Abstract

The Face Recognition Attendance System described utilizes a combination of Python libraries to automate attendance tracking through facial recognition technology. The core libraries employed include OpenCV (cv2) for video capture and image processing, face_recognition for face detection and encoding, and Streamlit (st) for building a user-friendly web interface. Additionally, other essential libraries such as os, numpy (np), and datetime are utilized for file operations, numerical computations, and timestamping attendance records, respectively. Initialization of the system involves loading images of known faces from a specified directory Each image undergoes facial encoding using the face_recognition library, producing numerical representations of the faces. Concurrently, the system extracts class names, presumably corresponding to student names, from the file names. During recognition and attendance marking, the main function captures video frames from the webcam in real-time. These frames are resized and converted to RGB format to ensure compatibility with the face recognition library. Faces within the frames are detected using the face_recognition.face_locations() function, and their corresponding facial encodings are generated. These encodings are then compared with the encoded faces of known individuals. Upon finding a match, the system retrieves the corresponding name and displays it on the frame. Furthermore, the attendance record is updated by appending the student's name, current time, and date to a CSV file. The user interface, powered by Streamlit, provides an interactive platform for monitoring attendance in real-time. The webcam feed is displayed alongside rectangles drawn around recognized faces and their associated names. This interface allows users to observe recognition results as they occur, facilitating efficient attendance tracking. Attendance logging is managed by storing records in CSV files, with each file named based on the current date If a student is recognized for the first time during a session, their attendance is marked by appending their name, timestamp, and date to the CSV file.

Table of Contents

List of Figures

List of Tables

CHAPTER 1
INTRODUCTION

The integration of computer vision and artificial intelligence has given rise to novel solutions across various domains. One such application that stands at the intersection of efficiency and security is the Face Recognition Attendance Management System. This cutting-edge solution harnesses the power of OpenCV (Open Source Computer Vision Library) alongside sophisticated face recognition algorithms to revolutionize traditional attendance tracking methods.

Traditional attendance management systems often rely on manual processes, such as paper-based registers or card-swiping systems, which can be error-prone, time-consuming, and lack robust security features. The Face Recognition Attendance Management System offers a contemporary alternative, ensuring not only accurate attendance tracking but also the streamlining of administrative processes

At the heart of this system lies OpenCV, an open-source library renowned for its extensive tools and functions in the field of computer vision. OpenCV facilitates the capture, processing, and analysis

of facial data, making it an ideal choice for implementing robust face recognition systems. Its pre-trained deep learning models enable accurate face detection within images and video streams, forming the foundational step for subsequent recognition processes.

The system's workflow involves a multi-step process. Initially, it employs OpenCV to detect faces accurately, utilizing pre-trained deep learning models that can identify and locate faces within a given frame. Subsequently, the system extracts unique facial features, often referred to as facial landmarks or keypoints, creating a distinctive representation for each face. These features serve as the basis for accurate identification and discrimination between different individuals.

OpenCV supports various face recognition algorithms, such as Eigenfaces, Fisherfaces, and Local Binary Pattern Histograms (LBPH). These algorithms analyze facial features to create a recognizable pattern, enabling the system to identify and match faces against a predefined database. In real-time, attendance records are updated and securely stored in a database, providing administrators with a comprehensive and easily accessible overview of attendance records.

1.1 Problem Statement

Traditional attendance systems face issues of inefficiency, inaccuracy, and security concerns. The need for a modern solution is amplified by the limitations of manual methods and a growing emphasis on contactless interactions. Developing a Face Recognition Attendance Management System using OpenCV aims to overcome these challenges, offering a technologically advanced, accurate, and secure alternative that aligns with contemporary requirements in educational and organizational settings.

