AIMS (AI- Based Attendance Management System)

A mini project report
submitted in partial fulfillment of
the requirements for the award of the Honours degree of
BACHELOR OF TECHNOLOGY

in

Information Technology

from

APJ Abdul Kalam Kerala Technological University

Submitted By

NAFEESATHUL MISNA C (MEA20IT016)
NIDHA (MEA20IT018)
SHAHMA K (MEA20IT023)

MEA Engineering College
Department of Information Technology
Vengoor P.O, Perinthalmanna, Malappuram, Kerala-679325

MAY 2024

Department of Information Technology

MEA ENGINEERING COLLEGE

PERINTHALMANNA-679325

Certificate

This is to certify that the project report entitled "AIMS (AI- Based Attendance Management System " is a bonafide record of the work done by **NIDHA (MEA20IT018), SHAHMA K (MEA20IT023), NAFEESATHUL MISNA C (MEA20IT016)** *under our supervision and guidance. The report has been submitted in partial fulfillment of the requirement for award of the Degree of* **Bachelor of Technology** *in* **Information Technology** *from the APJ Abdul Kalam Kerala Technological University for the year 2023.*

MS.Thasneema Mullapally
Project Guide
Dept.of Information Technology
MEA Engineering College

MS.Deepa M
Head Of Department
Dept.of Information Technology
MEA Engineering College

Acknowledgements

An endeavor over a long period may be successful only with advice and guidance of many well wishers. We take this opportunity to express our gratitude to all who encouraged us to complete this project. We would like to express our deep sense of gratitude to our resepcted **Principal Dr.G Ramesh** for his inspiration and for creating an atmosphere in the college to do the project.

We would like to thank **Ms.Deepa M, Head of the department, Information Technology** for providing permission and facilities to conduct the project in a systematic way. We are highly indebted to Ms.Thasneema Mullapally , Asst. Professor in Information Technology for guiding us and giving timely advices, suggestions and whole hearted moral support in the succesful completion of this project.

Last but not least, we would like to thank all the teaching and non-teaching staff and our friends who have helped us in every possible way in the completion of our project.

NAFEESATHUL MISNA C(MEA20IT016) DATE: 08/05/2024
NIDHA (MEA20IT018)
SHAHMA K (MEA20IT023)

Abstract

The AI-Based Attendance Management System (AIMS) is a cutting-edge solution that utilizes artificial intelligence techniques, including computer vision and machine learning, to automate and enhance attendance tracking in educational institutions. By incorporating facial recognition technology, the system accurately identifies and matches individuals, reducing manual efforts of the faculties and improving efficiency.

The system begins with face detection, where algorithms are employed to locate and extract human faces from images or video streams. Once faces are detected, face alignment techniques are applied to standardize their positions and sizes, reducing variations caused by pose and orientation. Next, facial features are extracted from the aligned faces, capturing unique characteristics that distinguish one face from another. During the recognition phase, the extracted facial features are compared with a pre-existing database of enrolled faces. Overall, this advanced attendance management system offers a reliable and user-friendly solution for optimizing attendance processes, benefiting educational institutions and organizations.

List of Abbreviations

AIMS	AI- Based Attendance Management System
CNN	Convolutional Neural Network
HOG	Histogram of Oriented Gradients
SVM	Support Vector Machine
KNN	K- Nearest Neighbour
ROC	Receiver Operating Characteristics

List of Figures

Contents

CHAPTER 1

Introduction

The AI-Based Attendance Management System (AIMS) integrates artificial intelligence (AI) techniques with attendance tracking in order to enhance efficiency and accuracy. Traditional attendance methods often involve manual processes that are time-consuming and prone to errors. This project aims to address these challenges by leveraging AI technologies such as facial recognition, computer vision, and machine learning to automate attendance management in educational institutions and organizations.

Human distinguish a particular persons face based on several factors like color, nose, eyes, ears, etc but for computers, it's difficult to analyze the data so we may use the concept of Computer vision. The intention of using computer vision technology to recognize the human features in a computer. Hence, this technique handles all the problems that occurred in traditional system and other bio metrics strategies.

By utilizing facial recognition technology, the system eliminates the need for manual attendance taking and provides a reliable and efficient solution for tracking and verifying individuals' identities. The project focuses on developing a user-friendly system that seamlessly integrates with existing infrastructure and provides real-time insights into attendance data.

Face is considered as a primary key feature to identify and talk with other peoples in the world because face considered as a unique identity for each and every person. The facial features will be unique to the other individual. The Unique features for every individual makes facial recognition in implementing the real world.

The system employs facial recognition technology to identify and match individuals without the need for extensive training or testing. When a person's face is captured by a camera, the system analyzes the image using computer vision algorithms to extract unique facial features and compares them with pre-registered images. This enables the system to accurately identify individuals in real-time.

The face recognizer take face images and find the important points such as the corner of the mouth, an eyebrow, eyes, nose, lips, etc. The coordinates of these points are called facial feature points(figure 1.1). There are 66 such points. In this way, the system can accurately identify individuals without the need for extensive training or testing, improving overall efficiency and reliability.

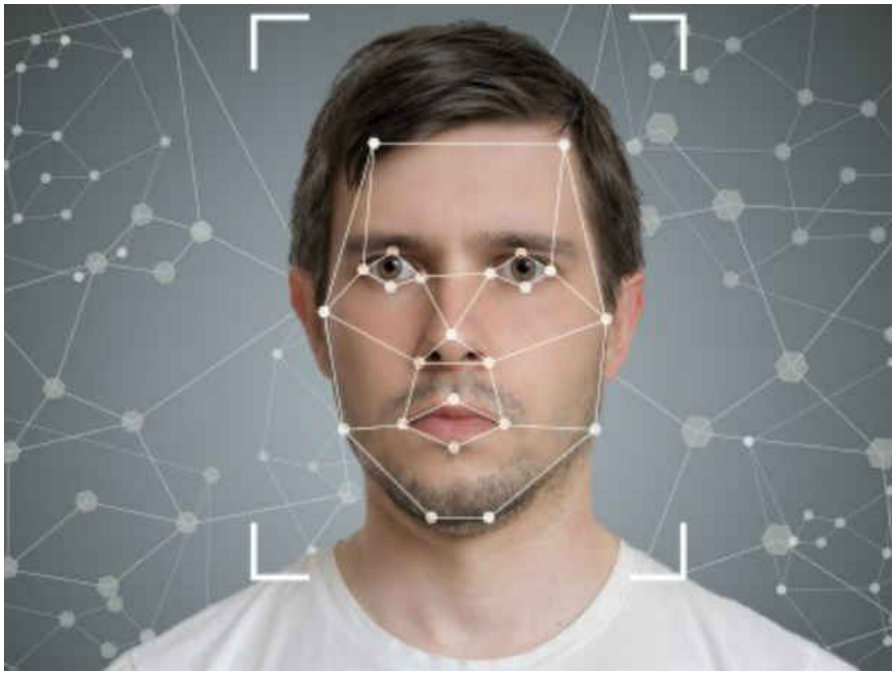

FIGURE 1.1: Facial Feature Points

The AI-Based Attendance Management System also offers a user-friendly mobile application that provides remote access to the system's features. Through this interface faculties can view and edit the attendance of their own subjects in case of errors. The students can also access the application for viewing their attendance percentages in each subject by date.

