies):

Employee Break Tracking: Automatically log when employees go on breaks and return, ensuring adherence to schedules.Shift Handovers**:** Verify the presence of staff at shift changeovers. (This overlaps slightly with general employee attendance but focuses on specific operational aspects).

**CONCLUSION**

In this section, we have outlined the objectives, tools, system requirements, and the overall design of the face detection system using Python and OpenCV. We have proposed an efficient design that uses Haar Cascade Classifiers for detecting faces in static images and real-time video streams. The next steps in the project will involve the implementation phase, where the system will be coded and tested. The design ensures that the system can scale to support future improvements, such as adding facial recognition or emotion detection capabilities. The implementation will focus on optimizing the system for both accuracy and speed, ensuring that it meets the real-time demands of video-based applications while maintaining usability and robustness. Face recognition systems are part of facial image processing applications and their significance as a research area are increasing recently. Implementations of system are crime prevention, video surveillance, person verification, and similar security activities. The face recognition system implementation can be part of Universities. Face Recognition Based Attendance System has been envisioned for the purpose of reducing the errors that occur in the traditional (manual) attendance taking system. The aim is to automate and make a system that is useful to the organization such as an institute. The efficient and accurate method of attendance in the office environment that can replace the old manual methods. This method is secure enough.

## REFERENCES

## \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

[1] Viola P. & Jones M., "Rapid object detection using a boosted cascade of simple

Features", in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn.*, 2001.

[2] Bradski G. & Kaehler A., "Learning OpenCV: Computer Vision with the OpenCV

Library", O’Reilly Media, 2008.

[3] Lowe D. G., "Distinctive image features from scale-invariant keypoints", *Int. J. Comput.*

*Vis.*, Vol. 60, No. 2, pp. 91–110, 2004.

[4] Rowley H. Baluja, S. & Kanade T., "Neural network-based face detection", *IEEE Trans.*

*Pattern Anal. Mach. Intell.*, Vol. 20, No. 1, pp. 23–38, 1998.