1.2 Objectives

i. To accurately recognize the faces of the students.

ii. To annotate the recognized faces of the students.

iii. To generate a CSV file containing the names of the students whose attendance has been marked along with the date and time at which the attendance has been captured

CHAPTER 2
LITERATURE SURVEY

The literature survey pertaining to the Face Recognition Attendance Management System using OpenCV explores existing research and developments in the fields of computer vision, facial recognition technology, and attendance management systems. The survey encompasses a review of relevant academic papers, articles, and projects to identify the current state-of-the-art methodologies, challenges, and advancements in the domain.

2.1 Existing system

S. Chintalapati and M. V. Raghunadh Explains about the automated attendance management system. This system, which is based on face detection and recognition algorithms, automatically detects the student when he enters the class room and marks the attendance by recognizing him. The system architecture and algorithms used in each stage are described. Different real time scenarios are considered to evaluate the performance of various face recognition systems. Also proposes the techniques to be used in order to handle the threats like spoofing. When compared to traditional attendance marking this system saves the time and also helps to monitor the students [1].

Kasotiya , Ritik, Raj, Sagar, Akhtar, Sohail illustrate the number of users based on face recognition, the recognition of the key image is used in a number of areas. The purpose of the organization is to show the facial features of the applicant's identity. On the face of the procedure is, in principle, a two-stage, more accurate, face recognition, and face recognition is a system used to manage the number of users who are able to carry out day to day activities, labeling, and analyzing involvement with a minimum of human intervention. User is not required to wait in queue as attendance of multiple people can be recorded at once by simultaneous recognition of faces of multiple people. According to current scenario of covid-19 we proposed this system for preventing infection due to contact and to maintain social distancing. In this project, by using OpenCV library facial recognition based attendance management system is developed the attendance results will be stored in the MySQL database[2].

O. A. R. Salim, R. F. Olanrewaju and W. A. Balogun illustrate the The internet of things, image processing, and machine learning are evolving day by day. Many systems have been completely changed due to this evolve to achieve more accurate results. The attendance system is a typical example of this transition, starting from the traditional signature on a paper sheet to face recognition. This paper proposes a method of developing a comprehensive embedded class attendance system using facial recognition with controlling the door access. The system is based on Raspberry Pi that runs Raspbian (Linux) Operating System installed on micro SD card. The Raspberry Pi Camera, as well as a 5-inch screen, are connected to the Raspberry Pi. By facing the camera, the camera will capture the image then pass it to the Raspberry Pi which is programmed to handle the face recognition by implementing the Local Binary Patterns algorithm LBPs. If the student's input image matches with the trained dataset image the prototype door will open using Servo Motor, then the attendance results will be stored in the MySQL database. The database is connected to Attendance Management System(AMS) web server, which makes the attendance results reachable to any online connected web browser. The system has 95{\% accuracy with the dataset of 11 person images[3].

2.2 Proposed **System**

The purpose of the provided code is to create a Face Recognition Attendance System. It aims to automate the process of marking attendance by utilizing facial recognition technology. The code loads images of known faces, encodes them for recognition, and then captures live video from the webcam. It detects faces in each frame, compares them with the known faces, and if a match is found, it marks the attendance by recording the student's name, timestamp, and date in a CSV file. The system provides a user-friendly interface using the Streamlit framework, allowing real-time monitoring of attendance with recognized faces displayed on the screen. Overall, the purpose is to streamline attendance tracking in educational or organizational settings through the efficient use of facial recognition technology.

CHAPTER 3
SOFTWARE REQUIREMENT SPECIFICATIONS

The Face Recognition Attendance Management System is a software application designed to automate attendance tracking using facial recognition technology. The system utilizes computer vision libraries such as OpenCV and face_recognition to achieve real-time face detection, recognition, and attendance logging.

3.1 Functional Requirements:

Image Database Management: The system must be able to read facial images from a specified directory. It should process and store these images for training the face recognition model.

Face Recognition: The system must use the face_recognition library to extract facial encodings from input images. It should employ pre-trained models for accurate face detection and recognition in real-time webcam frames.

Attendance Logging: The system must log attendance in a structured format (CSV file) with entries including the recognized person's name, timestamp, and date. Duplicate entries for the same person during a single session should be avoided.