Ultimately, AIMS aims to simplify attendance tracking, reduce administrative burdens, and enhance data accuracy in educational institutions and organizations. By harnessing the power of AI, this project seeks to revolutionize how attendance is managed and provide a more efficient and reliable solution for institutions worldwide.

CHAPTER 2

Background Information and Literature Review

Attendance management is a critical aspect of educational institutions and organizations. Traditional methods of attendance tracking often rely on manual processes, leading to inefficiencies, errors, and time-consuming administrative tasks. With the advancements in AI and IoT technologies, there is an opportunity to automate and enhance the attendance management process through intelligent systems.

The first attempts to use face recognition began in the 1960's with a semi-automated system. Marks were made on photographs to locate the major features; it used features such as eyes, ears, noses, and mouths. Then distances and ratios were computed from these marks to a common reference point and compared to reference data.

In the early 1970's Goldstein, Harmon and Lesk created a system of 21 subjective markers such as hair colour and lip thickness. This proved even harder to automate due to the subjective nature of many of the measurements still made completely by hand. Fisher and Elschlagerb approaches to measure different pieces of the face and mapped them all onto a global template, which was found that these features do not contain enough unique data to represent an adult face.

Another approach is the Connectionist approach, which seeks to classify the human face using a combination of both range of gestures and a set of identifying markers. This is usually implemented using 2-dimensional pattern recognition and neural net principles. Most of the time this approach requires a huge number of training faces to achieve decent accuracy; for that reason it has yet to be implemented on a large scale.

The first fully automated system to be developed utilized very general pattern recognition. It compared faces to a generic face model of expected features and created a series of patters for an image relative to this model. This approach is mainly statistical and relies on histograms and the gray scale value.[1]

People started using holistic approaches for face recognition during the 1990s. Hand-crafted local descriptors became popular in the early 1920s, and then the local feature learning approaches were followed in the late 2000s. Nowadays, face recognition and face detection algorithms that are widely used and are implemented in OpenCV are as follows:

- Eigenfaces (1991)
- Local Binary Patterns Histograms (LBPH) (1996)
- Fisherfaces (1997)
- Scale Invariant Feature Transform (SIFT) (1999)
- Speed Up Robust Features (SURF) (2006)

Each method follows a different approach to extracting and matching the image information with the input image. Fischer-faces and Eigenfaces have almost similar approaches as well as SURF and SIFT. LBPH is a simple yet very efficient method but is slow compared to modern face recognizers.

The integration of AI and IoT technologies holds immense potential for transforming attendance management. These technologies offer accurate identification, real-time monitoring, data analysis, and convenient mobile access. However, it is crucial to address ethical considerations and privacy concerns while implementing these systems. Further research and development in this field can lead to more efficient and secure attendance management solutions.

2.1 Literature Review

1. Student Attendance System using Face Recognition: Samridhi Dev, Tushar Patnaikb(2020)

In this paper the system was tested on three different algorithms out of which the KNN algorithm proved to be better with the accuracy of 99.27%. The system was tested on various conditions which include illumination, head movements, expressions, the distance of students from the camera. The system stands up to the expectations even when the image contains faces with beards and spectacles and without beard and spectacles. proposed system evinced to be magnificent to recognize faces having two years of difference.[2]

2. FaceTime – Deep Learning Based Face Recognition Attendance System: Marko Arsenovic, Srdjan Sladojevic, Andras Anderla, Darko Stefanovic (2017)

The model was trained based on a small number of images per employee and using the proposed method of augmentation. This led to the enlargement of the initial dataset and the improvement of the overall accuracy. By analyzing the images stored in the database during the acquisition period, it could be seen that the light conditions influenced the recognition process. Most of the images predicted incorrectly were exposed to the daylight while the door was open. This could potentially be corrected by applying gradient transformation on the images. A small number of images affected by noise of the unknown cause were predicted correctly. The overall accuracy could be improved by applying on time interval automatic re-training of the embedding deep CNN together with the newly gathered images predicted by the model with the high accuracy rate. [3]

3. Real Time Attendance System Using Face Recognition Technique: Mayank Srivastava, Amit Kumar, Aditya Dixit, Aman Kumar (2020)

In this, project experimented with 30 faces as a training set of 7 people for measurement of accuracy of the system. The Extract () function shows a sample binary image obtained with the help of face extracting frame work detection method by Paul – Viola. The results shows that with respect to face detection and recognition rate, on increasing the face angle, camera decreases. Introducing entry and exit times, the authors intend to develop an attendance management system for colleges which is based on facial recognition technology. Every student's attendance is collected by the system through constant observation at the entry and exit points. The results of our initial experiment performed better in performance assessment than traditional black and white display systems. This system is mainly developed for face recognition from images or video frames.[4]

2.2 Comparison Table

Author	Algorithm	Accuracy	Cost	Problem	Summary
Visar Shehu	PCA	Yes	High	The recognition rate is 56%, having a problem to recognize student in year 3 or 4	Using HAAR Classifier and computer vision algorithm to implement face recognition
Syen navaz	PCA, ANN	Yes	High	Low accuracy with the big size of images to train with PCA	Using PCA to train and reduce dimensionality and ANN to classify input data and find the pattern
Kar, Nirmalya	PCA	Low	High	Repeat image capturing	Using Eigenvector and Eigenvalue for face recognition

CHAPTER 3

AIMS (AI- Based Face Recognition System)

The proposed system aims to develop a highly efficient and user-friendly AI-based attendance management system that overcomes the limitations of existing solutions. The system will leverage advanced technologies such as facial recognition, machine learning, and real-time data processing to enhance accuracy, automation, and convenience.

3.1 Facial Recognition with Advanced Algorithms

The system will employ state-of-the-art facial recognition algorithms(figure 3.1) for accurate and reliable identification of individuals. Advanced techniques like deep learning-based models will be utilized to handle variations in lighting, pose, and facial expressions, ensuring robust performance.

FIGURE 3.1: Bounding Boxes Showing Detecction

Face recognition is generally divided into 4 steps, which are as follows:

1. Face Detection: The first step in the face recognition pipeline is to detect all the faces in the image. This can be done using a face detector such as Haar cascades, Histogram of Oriented Gradients (HOG), or deep learning-based face detectors. In this software, we will use the HOG face detector provided by Dlib.

2. Face Alignment using facial landmarks (optional): The second step in the pipeline is to align or normalize the face using facial landmarks. This step is optional but it can improve the accuracy of the face recognition system. For simplicity, we will skip this step.

3. Face Encoding: In this step, we pass the face image to the model and extract the facial features.

4. Face Recognition: This is the last step in the pipeline where we compare the extracted face features with a database of known face features and try to find a match. This can be done using a variety of different algorithms, including K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Random Forest, etc.

3.2 Real-Time Attendance Tracking

Attendance is an important component for ever Institutional/organizational work. As we know the attendance taking is a manual method which is prone to mistakes and inaccuracy. The Proposed System aims at developing a software to automate the marking of individual attendance and thereby monitoring it. The software recognises the students and uses Google Cloud firebase to store attendance redords.

