VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JNANA SANGAMA", BELAGAVI – 590 018

MINI PROJECT REPORT ON,

"FACE RECOGNITION BASED ATTENDANCE SYSTEM USING MTCNN"

Submitted in partial fulfilment of the requirement

for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted By

MEHAK BANDHRAL : 4GH22CS028

RADHIKA : 4GH22CS041

ROSHNI.A : 4GH22CS049

VIDYA S.M : 4GH22CS063

Under the Guidance of

DR. Thirthe Gowda B.E, M.Tech
Assistant Professor,
Dept. of CS&E,
GEC Hassan.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

GOVERNMENT ENGINEERING COLLEGE,

HASSAN-573201 2024-25

GOVERNMENT ENGINEERING COLLEGE HASSAN–573201

Affiliated to VTU, Belagavi & Approved by AICTE, New Delhi

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

Certified that the Mini Project work entitled "FACE RECOGNITION BASED ATTENDANCE SYSTEM USING MTCNN" carried out by **Ms. MEHAK BANDHRAL (4GH22CS028), Ms. RADHIKA (4GH22CS041), Ms. ROSHNI A (4GH22CS049)** and **Ms. VIDYA S M(4GH22CS049)** a bonafide students of **Government Engineering College Hassan** in partial fulfillment of **5th semester mini project** in **Computer Science and Engineering** of the **Visvesvaraya Technological University**, **Belagavi** during the year 2024-2025.It is certified that all correction/suggestions indicated during internal evaluation has been incorporated in the report. The Project report has been approved as it satisfies the academic requirements in respect of Mini Project work prescribed for the Degree.

Project Guide	**Project Co-Ordinator**	**Head Of Department**
Dr.Thirthe Gowda B.E.M.Tech	**Prof. Kiran M P** B.E,M.Tech	**Dr. Vani V G** B.E. M.Tech.Ph.D.
Assistant Professor,	Assistant Professor,	Head Of Department,
Dept. Of CS&E,	Dept. Of CS&E,	Dept. Of CS&E,
GEC Hassan.	GEC Hassan.	GEC Hassan.

External Viva

Name and Signature of Examiner with Date:

1. ……. ……

2. ……. …….

DECLARATION

We, **MEHAK BANDHRAL, RADHIKA, ROSHNI A** and **VIDYA S M** Students of fifth semester B.E**, GOVERNMENT ENGINEERING COLLEGE, HASSAN** bearing USN **4GH22CS028, 4GH22CS041**, **4GH22CS049** and **4GH22CS063** respectively, hereby declare that the Project entitled "FACE RECOGNITION BASED ATTENDANCE SYSTEM**"** has been carried out by me under the supervision of our Guide, **Dr. Thirthe Gowda** B.E., M. Tech, Department of CS&E, GEC Hassan, have submitted in partial fulfilment of the requirements for the award of the fifth semester of B.E in CS&E by the Visvesvaraya Technological University, Belagavi during the academic year 2024- 2025. This report has not been submitted to any other Organization/University for the award of degree or certificate.

Date:

Place: Hassan

Project Associates

MEHAK BANDHRAL

RADHIKA

ROSHNI A

VIDYA S M

ACKNOWLEDGEMENT

We consider it a privilege to whole-heartedly express our gratitude and respect to each and every one who guided and helped us in the successful completion of this Project Report.

We very thankful to the Principal **Dr. Girish D P,** for being kind enough to provide me an opportunity to work on a project in this institution.

We also thankful to **Dr. Vani V G**, Head of Department, Department of Computer Science, for their co-operation and encouragement at all moments of our approach.

We would greatly mention the enthusiastic influence provided by **Dr. Thirthe Gowda,** Assistant Professors, as Project Guide for his ideas and co-operation showed on us during our venture and making this Project as a great success.

We also thankful to **Prof. Kiran M P**, Project Co-Ordinator, Department of Computer Science and Engineering for the Co-Operations and encouragement at all the moments of our approach.

We would also like to thank our parents and well-wishers as well as our dear classmates for their guidance and their kind co-operation.

Finally, it is our pleasure and happiness to the friendly co-operation showed by all the staff members of Computer Science Department, GECH.

Project associates:

MEHAK BANDHRAL (4GH22CS028)

RADHIKA (4GH22CS041)

ROSHNI A (4GH22CS049)

VIDYA S M (4GH22CS063)

ABSTRACT

The main purpose of this project is to build a face recognition-based attendance system to enhance and upgrade the current attendance system into more efficient and effective as compared to before. The current old system has a lot of ambiguity that caused inaccurate and inefficient of attendance taking. Many problems arise when the authority is unable to enforce the regulation that exist in the old system. Thus, by means of technology, this project will resolve the flaws existed in the current system while bringing attendance taking to a whole new level by auto mating most of the tasks. The technology working behind will be the face recognition system. The human face is one of the natural traits that can uniquely identify an individual. Therefore, it is used to trace identity as the possibilities for a face to deviate or being duplicated is low. In this project, face databases will he created to pump data into the recognizer algorithm. Then, during the attendance taking session, faces will be compared against the database to seek for identity. When an individual is identified, its attendance will be taken down by clicking the take image button and saving necessary information into a database system. At the end of the day, the attendance information regarding an individual can be accessed from an Excel sheet.

TABLE OF CONTENT

LIST OF FIGURES

INTRODUCTION

CHAPTER 1

1.1 Project Overview

The main purpose of this project is to build a face recognition-based attendance system to enhance and upgrade the current attendance system into more efficient and effective as compared to before. The current old system has a lot of ambiguity that caused inaccurate and inefficient of attendance taking. Many problems arise when the authority is unable to enforce the regulation that exist in the old system. Thus, by means of technology, this project will resolve the flaws existed in the current system while bringing attendance taking to a whole new level by automating most of the tasks. The technology working behind will be the face recognition system. The human face is one of the natural traits that can uniquely identify an individual. Therefore, it is used to trace identity as the possibilities for a face to deviate or being duplicated is low. In this project, face databases will be created to pump data into the recognizer algorithm. Then, during the attendance taking session, faces will be compared against the Databases to seek for identity. When an individual is identified, its attendance will be taken down by clicking the take button and saving necessary information into a database system. At the end of the day, the attendance information regarding an individual can be accessed from an Excel sheet.

1.1.1 Python:

- Python is a general purpose, dynamic, high-level, and interpreted programming language. It supports Object-Oriented programming approach to develop application. It is simple and easy to learn and provides lots of high-level data structures.
- Python is easy to learn yet powerful and versatile scripting language, which makes it attractive for application development.
- Python’s syntax and dynamic typing with its interpreted nature make it an ideal language for scripting and rapid application development.

- Python supports multiple programming pattern, including object-oriented, imperative, and functional or procedural programming styles.
- Python is not intended to work in a particular area, such as web programming. That is why it is known as multi-purpose programming language because it can be used with web, enterprise, 3D CAD, etc.
- The advantages are enormous. First, it is much easier to program in a high-level language. Programs written in a high-level language takes less time to write, they are shorter and easier to read, and they are more likely to be correct. Second, high-level languages are portable, meaning that they can run on different kinds of computers with few or no modifications. Low-level programs can run on only one kinds of computer and have to be rewritten to run on another.