User Interface: The system should display the webcam feed with bounding boxes around recognized faces. It must provide a clear visual indication of recognized individuals and update attendance in real-time.

3.2 User Interface Functionalities Requirements

Real-Time Webcam Feed: The interface should display the live video feed captured from the webcam, allowing users to see themselves and any recognized faces in real-time.

Recognition Feedback: Recognized faces should be highlighted within the webcam feed, preferably with rectangles drawn around them. Additionally, the corresponding names of recognized individuals should be displayed adjacent to their faces.

Attendance Information: The interface should provide a section to display attendance-related information, such as the names of recognized individuals, along with the date and time of their attendance.

Clear and Intuitive Layout: The layout should be designed to be easy to understand and navigate, with clear labeling of different sections and functionalities.

Responsive Design: The interface should be responsive to different screen sizes and resolutions, ensuring an optimal viewing experience across various devices.

User Interaction: Users should have the ability to interact with the interface, such as starting and stopping the attendance monitoring process, accessing settings or additional information, and exporting attendance records if necessary.

Feedback and Error Handling: The interface should provide feedback to users about the status of recognition processes, such as indicating when a face is successfully recognized or alerting users if there are errors or issues encountered during the process

Accessibility: The interface should be designed with accessibility considerations in mind, ensuring that it is usable by individuals with different abilities and needs.

Security: If applicable, the interface should incorporate security measures to protect sensitive data, such as attendance records, and ensure that only authorized users can access certain functionalities or information.

Customization Options: Users may require customization options such as adjusting webcam settings, selecting different recognition algorithms or thresholds, or specifying which individuals are considered for attendance tracking.

3.3 Non-Functional Requirements:

Performance: The system should process webcam frames in real-time, ensuring minimal latency for face recognition. It should handle a moderate number of faces in the frame without significant performance degradation.

Reliability: The face recognition algorithm must be reliable and accurate, minimizing false positives and false negatives. The system should maintain recognized students within a session to prevent redundant attendance entries.

Security: The system must securely store attendance records, and access to these records should be restricted. Facial data should be handled in compliance with privacy and data protection regulations.

3.3 Tools and Technologies

3.3.1 Tools

Python: Python is a versatile, high-level programming language known for its readability and ease of use. It is widely adopted in the field of artificial intelligence, computer vision, and machine learning.

OpenCV (Open Source Computer Vision Library): OpenCV is an open-source computer vision and machine learning library that provides a comprehensive set of tools for image and video processing. It includes functions for face detection, image manipulation, and feature extraction.

face_recognition Library: The face_recognition library is a high-level wrapper built on top of dlib. It simplifies facial recognition tasks by using pre-trained deep learning models for face detection and encoding.

NumPy: NumPy is a powerful numerical computing library for Python. It provides support for large, multi-dimensional arrays and matrices, along with mathematical functions for efficient data manipulation.

datetime Module: The datetime module is part of the Python standard library and provides functions to work with dates and times. It enables the creation and formatting of timestamps.

3.3.2 Technologies

Computer Vision: Computer vision is a field of artificial intelligence that enables machines to interpret and understand visual information from the world, similar to the way humans perceive and interpret visual data.

Facial Recognition Technology: Facial recognition technology involves identifying and verifying individuals by analysing patterns and features of their faces. It often utilizes deep learning models and machine learning algorithms for accurate recognition.

OpenCV (Open Source Computer Vision Library): OpenCV is an open-source computer vision and machine learning library that provides a vast array of tools and functions for image and video processing.

CHAPTER 4
DESIGN

The design phase of the Face Recognition Attendance Management System is a critical step in transforming conceptual ideas into a structured and efficient solution. This phase involves outlining the architectural elements, data flow, and user interfaces that collectively contribute to the functionality and performance of the system.