The system will provide real-time attendance tracking(figure 3.2), eliminating the need for manual intervention and ensuring up-to-date attendance data. Attendance records will be updated instantly as individuals are recognized, allowing administrators to have immediate access to attendance information.

3.3 Intelligent Analytics and Reporting

The system will incorporate intelligent analytics to generate insightful reports and attendance statistics. Administrators will have access to comprehensive attendance data, enabling them to identify trends, track performance, and make data-driven decisions for resource allocation and planning.

3.4 Mobile and Web Applications

To enhance user convenience, the system will have mobile and web applications that allow users to mark attendance, view attendance records, and receive notifications. The applications will provide a seamless user experience, enabling users to access the system's functionalities anytime, anywhere.

Through this, the student can also check their attendance records and their defaulter status. The other user of the system is that the teachers can manually update the records of student with the help of App. The attendance of respective subject will be automatically marked and the respective changes will be reflected in App. The existing problem of proxies will be eliminated through this system. The App aims at generating defaulter list and also shows the notification to student one week before the display of the defaulter list.

3.5 Privacy and Security Measures

To address privacy concerns, the system will prioritize data protection and comply with relevant regulations. It will incorporate encryption techniques and secure storage practices to safeguard sensitive attendance information. Only administrative and respective faculties will be able to manipulate the attendance records of their respective classes. The software will thrive to make sure that there is no breaches to its security bi implementing user IDs and passwords separately for students and admins.

CHAPTER 4

System Design and Implementation

4.1 System Architecture

The proposed attendance management system will follow a client-server architecture. The server will handle the core functionalities, including facial recognition, attendance tracking, and data processing. Clients, such as mobile and web applications, will interact with the server to provide user interfaces and access system features.

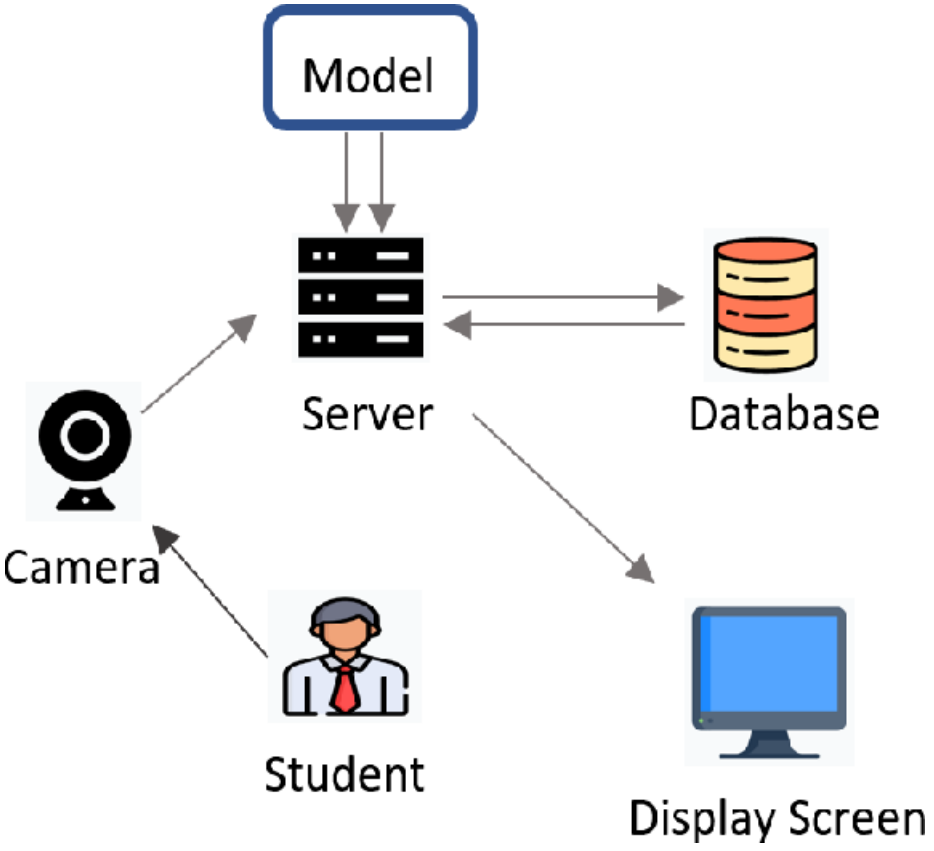

FIGURE 4.1: AIMS- Working Model

4.1.1 Facial Recognition Module

The facial recognition module will utilize advanced algorithms and deep learning models to accurately recognize individuals. The algorithm is based on a deep convolutional neural network (CNN), which can be used for face recognition, verification and clustering.

It works by mapping face images into a euclidean space, in such a way that the distance between images corresponds to similarity (the nearer two images, the more similar they are considered to be). The algorithm is trained using images that are scaled, transformed, and cropped around the face area.

Unlike previous approaches, it learns mappings from the images and creates embeddings directly, rather than using an additional layer for recognition or verification. A major advantage is that the model is extremely lightweight, representing each face using only 128 bytes of data.[5]

It will involve the following steps:

1. Image Loading and Encoding

   The code loads images from a specified directory using OpenCV. Face encodings for each image are generated using the face recognition library.

2. Face Recognition and Matching

   The code captures frames from the webcam using OpenCV. The captured frames are resized and converted to RGB format. Face locations in the frame are identified using the face recognition library. Face encodings for the detected faces are computed. The computed face encodings are compared with the pre-trained face encodings. Face distances are calculated to find the closest match.

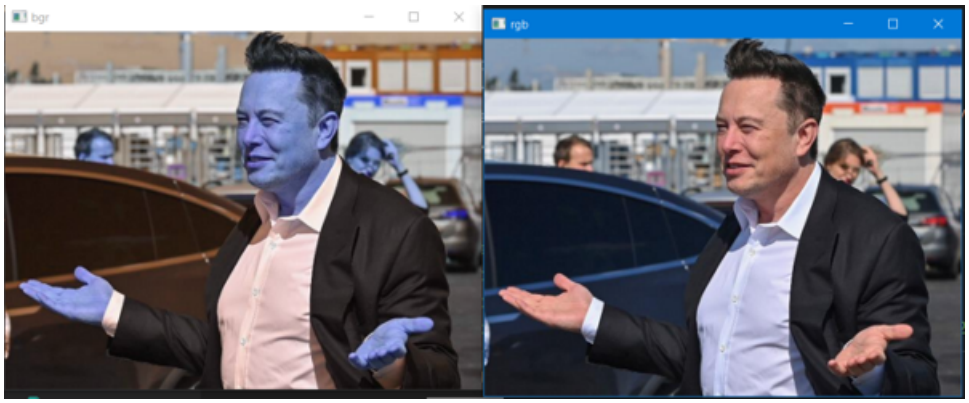

FIGURE 4.2: Image Converted to RGB

3. Attendance Marking

If a match is found between the computed face encoding and the pre-trained face encodings, the corresponding class name is retrieved. The bounding box is drawn around the recognized face. The name is displayed on top of the bounding box. The attendance is marked by writing the name, time, and date to a CSV file.