1.1.2 Import Modules in python

- **OpenCV**

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. Python is a general-purpose programming language started by Guido van Rossum that became very popular very quickly, mainly because of its simplicity and code readability. It enables the programmer to express ideas in fewer lines of code without reducing readability. OpenCV-Python makes use of Numpy, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the OpenCV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

- **Pandas**

In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license.

- **Tkinter**

Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's de facto standard GUI. Tkinter is included with standard Li

1.2 Scope of the project

The Real-Time Face Recognition Attendance System automates attendance using face recognition, eliminating manual roll calls. It captures and encodes faces during registration by processing multiple video frames to ensure accuracy. Once registered, the system continuously scans for faces using the MTCNN model and compares them to stored data. If a match is found, it records the person's name and timestamp in a CSV file, preventing duplicate attendance by checking if they've been marked present already.

The user-friendly Tkinter interface allows easy face registration, live video feed viewing, and real-time feedback, making it simple for administrators and users alike. Future improvements could include multi-camera support, cloud integration for centralized attendance data, a mobile app for user convenience, and enhanced face recognition accuracy using advanced AI models.

This system is suitable for various sectors, including education, corporate environments, events, healthcare, and government, offering automated attendance tracking, reducing administrative workload, and improving efficiency.

1.3 Challenges

The Real-Time Face Recognition Attendance System presents several challenges that need to be addressed for optimal performance. One major issue is lighting conditions. The system's accuracy is sensitive to lighting, and poor or inconsistent lighting can cause missed detections or incorrect identifications. This is especially problematic in environments like classrooms or offices with fluctuating lighting. Solutions such as infrared cameras or improved lighting setups could help, but they would increase system complexity and cost.

Another challenge is accuracy and variability in faces. Changes in appearance due to aging, weight fluctuations, or different hairstyles can impact face recognition. These variations may lead to false negatives or missed identifications. To mitigate this, the system could incorporate continuous learning algorithms or more advanced models to handle such changes more effectively.

Privacy and data security are also critical concerns. Since facial data is sensitive, there is a risk of unauthorized access and privacy violations. Strong data encryption and compliance with regulations like GDPR are necessary to protect this data. Additionally, obtaining clear consent from individuals before capturing their facial images is essential for ethical use.

Lastly, computational power and processing speed are challenges, particularly when processing high-resolution video streams or handling multiple faces. The system may experience lag or slow performance in larger settings. Optimizing algorithms or utilizing cloud computing could help improve processing times, but this introduces potential network issues and added latency.

Overall, while the system offers significant benefits, overcoming these challenges is crucial for its success and widespread adoption. Ensuring accuracy, security, and efficiency will require ongoing improvements in both hardware and software.

1.4 Objectives

The objectives of the Real-Time Face Recognition Attendance System are focused on creating an efficient, accurate, and user-friendly solution for automating the attendance process. Below are the key objectives:

1. **Automate Attendance Marking**: The primary objective is to eliminate manual attendance-taking by using face recognition technology. This allows for seamless, real-time attendance recording based on the identification of individuals.
2. **Enhance Accuracy and Reliability**: Ensure high levels of accuracy in recognizing faces, even in challenging conditions such as varying lighting, face orientation, or appearance changes. The system aims to minimize errors like false positives or missed detections.
3. **Provide Real-Time Recognition**: Implement a real-time face detection and recognition system that can accurately identify individuals from video streams and mark attendance immediately, without delays.
4. **User-Friendly Interface**: Develop a simple, intuitive interface that allows users to easily register faces, monitor real-time attendance, and manage the system. The interface should be accessible to both administrators and participants.
5. **Ensure Data Security and Privacy**: Protect sensitive facial data by implementing robust encryption methods and ensuring compliance with privacy regulations like GDPR. Data should only be accessible to authorized users, and clear consent must be obtained from individuals before their data is processed.

CHAPTER 2

LITERATURE SURVEY

Traditionally attendance was taken manually which is very time consuming and often leads to human error. Additionally, there are many uncertainties towards the sources of the attendance records which in fact, most of the attendance records are not retrieved from the actual situation. The old method that uses paper sheets for taking student's attendance can no longer be used. Based on the research, there are many solutions that are available to solve this issue.

2.1 Face Recognition Based Attendance Marking System

Authors: Senthamil Selvi, Chitrakala, Antony Jenitha

The face recognition-based attendance marking system is based on the identification of face-recognition to solve the previous attendance system's issues. This system uses camera to capture the images of the employee to do face detection and recognition. The captured image is compared one by one with the face database to search for the worker's face where attendance will be marked when a result is found in the face database. The main advantage of this system is where attendance is marked on the server which is highly secure where no one can mark the attendance of other. Moreover, in this proposed system, the face detection algorithm is improved by using the skin classification technique to increase the accuracy of the detection process. Although more efforts are invested in the accuracy of the face detection algorithm, the system is yet not portable. This system requires a standalone computer which will need a constant power supply that makes it not portable. This type of system is only suitable for marking staff's attendance as they only need to report their presence once a day, unlike students which require to report their attendance at every class on a particular day, it will be inconvenient if the attendance marking system is not portable. Thus, to solve this issue, the whole attendance management system can be developed on an embedded design so that it can be work similarly with just batteries that makes it portable.

2.2 RFID based Student Attendance System

Authors: Hussain, Dugar, Deka, Hannan

The RFID technology is used to improve the older attendance system. In this system, a tag and a reader is again used as a method of tracking the attendance of the students. The difference between the first journals with this is where attendance's information can be accessed through a web portal. It provides more convenient for information retrieval. Again, this system is imperfect in the sense that, firstly, it is not portable, as the RFID reader can only work when it is connected to a PC. Secondly, the RFID tag is not a guanine information that can uniquely identify a student, thus, resulting in the inaccuracy of the collected attendance information. In conclusion, a better attendance monitoring system should be developed based on its portability, accessibility and the accuracy of the collected attendance information.

2.3 Face Recognition for Attendance System using Machine Learning

Authors: Shriram, P., and colleagues.

There are various face-based attendance systems available in the market, but most of them either store the data locally or require a paid cloud storage service to maintain the attendance records. However, it is possible to develop a cost-effective attendance system using Python packages such as OpenCV and face-recognition to recognize faces, and storing attendance data in a free cloud storage service such as Google Sheets. By using these efficient and user-friendly Python packages, we can simplify the process of recognizing faces and storing attendance data. Additionally, we can use Google API to integrate Google Sheets with our attendance system, allowing us to maintain attendance records in a free and easily accessible cloud storage service. This proposed attendance system has the potential to improve the efficiency of existing attendance systems and make attendance management more affordable for organizations of all sizes. By leveraging the power of Python packages and free cloud storage, ourselves can simplify the attendance management process and reduce the cost of maintaining attendance records.