4.1 Flow Chart

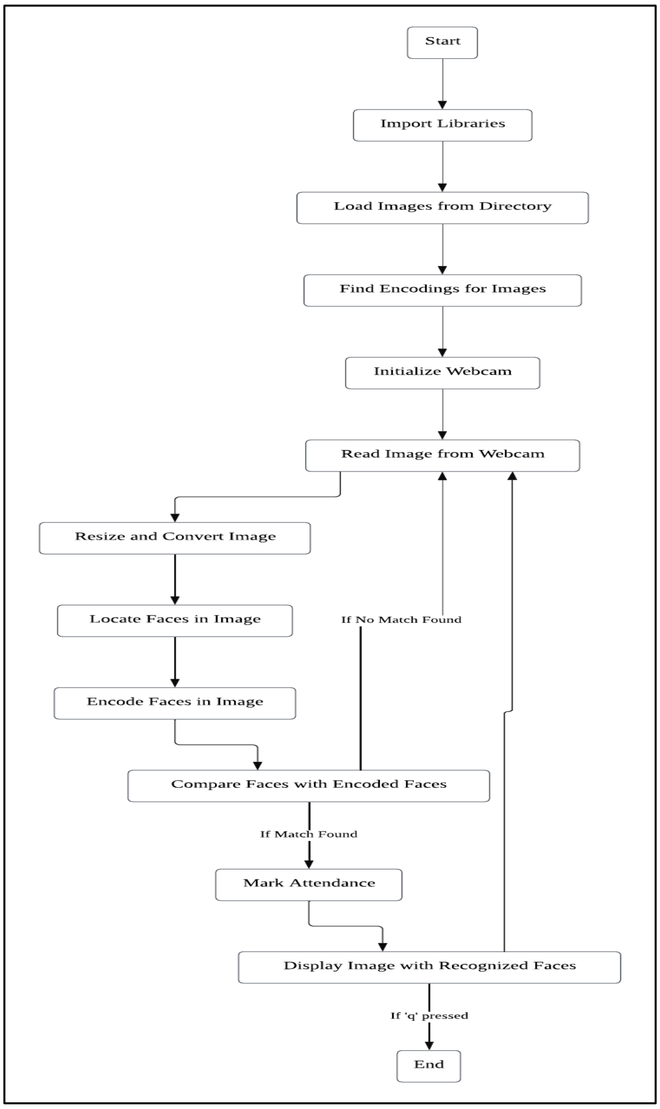

Fig 4.1 Flow Chart

Fig 4.1 system utilizes facial recognition to mark attendance. It begins by setting up the tools needed to process images. Then, it loads pictures containing faces from a designated folder. These faces are converted into a code the system can understand for future identification. Next, the system activates the webcam to capture a live video stream. From each frame, it extracts an image, resizes it, and prepares it for facial recognition. The system then searches for any faces within the image and creates unique codes for them. These newly created codes are compared against the ones stored earlier. If there's a match, the system marks attendance for the recognized person. Finally, the captured image is displayed, highlighting the recognized face with a box. If no match is found, the image is displayed without any markings. This process continues until the user presses the 'q' key to stop.