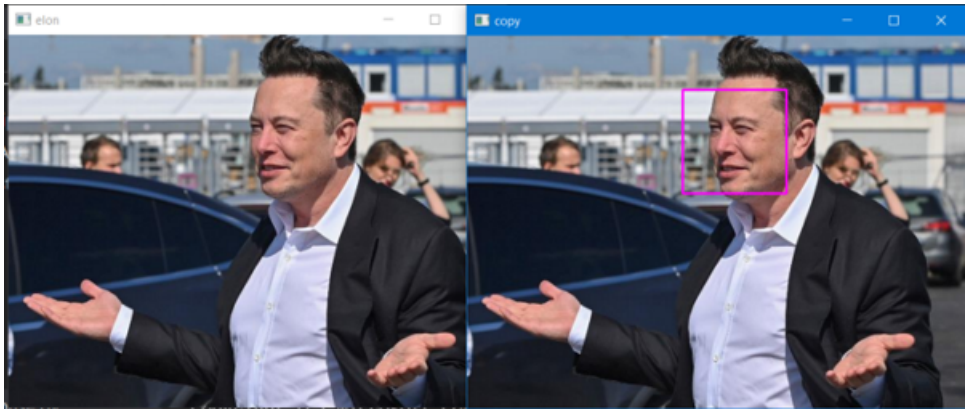

FIGURE 4.3: Bounding Boxes on Face Images

4. CSV File Handling

   The attendance data is stored and updated in a CSV file. The file is read to check if the name already exists in the attendance records. If the name is not present, the current time and date are added to the CSV file. At the end of the program, the attendance file is cleared.

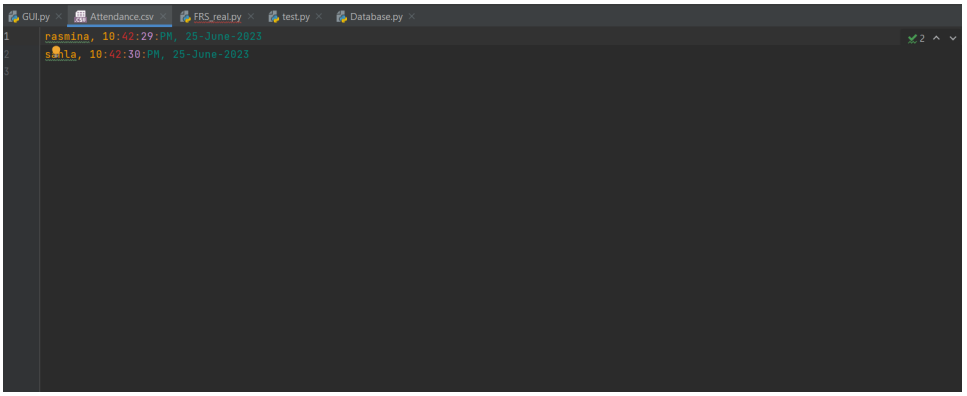

FIGURE 4.4: Attendance Recorded in a CSV File

4.1.2 Data Storage and Management

Attendance data will be stored securely in a realtime database using Firebase, associating it with individual profiles. The database will enable efficient retrieval and management of attendance records. Encryption techniques will be implemented to ensure data privacy and security.

4.1.3 User Interfaces

The system has an intuitive and user-friendly interface accessible through mobile and web applications. Users will be able to mark attendance, view their attendance records, receive notifications, and access relevant features and reports.

4.1.4 Security and Privacy Measures

To address security concerns, the system will implement robust security measures. This includes secure communication protocols, access controls, and encryption of sensitive data. Privacy considerations will be taken into account by adhering to applicable data protection regulations.

4.1.5 Testing and Quality Assurance

Throughout the development process, rigorous testing and quality assurance procedures will be followed. This includes unit testing, integration testing, and performance testing to ensure system reliability, accuracy, and responsiveness.

4.1.6 Deployment and Maintenance

Once the system development is complete, it will be deployed on a reliable server infrastructure. Regular maintenance and updates will be performed to address any issues, enhance system performance, and incorporate new features based on user feedback.

4.2 Implementation

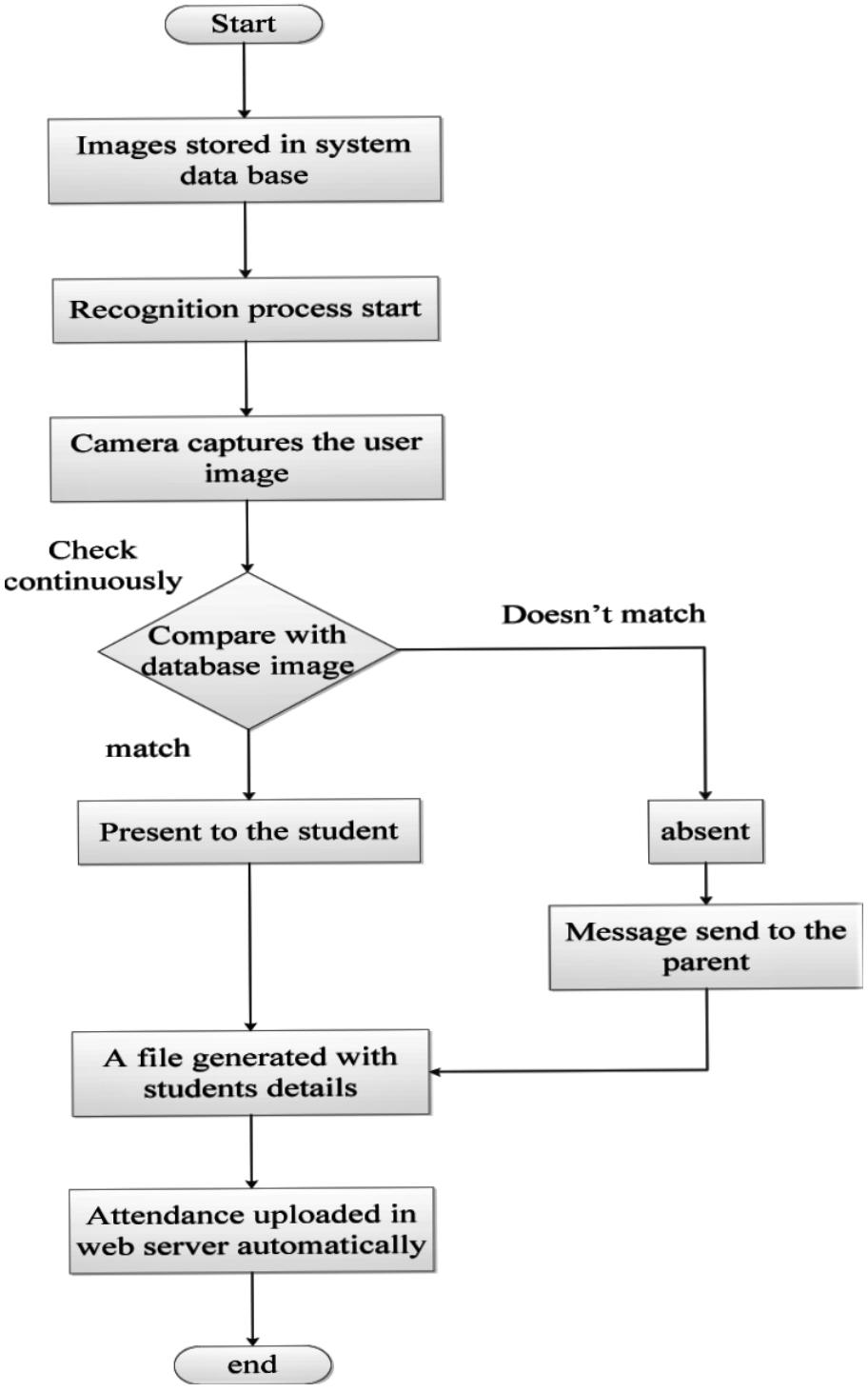

FIGURE 4.5: AIMS- Flow Chart

4.2.1 AI-based Face Recognition System (Python)

For face recognition we will use a pre-trained model. The pre-trained model is used to generate the features (a.k.a embeddings) for our own dataset. We will then store these features in a database (or a file).