2.4 Automatic Student Attendance System Based on Face Recognition Using MTCNN

Authors: Ms. S. D. Salunkhe, Prof. Dr. S. S. Patil

An intelligent method to attendance management is the use of facial recognition technology for attendance tracking. Compared to earlier methods, facial recognition is faster and more accurate, which reduces the possibility of attendance or proxy fraud. Additionally, facial recognition offers a non-intrusive method of identification in which the subject of the identification does not need to actively confirm their identity. To do this, we use the MTCNN technique to first extract features and detect faces, and then we recognize faces. Hence MTCNN is a deep cascaded multi-task framework that increases performance by taking advantage of the correlation that exists naturally between alignment and detection.

CHAPTER 3

REQUIREMENT SPECIFICATION

3.1 Functional Requirements

1. **User Registration**:
    - Users enter their name and look at the camera.
    - Faces are detected using MTCNN, then encoded with the face_recognition library.
    - Multiple frames are captured to ensure accurate encoding.
2. **Attendance Marking**:
    - The system detects faces in the live video feed and matches them with stored encodings.
    - When a match is found, the user's name and timestamp are saved in a CSV file.
    - Unrecognized faces are labelled as "Unknown."
3. **User Interface**:
    - Basic console and video feed interface for user registration and face recognition.
    - Displays detected faces, recognized names, and updates attendance in real-time.
4. **Exit Mechanism**:
    - The system exits when the 'exit' key is pressed.
    - Video capture stops, and any changes to the attendance data are saved before exiting.

3.2Non-Functional Requirements

1. **Performance:**
   - The system should be able to process video input in real-time and detect faces without significant lag.
   - The recognition process should be accurate enough to minimize false positives (incorrect recognition of faces).
2. **Security:**
   - Facial data (encodings) should be stored securely, ensuring that personal information is not accessible or misused.
3. **Usability:**
   - The system should be easy to use, with simple console prompts and real-time visual feedback through the video feed.
   - New users should be able to register their faces with minimal interaction and clear instructions.
4. **Reliability:**
   - The system should reliably recognize registered users under different lighting conditions and slight variations in facial appearance (e.g., glasses or facial hair).
5. **Maintainability:**
   - The code should be modular, with separate functions for registration and attendance marking, allowing for easy updates and modifications.
   - Attendance records should be stored in an easily accessible format (CSV file), which can be reviewed or transferred to other systems.
6. **Portability:**
   - The system should run on any machine with a webcam, Python, and the required libraries installed (OpenCV, MTCNN, face_recognition, pandas).

3.3 System Requirements

3.3.1 Hardware Requirements

- CPU: Minimum Intel Core i5, Recommended i7.
- RAM: Minimum 8 GB, Recommended 16 GB.
- Storage: 500 MB minimum.
- Webcam: 720p or higher.
- GPU: Optional (Integrated), Recommended (NVIDIA GTX 1050 or higher).

3.3.2 Software Requirements

- OS: Windows 10/11.
- Python: Version 3.7+, with Pip for package management.
- Libraries: OpenCV, MTCNN, face_recognition, Pandas.

3.3.3 Environmental Requirements

- Lighting: Well-lit, uniform lighting.
- Distance: 30-70 cm from camera.
- Background: Neutral and clutter-free.

3.3.4. Network & Storage Considerations

- Network: Optional (for cloud backups); system works offline.
- Storage: Local storage of attendance records in CSV format (minimal space required).

CHAPTER 4

SYSTEM ARCHITECTURE AND DESIGN

4.1 System Overview

The Face Recognition Attendance System is an innovative solution that leverages facial recognition technology to automate the process of attendance tracking. By using a webcam, the system captures live video feeds, processes them to detect and recognize faces, and marks attendance based on the identity of the detected individuals. It employs the MTCNN (Multi-task Cascaded Convolutional Networks) algorithm for detecting faces and the face-recognition library for recognizing and encoding facial features, ensuring high accuracy and efficiency in face detection and recognition.

Once a user registers by providing their name and looking at the camera, the system captures their facial data and stores it as an encoding for future comparison. This allows the system to recognize faces in real-time and match them against a database of known individuals. The system can handle multiple faces simultaneously, ensuring that it can recognize all individuals present in front of the camera at any given moment.

The attendance marking process is seamless and happens automatically in the background. When a face is detected and successfully matched to a registered individual, the system logs the person's name and the timestamp of the recognition in a CSV file. This feature eliminates the need for manual attendance-taking, reducing errors and time spent on administrative tasks. The system's use of Tkinter, a Python GUI library, makes it user-friendly, with buttons for face registration, starting the recognition process, and exiting the application.

Real-time performance is a key feature of this system, ensuring it is suitable for environments where efficiency is important, such as schools, corporate offices, or other organizations. The system runs smoothly and responds quickly, providing an effective solution for automating attendance tracking while ensuring security and ease of use.

4.2 Hardware Architecture

The hardware architecture of the Face Recognition Attendance System is designed to operate on a typical computer system with the necessary components to capture video input, process the data, and display the results. The primary hardware components involved are:

- Webcam/Camera: Captures live video for face detection and recognition. A 720p resolution is recommended for optimal performance.
- Processing Unit (Computer/Server): Runs the software, including face recognition algorithms and the user interface. A modern multi-core processor with at least 4GB of RAM is recommended.
- Display (Monitor): Shows the live video feed and user interface for registration and recognition feedback.
- Data Storage: Stores face data and attendance logs in a CSV file, either locally (e.g., HDD or SSD) or remotely via cloud storage for scalability and remote access.

4.3 Software Architecture

The system is structured into key layers to handle face detection, recognition, and attendance logging using Python libraries:

1. The User Interface (UI) Layer, built with Tkinter, provides an intuitive interface for face registration, starting face recognition, and exiting the app. It displays a real-time webcam feed with face detection results, ensuring an easy user experience.
2. The Face Detection & Recognition Layer uses MTCNN to detect faces in video frames and provide their coordinates. Face-recognition then encodes the faces and compares them with stored encodings to identify users.
3. The Backend Logic handles face registration by capturing face frames, creating encodings, and storing them. For attendance marking, it compares live face encodings with stored ones and logs the attendance in a CSV file.
4. Data Storage uses a Pandas Data Frame to manage attendance data, which is then saved to a CSV file (attendance.csv).

4.4 Data Flow

1. Start System:
    - The user launches the system, starting the webcam and displaying the video feed in the Tkinter window.

2. Face Registration:
    - Input: The user provides their name.
    - Process: The system captures multiple frames, detects faces with MTCNN, and encodes the face using face_recognition.
    - Output: The user's face encoding is saved in known_faces, and their name is stored in known_names.

3. Face Recognition:
    - Input: The webcam captures frames in real-time.
    - Process: MTCNN detects faces, and face_recognition compares them with stored encodings.
    - Output: If a match is found, the system records the person's attendance in face_data, which is saved to attendance.csv.

4. Display Feedback:
    - Process: Recognized faces are highlighted with a rectangle and labelled
    - with the person's name.
    - Output: The real-time video is shown in the Tkinter window with recognized faces.

5. Exit:
    - Input: The user clicks the "Exit" button.
    - Process: The system releases the webcam and closes the application.