4.2 Sequence Diagram

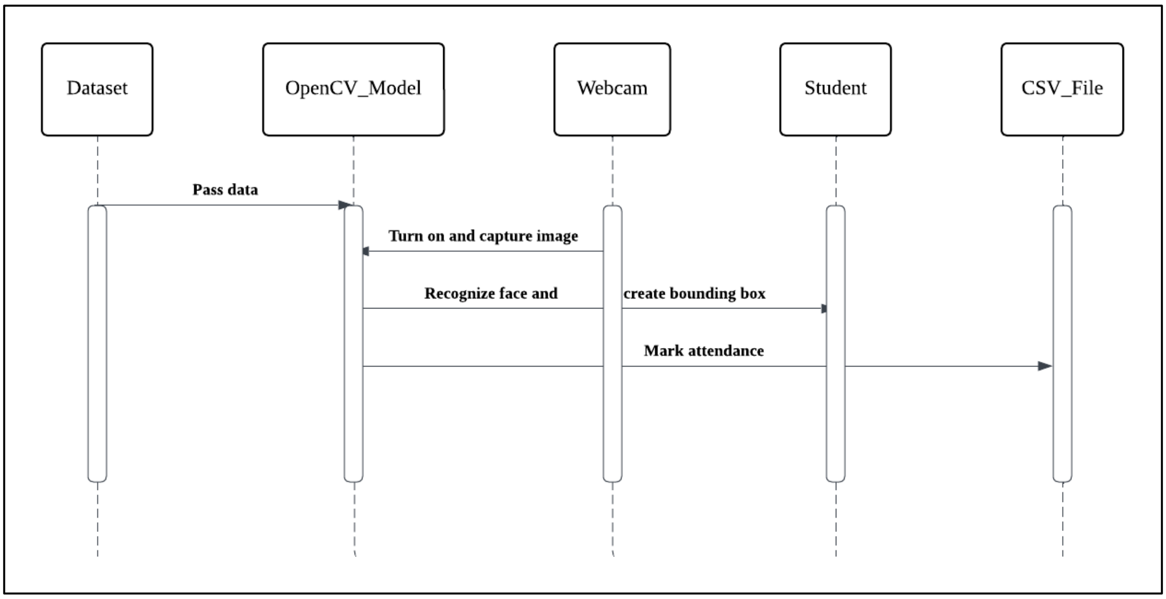

Fig 4.2 Sequence Diagram

The Face Recognition Attendance Management System initiates with a pre-existing dataset containing information about students. Utilizing the OpenCV model, the system processes this dataset, extracting relevant features and creating a set of facial encodings for each student.

Subsequently, the system engages with a webcam to capture real-time footage. OpenCV is employed to handle the webcam interaction, Fig 4.2 capturing frames as the basis for facial recognition. In this phase, the system utilizes the pre-trained face_recognition library to identify faces within the webcam frames. Upon recognition, the system dynamically creates bounding boxes around the detected faces, providing a visual representation of the identified individuals.

As faces are recognized, the system marks student attendance in a structured CSV file. To avoid duplicate entries during a single session, the system maintains a set of recognized students. The CSV file records attendance data, including the student's name, timestamp, and date. This systematic approach ensures accurate and real-time attendance tracking, leveraging the capabilities of computer vision and facial recognition technologies. The combination of OpenCV, face_recognition, and dataset processing enables an efficient and secure attendance management system.