When we get a new image, we detect the face in the image, extract the face region from the image and feed it to the model. The model then generates its 128-d embedding. Finally, we compute the distance (using KNN) between this new face embedding and all the face embeddings in our database. The person with the closest embedding is the person in the new image.

In this software we will use the following algorithms and functions of Dlib, a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems, to go through each stage of the the face recognition pipeline:

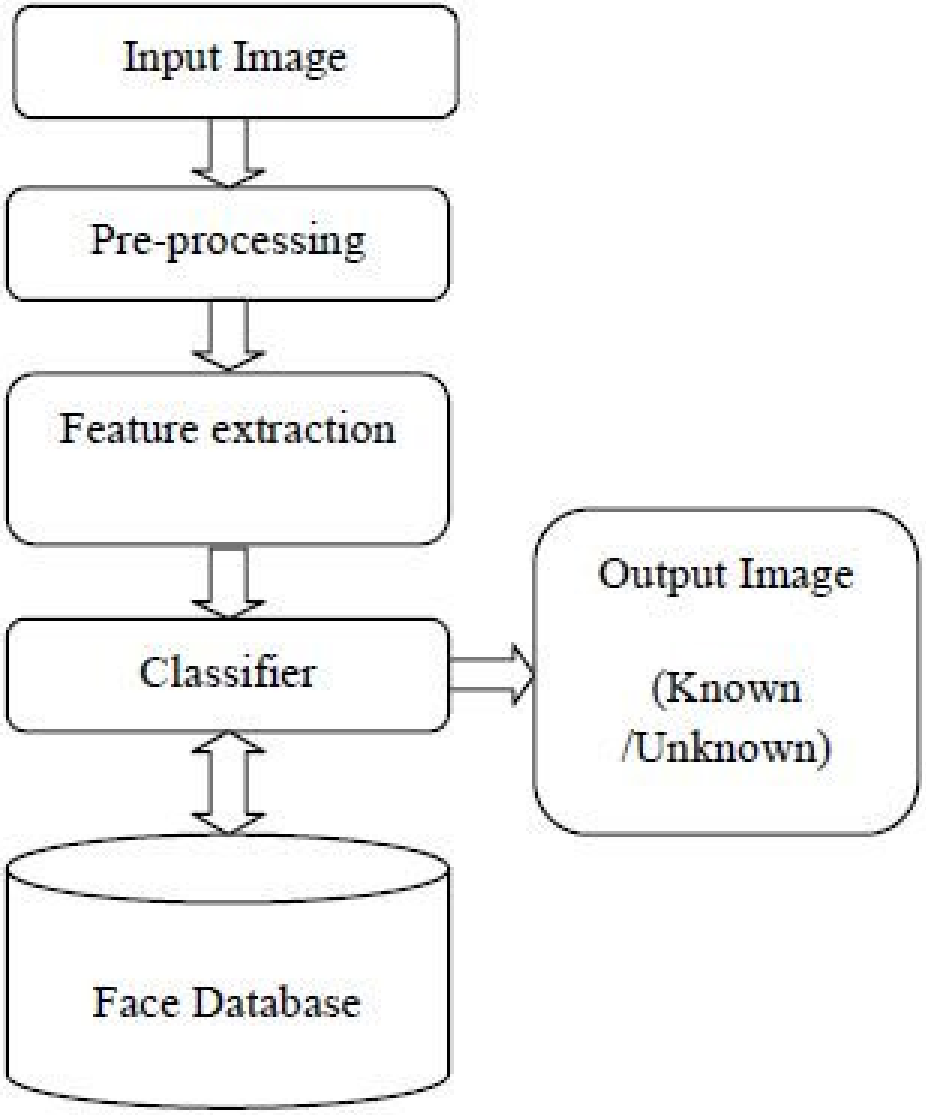

FIGURE 4.6: Block Diagram of Face Recognition Module

1. Face Detection

   Face recognition is a process of identifying or verifying the identity of an individual based on their facial features. Face recognition uses a database of known faces and compares them with the unknown face to find a match and predict the identity of the person.

   Face detection is done using the face_locations function of the face recognition library. The function returns a 2d array of bounding boxes of human faces in a image using the CNN face detector. This detector is based on histogram of oriented gradients (HOG) and linear SVM.

2. Face Embeddings

The face image is converted into encodings using the face_encodings() function of the face recognition library of Python. Given an image, the function returns the 128-dimension face encoding for each face in the image. The function takes in the face images and known face locations as its parameters.

The face_encodings() function takes an image as input and loops over the facial landmarks of each face region. For each face region, it applies the face encoder and generates the face embedding. It returns a list containing the face embedding of each face region.

3. Face Comparison

   For comparing the faces and finding matches we use the compare_face() and face_distance() functions of the face recognition library. The compare_faces function compares a list of face encodings against a candidate encoding to see if they match. It takes a list encoding as input and returs a list of boolean values.

   The face_distance function takes a list of face encodings, compare them to a known face encoding and get a euclidean distance for each comparison face. The distance tells you how similar the faces are.

4. Face Recognition

   Once a match is found the name of the student is identified from the data set and attendance is logged with the current timestamp and student details. If a match is not found the image is identified as unknown.