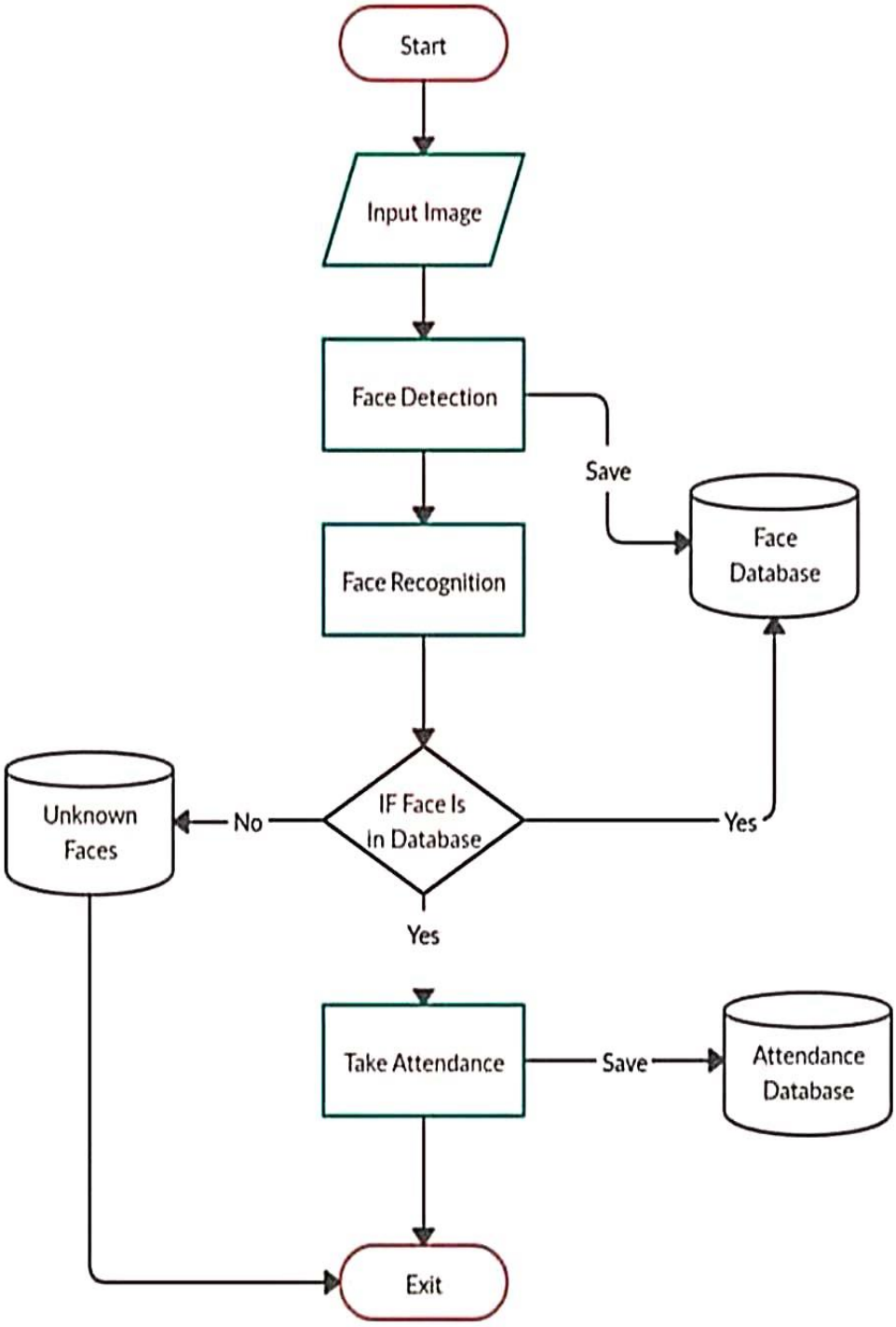

Fig 3.4: Flow chart of the Face Recognition Based Attendance System

4.5 Security Considerations

1. **Data Privacy:**
    - **Encryption**: Encrypt facial data both during storage and transmission.
    - **User Consent**: Obtain explicit consent from users before collecting facial Data.
2. **Access Control:**
    - **Multi-Factor Authentication (MFA)**: For admin access to the system.
    - **Role-Based Access**: Restrict sensitive data access based on user roles.
3. **Data Integrity:**
    - **Audit Logs**: Keep logs of data access and modifications.
    - **File Integrity**: Use file hashing to detect unauthorized changes.
4. **Face Recognition Accuracy:**
    - **Anti-Spoofing**: Implement liveness detection to prevent spoofing.
    - **Regular Model Updates**: Periodically update and test the recognition system.
5. **Physical Security:**
    - **Secure Devices**: Limit physical access to cameras and servers.
    - **Camera Placement**: Position cameras to respect privacy and security.
6. **Backup & Recovery:**
    - **Data Backups**: Regularly back up data to prevent loss.
    - **Disaster Recovery**: Have a recovery plan in case of failures.

4.6 Scalability and Flexibility

Scalability refers to the system's ability to handle increasing numbers of users and data without performance issues. A scalable face recognition system can expand by adding more processing power or cloud resources as user demand grows. Distributed storage and parallel processing help manage large datasets efficiently. Additionally, using load balancing techniques ensures that multiple requests are evenly distributed, maintaining smooth operation even under heavy traffic. Cloud-based infrastructure allows the system to dynamically scale according to demand.

Flexibility refers to the system’s ability to adapt to changing needs and integrate with other technologies. A flexible face recognition system can support various authentication methods (e.g., fingerprint or PIN) and integrate with third-party applications like HR or attendance systems. It can also adjust to different environments, such as varying lighting or angles, by retraining models as needed. This adaptability ensures that the system remains functional and relevant as the use cases and operational conditions evolved.

CHAPTER 5

IMPLEMENTATION

5.1 Setup and Configurations

1. Environment Setup:

- Download Python: Go to the official Python website and download Python 3.8 and verify the installation using the command python –version
- Install Jupyter Notebook for interactive development using command, pip install notebook and launch it.

2. Library Installation:

- OpenCV: pip install opencv-python
- face_recognition: pip install face_recognition
- pandas: pip install pandas
- mtcnn: pip install mtcnn
- dlib: pip install cmake dlib

Run the above commands in the Jupyter Notebook to install the required Python libraries.

3. Configuration Steps:

- Verify Webcam Accessibility via OpenCV.
- Set current working directory and verify the directory.
- CSV File: The system automatically generates attendance.csv in the working directory.

5.2 System Workflow

1. **Face Registration:**
    - A user can register their face by looking at the camera, during which face encodings are captured and saved. The face encoding are stored in the known_faces list, and the corresponding names are stored in the know_names list.
    - The registration process allows capturing multiple frames to improve accuracy in detecting the user's face.

2. **Attendance Tracking:**
    - The system continuously captures frames from the video feed, detects faces using MTCNN, and matches the detected faces with previously registered faces.
    - If a match is found,the attendance for that individual is recorded along with the timestamp in a CSV file (attendance.csv).
    - A new face is registered dynamically, and attendance is marked only once per detected person.

3. **CSV File for Attendance:**
    - The attendance data is saved to a CSV file, where each entry includes the name of the recognized person and the timestamp when they were detected.