4.3 Use case diagram

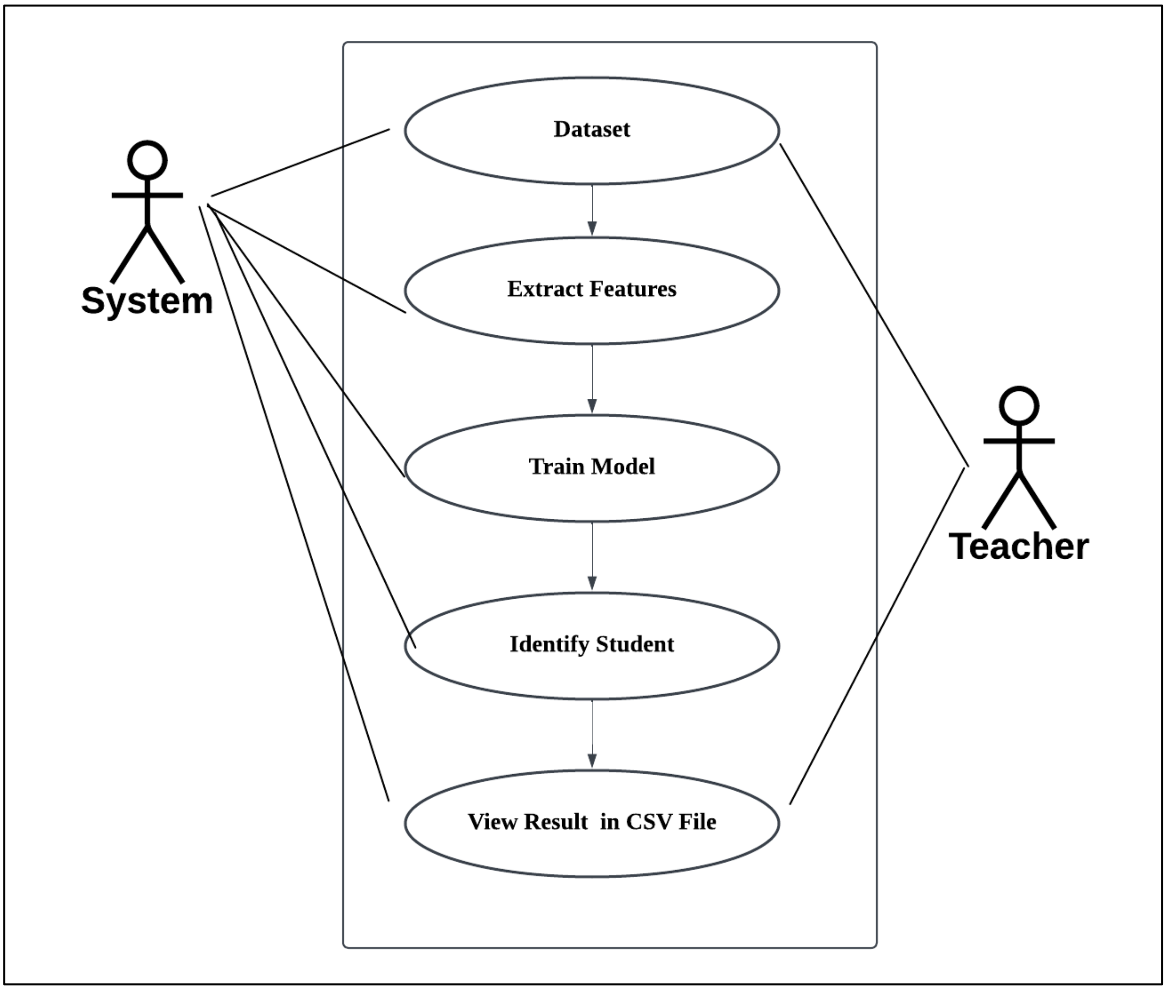

Fig 4.3 Use case Diageam

The teacher plays a pivotal role by providing a dataset containing essential student information. This dataset serves as the foundational material for the system's ability to recognize students later on. Fig 4.3 The information within the dataset may encompass various student details such as photos, ID numbers, or other unique identifiers, forming the basis for the system's recognition capabilities. Essentially, the teacher acts as the system's guide, supplying the necessary raw materials that enable the subsequent identification of students through the system.

Internally, the system operates by processing the supplied student data, extracting relevant information, and potentially focusing on key facial features or other distinctive characteristics. This data processing step serves as the precursor to another crucial functionality – model training. The system employs the processed student data to train a model, allowing it to learn and recognize students based on the extracted information. Although the intricate workings of these processes are not explicitly depicted in the use case diagram, it can be inferred that the system undergoes essential data processing and model training to achieve accurate student identification.

The ultimate output of the system is indicated in the text at the bottom of the diagram, mentioning "View Result in CSV File." This suggests that the system generates a CSV file containing the results of student identification. In this file, the recognized students may be listed, possibly accompanied by timestamps or additional details depending on the system's design. The CSV file serves as a tangible and accessible record of the system's identification outcomes, providing a structured and organized representation of student attendance or recognition results.

- **Summary**

The Face Recognition Attendance Management System is designed to efficiently track attendance using facial recognition technology. The system begins by setting up image processing tools and loading images of known faces from a designated folder. These faces are encoded for identification purposes. It then activates the webcam to capture live video streams, processes each frame, and searches for faces using pre-trained models. Upon recognition, attendance is marked in a structured CSV file, ensuring real-time tracking while avoiding duplicate entries. The system leverages OpenCV, face_recognition, and dataset processing to achieve accurate and secure attendance management. The teacher provides essential student information, serving as the system's guide, while internally, the system processes data and trains models for student recognition. The output of the system is a CSV file containing identification results, providing a structured record of student attendance or recognition outcomes. Overall, the system offers an efficient and user-friendly solution for automated attendance tracking in educational settings.