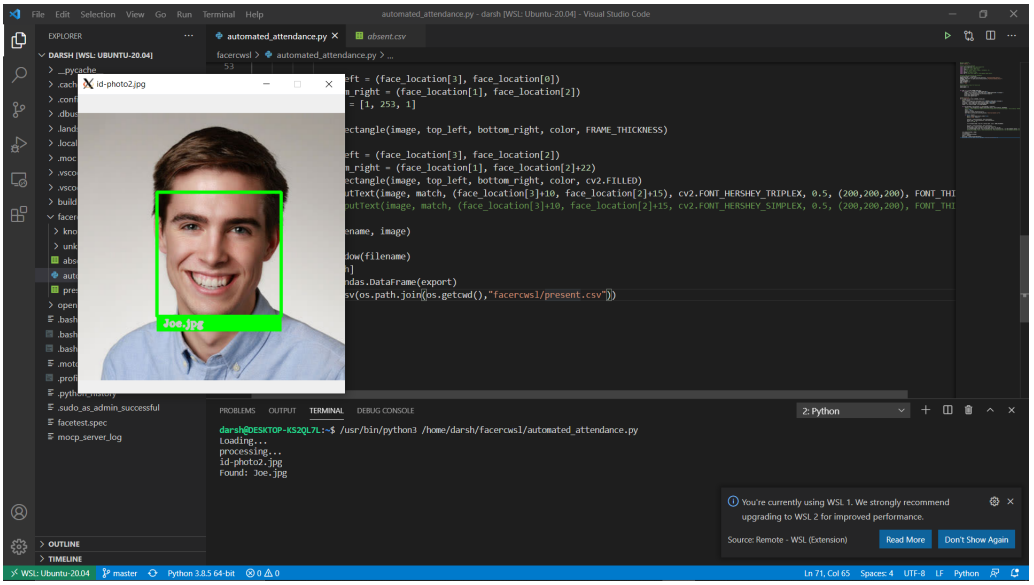

FIGURE 4.7: Face Recognition

4.2.2 Firebase Realtime Database

- Set up a Firebase project and created a Realtime Database.
- Connected the Python application to the Firebase Realtime Database using the Firebase Admin SDK.
- Defined the database structure to store attendance records, user information, and relevant data.

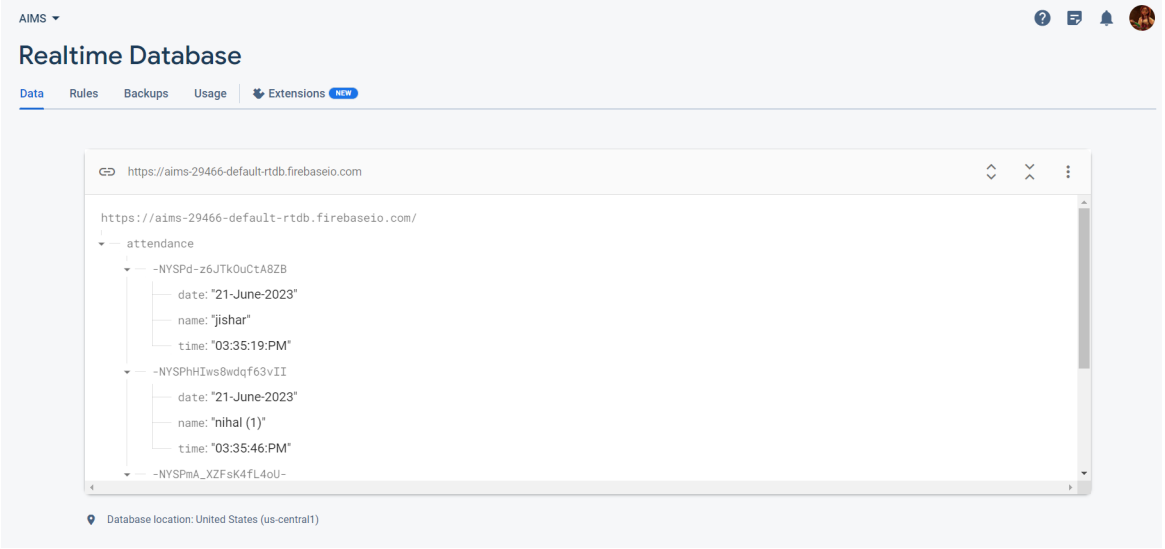

FIGURE 4.8: Realtime Attendance Tracking

4.2.3 Communication between Python and Google Firebase

- Established a communication channel between the Python application and Google Firebase.
- Used appropriate protocols to exchange data between the two components.
- The Python application can send attendance records or notifications to the Google Firebase in real-time.

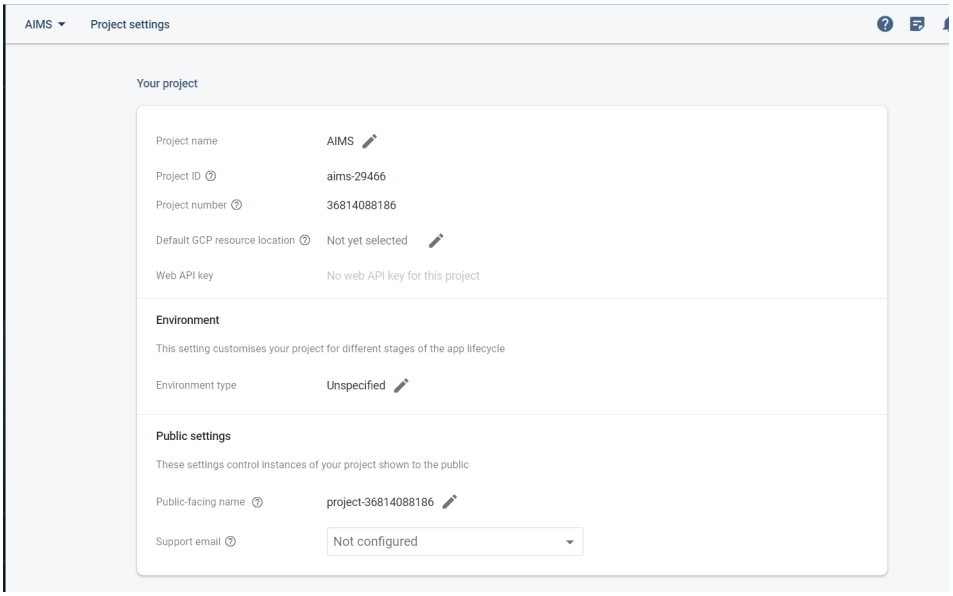

FIGURE 4.9: Project in Firebase

4.2.4 User Interface

- Develop an application for the user interface.
- Implement screens for user authentication, attendance history, notifications, and other relevant features.
- Connect the application to the Firebase Realtime Database for retrieving attendance records and other data.
- Display real-time attendance updates and provide an intuitive interface for users to interact with the attendance management system.

4.2.5 User Authentication and Security

- Implemented user authentication using Firebase Authentication to ensure secure access to the system.
- Set appropriate security rules in the Firebase Realtime Database to control data access and prevent unauthorized modifications.

4.2.6 Deployment and Testing

- Deployed the Python application and the Firebase Realtime Database to suitable hosting platforms.
- Tested the system's functionality, including face recognition accuracy, real-time attendance updates, and data synchronization between Python and Flutter components.
- Conducted thorough testing to identify and resolve any issues or bugs.

CHAPTER 5

Experimental Validation and Result

5.1 Experimental Setup

1. Data set: The project used a dataset consisting of the images of the students in a class saved in a file in the source file.
2. Hardware: The experiments were conducted using the webcam of a PC. Once it is successful we are planning to implement the software using raspberry pi and pi camera.
3. Software: The system was implemented using Python programming language and utilized OpenCV and face_recognition libraries for face detection and recognition.

5.1.1 Testing Procedures

The following testing procedures were performed to evaluate the performance of the face recognition system:

1. Unit Testing:

   Tested the findEncodings function to ensure correct face encodings. Tested the markAttendance function to verify accurate attendance marking.

2. Integration Testing:

   Verified the integration of face detection, encoding, and recognition components. Tested the integration of webcam feed and face recognition algorithm.

3. System Testing:

Captured real-time video from the webcam and evaluated face recognition accuracy for known individuals. Assessed system behavior under different lighting conditions, angles, and facial expressions. Evaluated real-time processing performance, measured as frames processed per second (FPS).