4. **User Interface:**
    - The user can choose to register a face or exit the registration mode via a simple console-based menu.
    - During face recognition, detected faces are drawn with a rectangle, and the name of the individual is displayed on the frame.

5. **Testing and Debugging:**
    - Test every application thoroughly to ensure all functionalities work correctly.
    - Debug any issues related to face registration, attendance tracking and user interactions.

5.3 Code Walkthrough

```
import cv2
import face_recognition
import pandas as pd
from datetime import datetime
from mtcnn import MTCNN
import os
import tkinter as tk
from tkinter import simpledialog, messagebox
from PIL import Image, ImageTk
```

Fig : Importing libraries for Face Recognition and GUI Development.

```
# Initialize video capture
video_capture = cv2.VideoCapture(0)

# Initialize lists for storing known face encodings and names
known_faces = []
known_names = []
face_data = pd.DataFrame(columns=["Name", "Time"])

# Initialize MTCNN detector
detector = MTCNN()

# Function to mark face in the attendance
def mark_face(name):
    current_time = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
    print(f"Marking attendance for {name} at {current_time}")  # Debugging line
    face_data.loc[len(face_data)] = [name, current_time]
    face_data.to_csv("attendance.csv", index=False)
    print(f"Attendance marked for {name} at {current_time}")
```

Fig : Initializing and Attendance Marking.

```
# Function to register a new face (capturing and storing face encoding)
def register_face(name):
    print(f"Please look at the camera for registration, {name}...")
    for _ in range(5):  # Capture multiple frames for better face encoding
        ret, frame = video_capture.read()
        if not ret:
            print("Failed to grab frame")
            continue

        # Detect faces using MTCNN
        results = detector.detect_faces(frame)

        if results:
            # Assume we are working with the first detected face
            x, y, w, h = results[0]['box']
            face = frame[y:y+h, x:x+w]

            # Convert the frame to RGB (OpenCV uses BGR)
            rgb_frame = cv2.cvtColor(face, cv2.COLOR_BGR2RGB)
            face_encoding = face_recognition.face_encodings(rgb_frame)

            if face_encoding:
                encoding = face_encoding[0]
                known_faces.append(encoding)
                known_names.append(name)
                print(f"{name} has been registered successfully!")
                break
        else:
            print("No face detected, please try again.")
```

Fig: Face Recognition and Encoding for Attendance System.

```
# Function to handle face recognition and attendance marking
def start_face_recognition():
    recognized_faces = []

    while True:
        ret, frame = video_capture.read()
        if not ret:
            break

        # Detect faces using MTCNN
        results = detector.detect_faces(frame)

        for result in results:
            x, y, w, h = result['box']
            face = frame[y:y+h, x:x+w]

            # Convert the frame to RGB (OpenCV uses BGR)
            rgb_frame = cv2.cvtColor(face, cv2.COLOR_BGR2RGB)

            # Find face encodings
            face_encodings = face_recognition.face_encodings(rgb_frame)

            if face_encodings:
                face_encoding = face_encodings[0]
                matches = face_recognition.compare_faces(known_faces, face_encoding)
                name = "Unknown"

                if True in matches:
                    first_match_index = matches.index(True)
                    name = known_names[first_match_index]

                    # Mark face if not already marked in this frame
                    if name not in recognized_faces:
                        if name not in face_data['Name'].values:
                            mark_face(name)
                        recognized_faces.append(name)

             # Draw a rectangle around the face with green color (BGR: (0, 255, 0))
             cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)

             # Display the name below the face
             cv2.putText(frame, name, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 0), 2)

       # Convert frame to ImageTk format for displaying in Tkinter
       rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
       img = Image.fromarray(rgb_frame)
       imgtk = ImageTk.PhotoImage(image=img)

       # Update the GUI window with the new frame
       video_label.imgtk = imgtk
       video_label.configure(image=imgtk)

       # Add a delay to simulate real-time video
       root.update_idletasks()
       root.update()
```

Fig : Functions to handle Face Recognition and Attendance Marking.

```
# Function to handle registration mode
def register_face_gui():
    name = simpledialog.askstring("Register Face", "Enter the name to register:")
    if name:
        register_face(name)
        messagebox.showinfo("Registration", f"{name} has been registered!")

# Function to exit the application
def exit_application():
    video_capture.release()  # Release the video capture object
    cv2.destroyAllWindows()  # Close any OpenCV windows
    root.quit()  # Exit the Tkinter event loop and close the window

# Initialize Tkinter window
root = tk.Tk()
root.title("Face Recognition Attendance System")

# Set the window background color
root.configure(bg='#f0f0f0')

# Add a label to show the video stream
video_label = tk.Label(root, bg='#f0f0f0')
video_label.pack(padx=10, pady=10)

# Add some space between components
frame = tk.Frame(root, bg='#f0f0f0')
frame.pack(pady=20)

# Add buttons for registering a face and starting face recognition
register_button = tk.Button(frame, text="Register Face", command=register_face_gui,
                            font=("Arial", 12, "bold"), bg="#4CAF50", fg="white", padx=20, pady=10)
register_button.pack(side=tk.LEFT, padx=10)
```

```
# Add buttons for registering a face and starting face recognition
register_button = tk.Button(frame, text="Register Face", command=register_face_gui,
                            font=("Arial", 12, "bold"), bg="#4CAF50", fg="white", padx=20, pady=10)
register_button.pack(side=tk.LEFT, padx=10)

start_button = tk.Button(frame, text="Start Face Recognition", command=start_face_recognition,
                         font=("Arial", 12, "bold"), bg="#2196F3", fg="white", padx=20, pady=10)
start_button.pack(side=tk.LEFT, padx=10)

exit_button = tk.Button(frame, text="Exit", command=exit_application,
                        font=("Arial", 12, "bold"), bg="#f44336", fg="white", padx=20, pady=10)
exit_button.pack(side=tk.LEFT, padx=10)

# Set a stylish header label
header_label = tk.Label(root, text="Face Recognition Attendance System", font=("Arial", 18, "bold"), bg="#f0f0f0", fg="#333")
header_label.pack(pady=10)

# Start the Tkinter event loop
root.mainloop()

# Release the camera when the window is closed
video_capture.release()
cv2.destroyAllWindows()
```

Fig: Functions to handle Registration mode and GUI implementation for Face Registration and Attendance System.

5.4 Maintenance Plan:

1. **Performance Monitoring:**

   - Regularly test the system to ensure it handles real-time video capture and face recognition efficiently.
   - Optimize code if lag or delays are observed, especially for classrooms with a large number of students.

2. **Database Backup:**

   - Regularly back up the attendance data (attendance.csv).

3. **Updating Dependencies:**

   - Periodically update Python libraries to incorporate security patches and new features

4. **System security:**

   - Protect attendance data by encrypting sensitive files.

5. **Hardware Maintenance:**

   - Ensure the webcam is in good working condition and periodically clean it for optimal image quality.
   - Replace hardware components (eg: webcams) as required.

6. **User Training:**

   - Provide training to users on how to operate the system effectively.
   - Offer troubleshooting guides for common issues like webcam detection or system errors.