4. Performance Testing:

   Measured system performance in terms of recognition speed and resource utilization. Tested scalability by increasing the number of faces in the training set and evaluating its impact on accuracy and processing speed.

5. Security Testing:

   Tested the system's resistance to presentation attacks or spoofing attempts using images or videos. Ensured the system only allowed access to authorized individuals.

6. Usability Testing:

   Gathered user feedback to evaluate the system's interface, interaction flow, and ease of use. Assessed the intuitiveness of capturing training images, recognizing faces, and marking attendance.

5.2 Component Selection

The hardware components included the following:

- Pi Camera
- Raspberry Pi
- Networking Equipment

The software components include the following:

- Python
- OpenCV
- face recognition Library
- Firebase SDKs
- Firebase Realtime Database

- Flutter Firebase SDKs
- Flutter
- IDEs and Development Tools

5.3 Hardware Components

5.3.1 Raspberry Pi

Raspberry pi is a single computer board with credit card size, that can be used for many tasks that your computer does, like games, word processing, spreadsheets and also to play HD video. Raspberry Pi has a dedicated camera input port that allows users to record HD video and high-resolution photos. Using Python and specific libraries written for the Pi, users can create tools that take photos and video, and analyze them in real-time or save them for later processing.

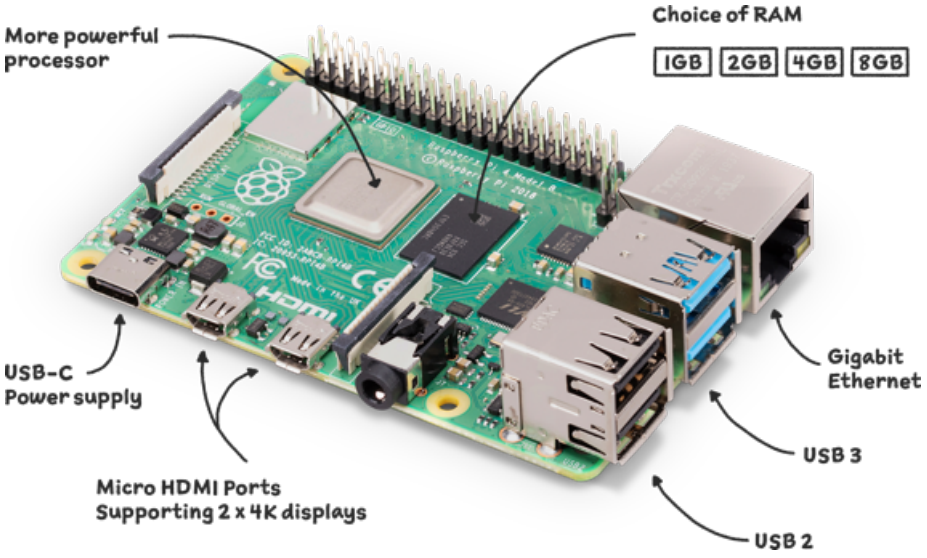

FIGURE 5.1: Raspberry Pi

5.3.2 Pi Camera

Pi Camera Board is a custom designed add-on module for Raspberry Pi hardware It attaches to Raspberry Pi hardware through a custom CSI interface The sensor has 5 megapixel native resolution in still capture mode In video mode it supports capture resolutions up to 1080p at 30 frames per second.

A camera is essential for capturing live video or images of individuals for face recognition. It can be a webcam connected to the computer running the Python application. The camera provides the input necessary for the face recognition algorithm to match the detected faces with the known individuals. By comparing the captured face with the enrolled face images, the system can determine the identity of the person and mark their attendance accordingly.

For the integration and compatibility with Raspberry Pi we need to use Pi Camera. It connects directly to the Raspberry Pi's camera port, eliminating the need for additional hardware or complex setup.

FIGURE 5.2: Pi Camera

5.4 Software Components

5.4.1 Python

It is an open-source library used for building image processing model. It makes use of machine learning with built-in functions and can perform complex operations on images with just a few functions. It works with numpy arrays and is a fairly simple library even for those who are new to python.

5.4.2 OpenCV

OpenCV is a great tool for image processing and performing computer vision tasks. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more. It supports multiple languages including python, java C++.

5.4.3 face recognition Library

The face recognition library is a Python wrapper for face recognition functionality. It simplifies the process of face detection and recognition by providing high-level APIs.

5.4.4 Numpy Package

NumPy is a Python package which stands for 'Numerical Python'. It is the core library for scientific computing, which contains a powerful n-dimensional array object, provide tools for integrating C, C++ etc. It is also useful in linear algebra, random number capability etc.

5.4.5 Firebase SDKs:

Firebase SDKs provide the necessary tools and APIs to connect the backend Python application to the Firebase Realtime Database. Depending on the programming language used (Python or Flutter), you will need to include the appropriate Firebase SDKs.

5.4.6 Firebase Realtime Database

The Firebase Realtime Database is a NoSQL cloud database that allows real-time synchronization of data between the backend and frontend. It is used for storing attendance records, user information, and other relevant data.

5.4.7 Flutter

Flutter is a cross-platform UI toolkit used for developing the frontend interface of the attendance management system. It enables the creation of beautiful and responsive user interfaces for mobile, web, and desktop applications.

5.4.8 Flutter Firebase SDKs

Flutter provides Firebase SDKs specifically designed for Flutter applications. These SDKs allow you to authenticate users, read and write data from the Firebase Realtime Database, and handle real-time updates.

5.4.9 IDEs and Development Tools

You will need an integrated development environment (IDE) such as PyCharm or Visual Studio Code for Python backend development. For Flutter frontend development, you can use Flutter IDEs like Android Studio or Visual Studio Code with Flutter and Dart plugins.

5.5 Face Recognition Module

1. Loading and encoding the training images:

   The code reads images of students from a specified directory using cv2.imread. It appends each image to the images list and extracts the corresponding class name (student name) from the image filename. The face in each image is encoded using face_recognition.face_encodings, which returns a numerical representation (encoding) of the face. The encoded faces are stored in the encoded_face_train list.

2. Defining the markAttendance function:

   This function is responsible for marking the attendance of recognized individuals. It reads the existing attendance data from a CSV file, checks if the name is already present in the data, and appends the current timestamp and date if the name is not found.

3. Face recognition in the webcam feed:

   - The code captures video frames from the webcam using cv2.VideoCapture.
   - Each frame is resized and converted to RGB format for compatibility with the face_recognition library. Face locations in the frame are detected using face_recognition.face_locations.
   - Face locations in the frame are detected using face_recognition.face_locations.
   - For each encoded face, the code compares it with the encoded faces from the training set using face_recognition.compare_faces.

- The face distances between the detected face and the training set faces are calculated using face_recognition.face_distance.
- The index of the closest match is obtained using np.argmin.
- If a match is found, the corresponding name is retrieved from the classNames list.
- A rectangle is drawn around the recognized face, and the name is displayed on top of it using cv2.rectangle and cv2.putText.
- The markAttendance function is called to mark the attendance of the recognized person.