CHAPTER 6

RESULTS

The following Snapshots shows the efficient Running of the face recognition based attendance system

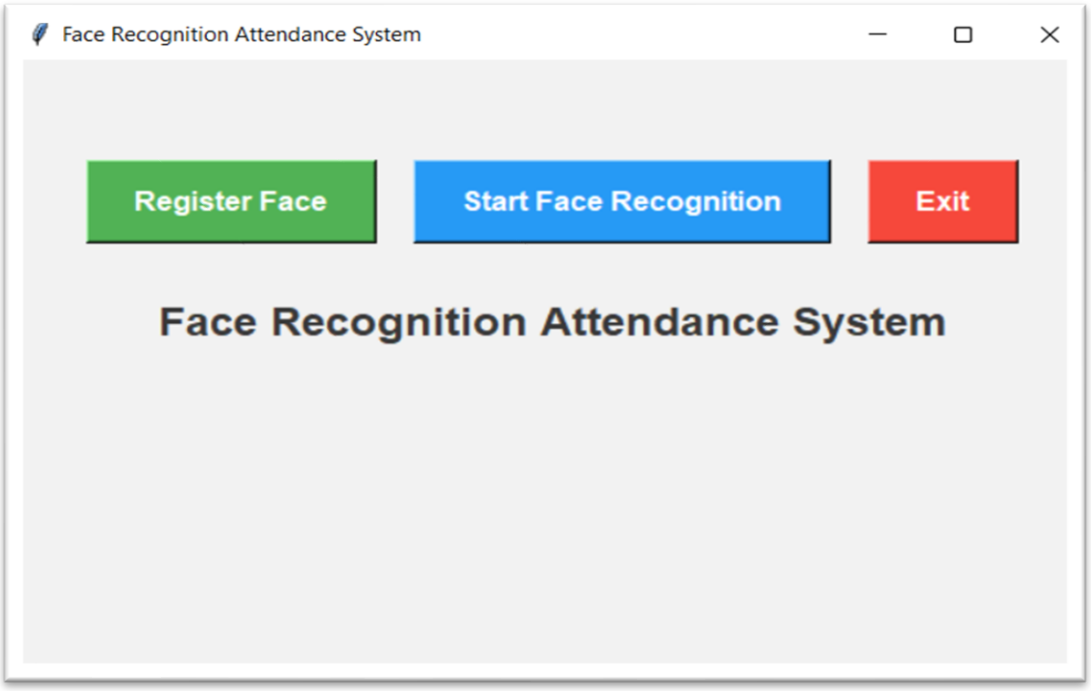

Fig 6.1 : Tkinter Window : Face Recognition Attendance System.

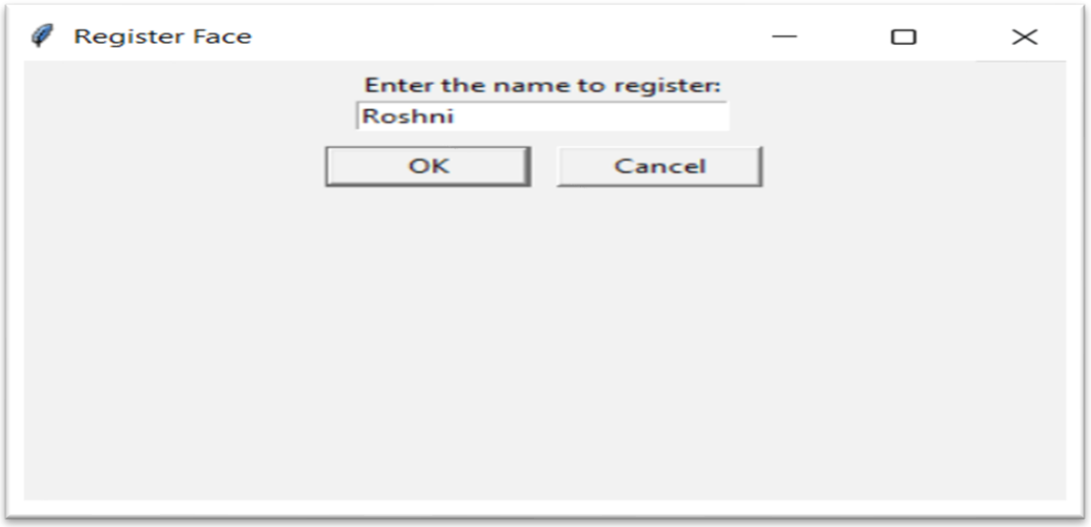

Fig 6.2: Registration Window.

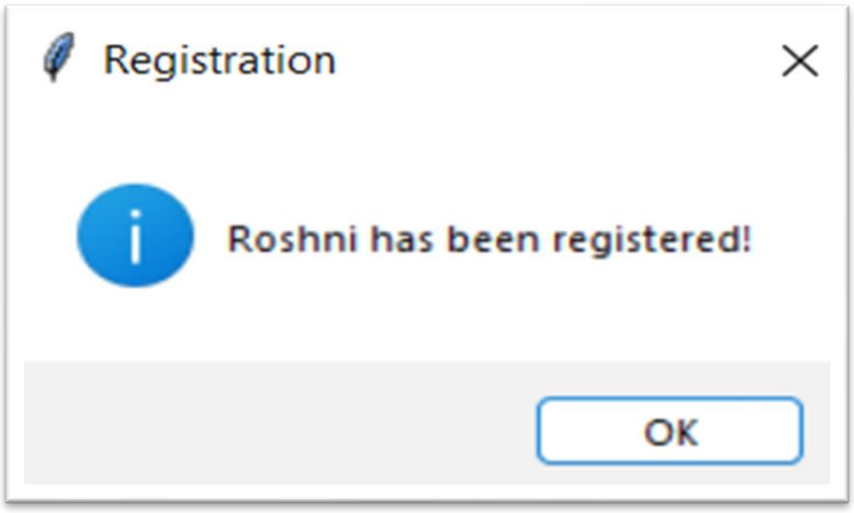

Fig 6.3 : Confirmation of Registered face.

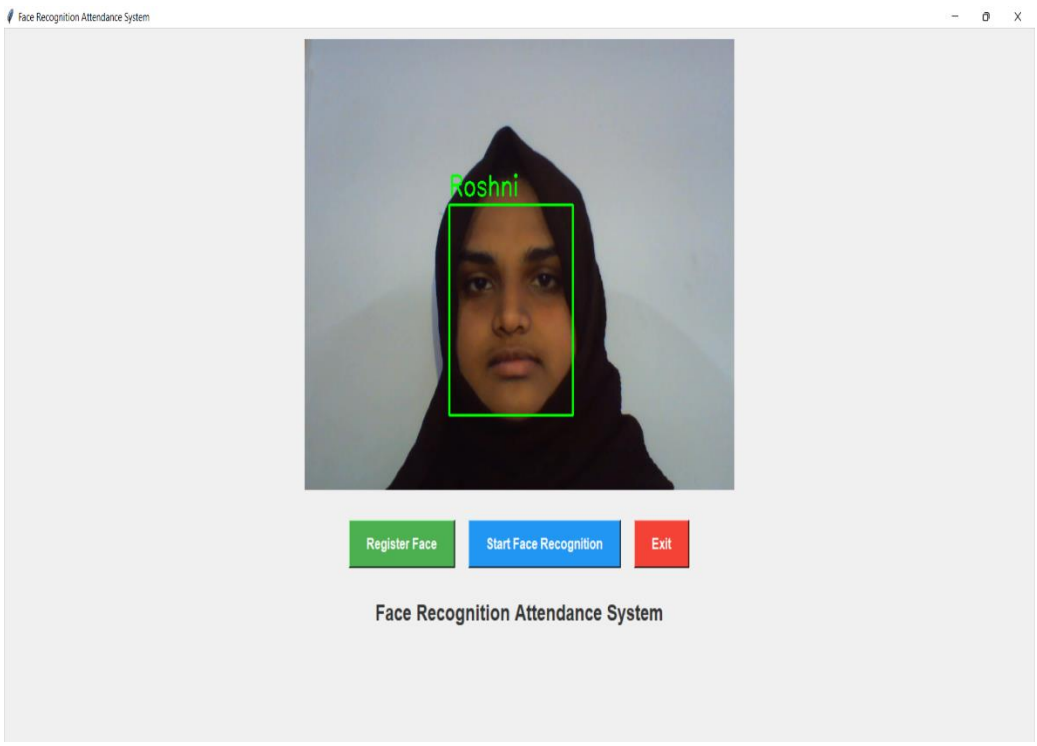

Fig: Recognition of Face 1.

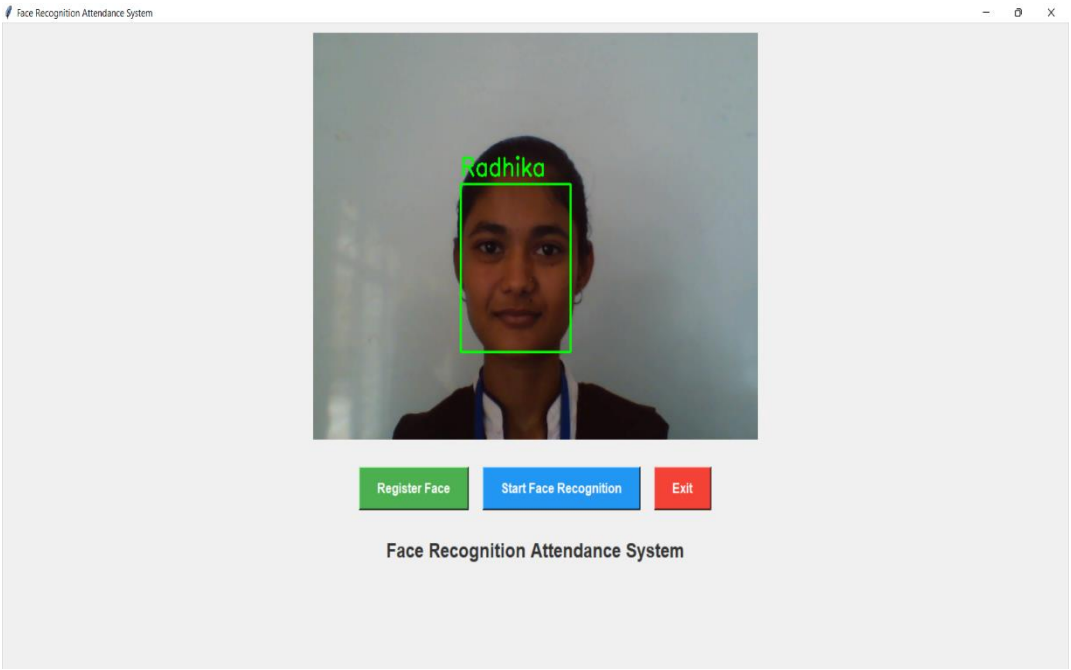

Fig: Recognition of Face 2.

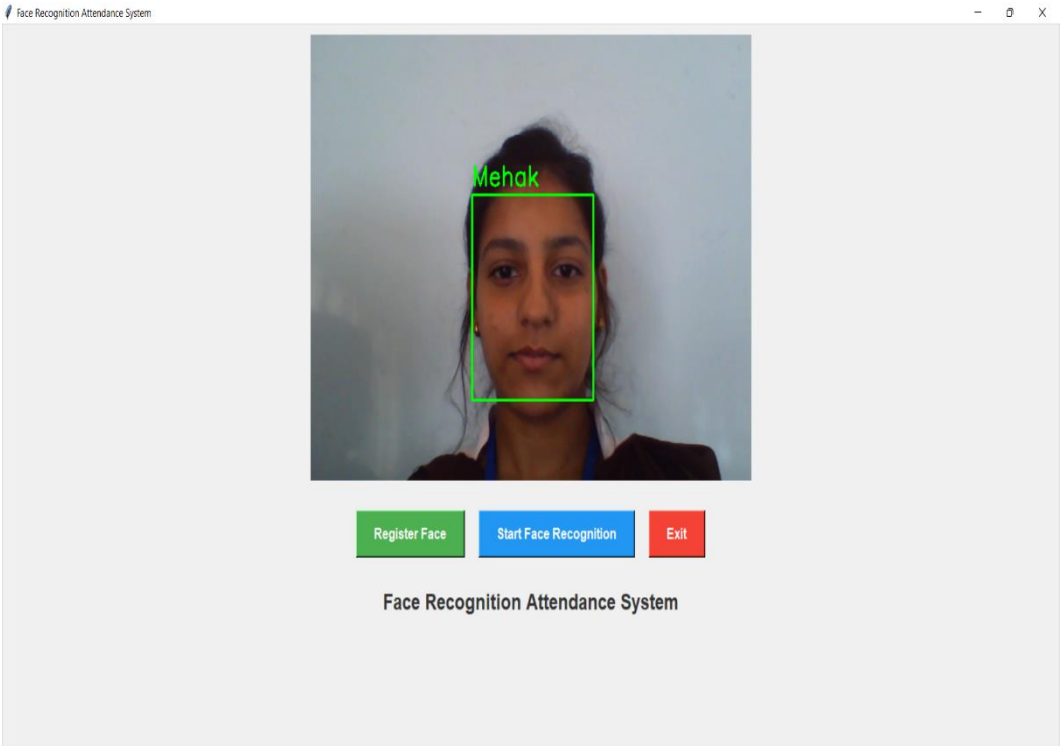

Fig: Recognition of Face 3.