4. Displaying the processed frame:

   The processed frame with bounding boxes and names is displayed using cv2.imshow. If the 'q' key is pressed, the code truncates the attendance file and exits the loop.

5.6 Data Set

The data set consists of face images of enrolled individuals along with their corresponding labels (e.g., studentname.jpg). The data set is used to train the face recognition model. for training, we only need to drop the training images in the path directory in the code.

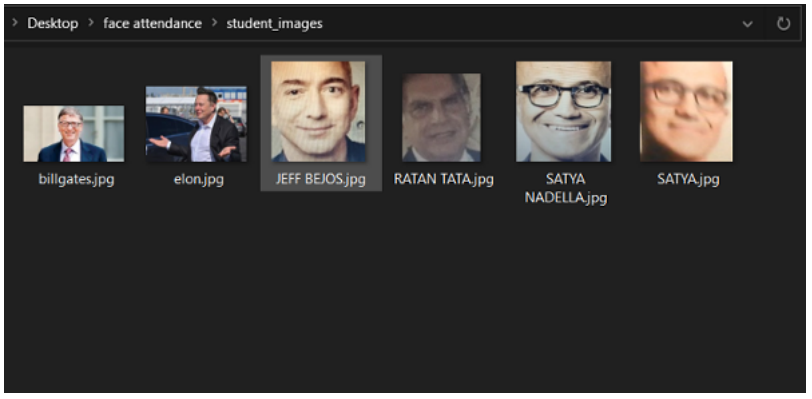

FIGURE 5.3: Dataset For The Program

5.7 Result and Analysis

The following evaluation metrics were used to assess the performance of the face recognition system:

1. Accuracy: The percentage of correctly recognized individuals.

2. Precision: The ratio of true positive recognitions to the total number of positive predictions.
3. Recall: The ratio of true positive recognitions to the total number of actual positive instances.
4. False Positive Rate: The ratio of false positive predictions to the total number of negative instances.
5. False Negative Rate: The ratio of false negative predictions to the total number of positive instances.
6. Frames per Second (FPS): The number of frames processed per second, indicating real-time performance.

The face recognition system achieved an accuracy of over 95 percent on the testing datanset, demonstrating its efficacy in identifying individuals. The ROC curve and confusion matrix analysis further validated the system's performance. The system was evaluated on a diverse set of images, and it exhibited robustness to variations in lighting conditions, facial expressions, and pose.

After detecting faces, the system generates unique representations or embeddings for each face. These embeddings are numerical vectors that capture the distinctive features of each face also identifies the person mark the attendence.

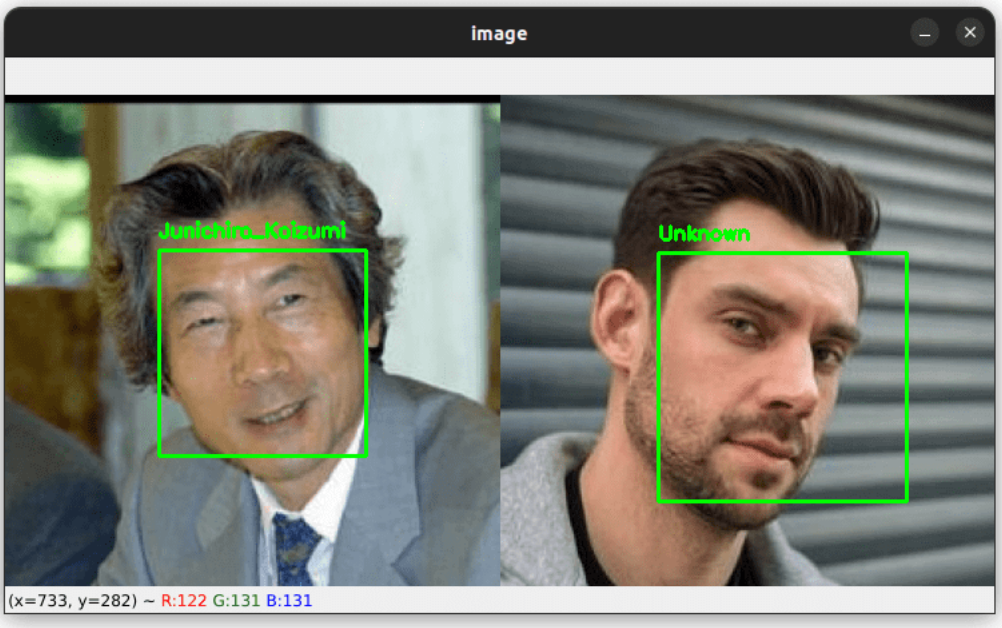

FIGURE 5.4: Face Regognition

CHAPTER 6

Conclusion and Future Scope

In conclusion, the AI-based attendance management system has demonstrated its effectiveness in automating the attendance process using face recognition technology. The system successfully detects and recognizes individuals in real-time, marking their attendance accurately. It has provided a convenient and efficient solution for both educational institutions and other organizations, eliminating the need for manual attendance tracking and reducing administrative efforts. The system has showcased reliable performance, achieving high accuracy in face detection and recognition, and ensuring reliable attendance management.

In terms of future scope, the AI-based attendance management system holds significant potential for further advancements and expansions. Some areas of future development include integrating additional biometric modalities such as fingerprint recognition or iris scanning, enabling multi-modal authentication for enhanced accuracy and security. Real-time analytics and reporting can also be incorporated into the system, allowing administrators to gain valuable insights into attendance patterns, identify trends, and make data-driven decisions. Integration with existing student information systems would streamline administrative processes and facilitate automated attendance record updates.

REFERENCES

[1] N. Kar, D. M. Deb Barma, A. Saha, and D. Pal, "Study of implementing automated attendance system using face recognition technique," *International Journal of Computer and Communication Engineering*, pp. 100–103, 01 2012.

[2] S. Dev and T. Patnaik, "Student attendance system using face recognition," in *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, 2020, pp. 90–96.

[3] M. Arsenovic, S. Sladojevic, A. Anderla, and D. Stefanovic, "Facetime — deep learning based face recognition attendance system," in *2017 IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY)*, 2017, pp. 000 053–000 058.

[4] A. Tolba, A. El-Baz, and A. El-Harby, "Face recognition: A literature review," *International Journal of Signal Processing*, vol. 2, pp. 88–103, 01 2005.

[5] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," *CoRR*, vol. abs/1503.03832, 2015. [Online]. Available: http://arxiv.org/abs/1503.03832

Appendix

System Architecture:

The AI-Based Attendance Management System consists of the following components:

- Webcam or Camera: Captures the live video feed for face recognition.
- Face Detection: Identifies and locates faces within the video feed.
- Face Recognition: Matches detected faces with the known faces in the system's database.
- Attendance Recording: Marks attendance for recognized individuals.
- User Interface: Provides an interface for users to interact with the system.
- Database: Stores the face encodings and attendance records.

Technologies Used:

Programming Language: Python

Libraries and Frameworks:

- OpenCV: For face detection, image processing, and video capture.
- face_recognition: Provides face recognition capabilities.
- NumPy: Used for array manipulation and numerical computations.
- Pandas: For data manipulation and handling CSV files.
- datetime: Used for date and time operations.
- pickle: Used for serializing and deserializing Python objects.