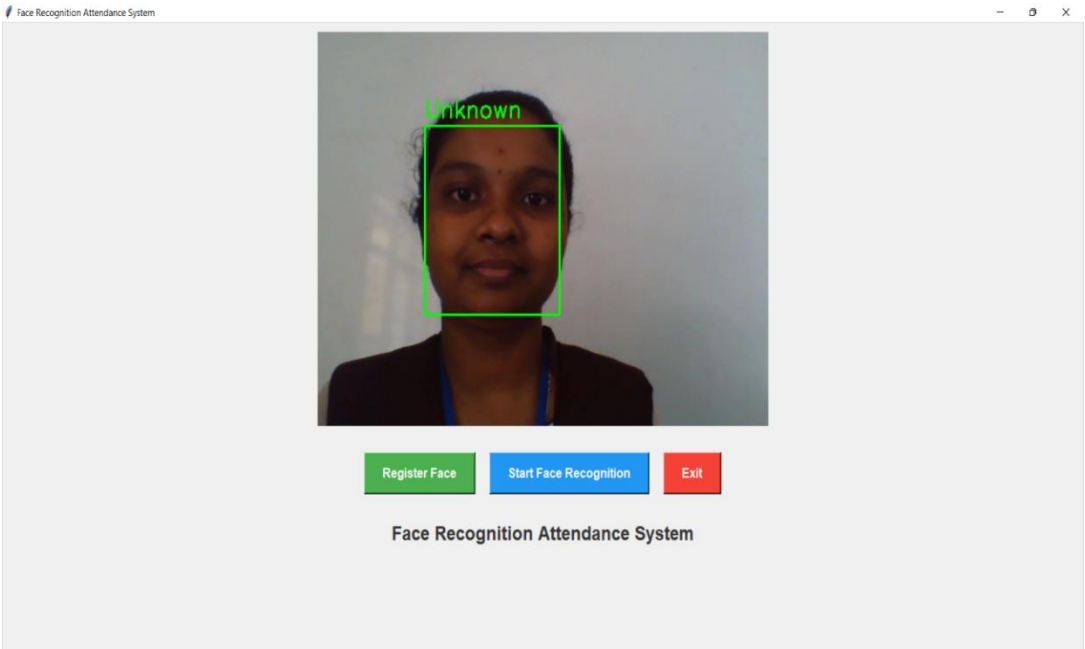

Fig: Unknown Face detected.

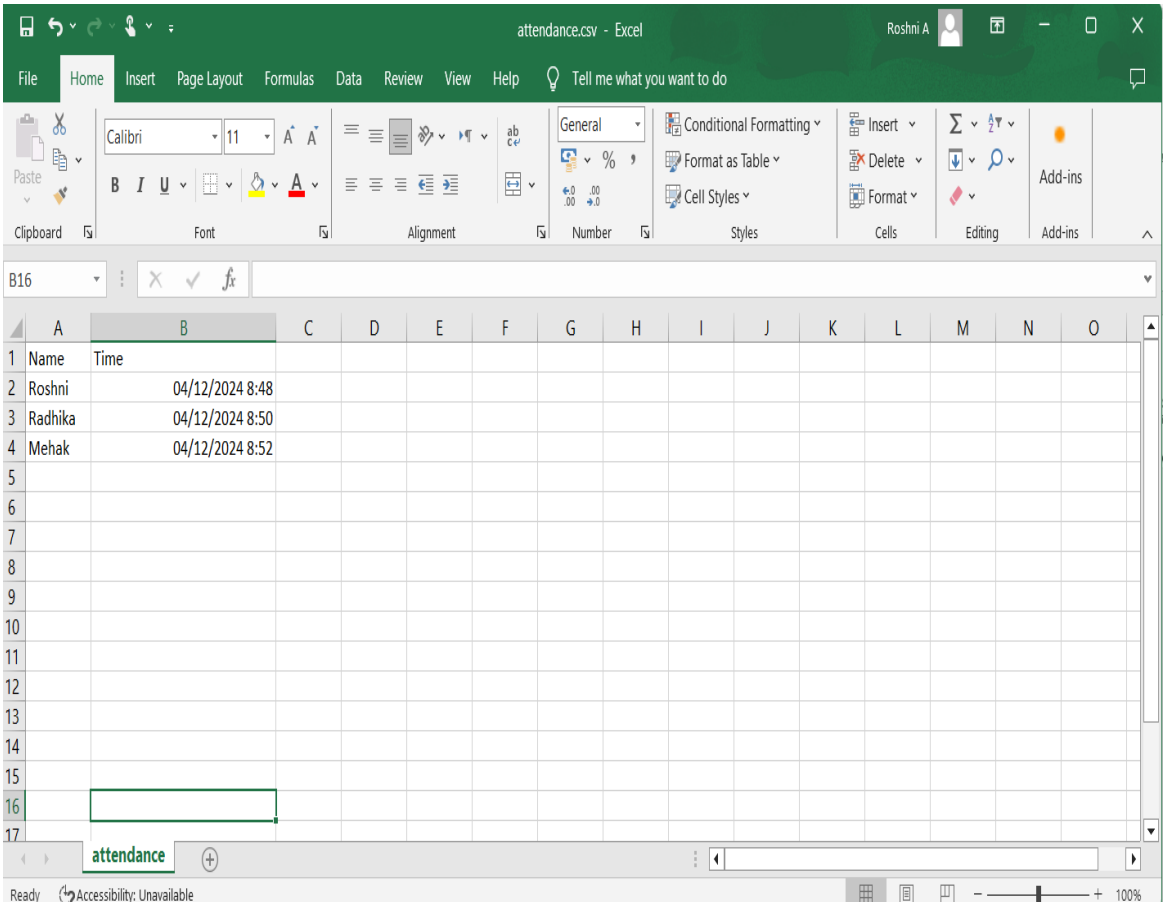

Fig: Attendance.csv file in Excel sheet

CONCLUSION

The "Face Recognition Attendance System" provides an efficient, modern approach to automating attendance tracking using facial recognition. By integrating advanced technologies like MTCNN for face detection, OpenCV for video handling, and the face_recognition library for accurate face matching, the system delivers reliable real-time recognition and attendance marking. This eliminates manual processes and reduces errors, offering a streamlined solution for schools, offices, and other environments.

The user-friendly Tkinter interface allows seamless face registration and recognition, while attendance data is securely logged in a CSV file for easy access. The system not only simplifies attendance management but also enhances accuracy and transparency.

This project demonstrates the practical application of AI and computer vision in everyday tasks, offering a scalable and adaptable solution. With future enhancements such as improved recognition in challenging conditions or deeper integration with cloud systems, this face recognition attendance system has the potential to further transform traditional methods of attendance tracking across various industries.

REFERENCES

[1] Zhang, K., Zhang, Z., & Li, Z. (2016). "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks." IEEE Signal Processing Letters.

[2] Face recognition-based attendance system using machine learning with location identification (2023) by Naveen Raj M., R. Vadivel.

[3] Face Recognition Based Attendance Marking System? (2014) by K.Senthamil Selvi P.Chitrakala A.Antony Jenitha.

[4] RFID based Student Attendance System (2014) by Elima Hussain, Priyanka Dugar, Vaskar Deka, Abdul Hannan.

[5] Szeliski, R. (2010). Computer Vision: Algorithms and Applications. Springer Science & Business Media. This book provides a detailed overview of computer vision algorithms, including face detection, and the role of hardware in real-time applications.

[6] Patil, M., & Kharat, S. (2017). Automated attendance system using face recognition. International Research Journal of Engineering and Technology (IRJET).

[7] https://chatgpt.com/