CHAPTER 5
IMPLEMENTATION

The implementation section outlines the key components of the system, such as image preprocessing, face encoding, real-time webcam capture, and attendance marking. The code is structured to load pre-existing student images, encode their facial features, and continuously process webcam frames for face recognition during a live session. The attendance of recognized students is recorded in a dynamically generated CSV file, named based on the current date.

5.1 Code Snippet

```
import cv2
import face_recognition
import os
import numpy as np
from datetime import datetime
import streamlit as st
path = 'MP/photos'
images = []
classNames = []
mylist = os.listdir(path)
for cl in mylist:
    curImg = cv2.imread(f'{path}/{cl}')
    images.append(curImg)
    classNames.append(os.path.splitext(cl)[0])

def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encoded_face = face_recognition.face_encodings(img)[0]
```

```
        encodeList.append(encoded_face)
    return encodeList

encoded_face_train = findEncodings(images)

recognized_students = set()  # Keep track of recognized students during the session

def markAttendance(name, attendance_file):
    if name.lower() not in recognized_students:
        recognized_students.add(name.lower())  # Add the recognized student to the set

        with open(attendance_file, 'a') as f:
            now = datetime.now()
            time = now.strftime('%I:%M:%S:%p')
            date = now.strftime('%d-%B-%Y')
            f.write(f'\n{name}, {time}, {date}')

def main():
    st.title("Face Recognition Attendance System")

    # Generate attendance file name based on the current date
    attendance_file_name = f'Attendance_{datetime.now().strftime("%Y-%m-%d")}.csv'

    # take pictures from the webcam
    cap  = cv2.VideoCapture(0)
    img_placeholder = st.empty()
    while True:
        success, img = cap.read()
        imgS = cv2.resize(img, (0, 0), None, 0.25, 0.25)
        imgS = cv2.cvtColor(imgS, cv2.COLOR_BGR2RGB)
```

```
        faces_in_frame = face_recognition.face_locations(imgS)
        encoded_faces = face_recognition.face_encodings(imgS, faces_in_frame)

        for encode_face, faceloc in zip(encoded_faces, faces_in_frame):
            matches = face_recognition.compare_faces(encoded_face_train, encode_face)
            faceDist = face_recognition.face_distance(encoded_face_train, encode_face)
            matchIndex = np.argmin(faceDist)

            if matches[matchIndex]:
                name = classNames[matchIndex].upper().lower()
                y1, x2, y2, x1 = faceloc
                # since we scaled down by 4 times
                y1, x2, y2, x1 = y1*4, x2*4, y2*4, x1*4
                cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 2)
                cv2.rectangle(img, (x1, y2-35), (x2, y2), (0, 255, 0), cv2.FILLED)
                cv2.putText(img, name, (x1+6, y2-5), cv2.FONT_HERSHEY_COMPLEX, 1,
(255, 255, 255), 2)
                markAttendance(name, attendance_file_name)

        img_placeholder.image(img, channels="BGR", use_column_width=True)
if __name__ == "__main__":
    main()
```

The provided code implements a real-time Face Recognition Attendance System using OpenCV, face_recognition library, and Streamlit framework. It starts by loading images of known faces from a specified directory and encoding them for recognition. The system captures video frames from the webcam, detects faces, and compares them with the encoded faces of known individuals. Upon recognition, the system marks attendance by appending the student's name, current time, and date to a CSV file. The Streamlit interface displays the live webcam feed with recognized faces highlighted and their corresponding names displayed. Overall, the system offers an automated and user-friendly solution for tracking attendance through facial recognition technology.

5.2 Screen Shorts

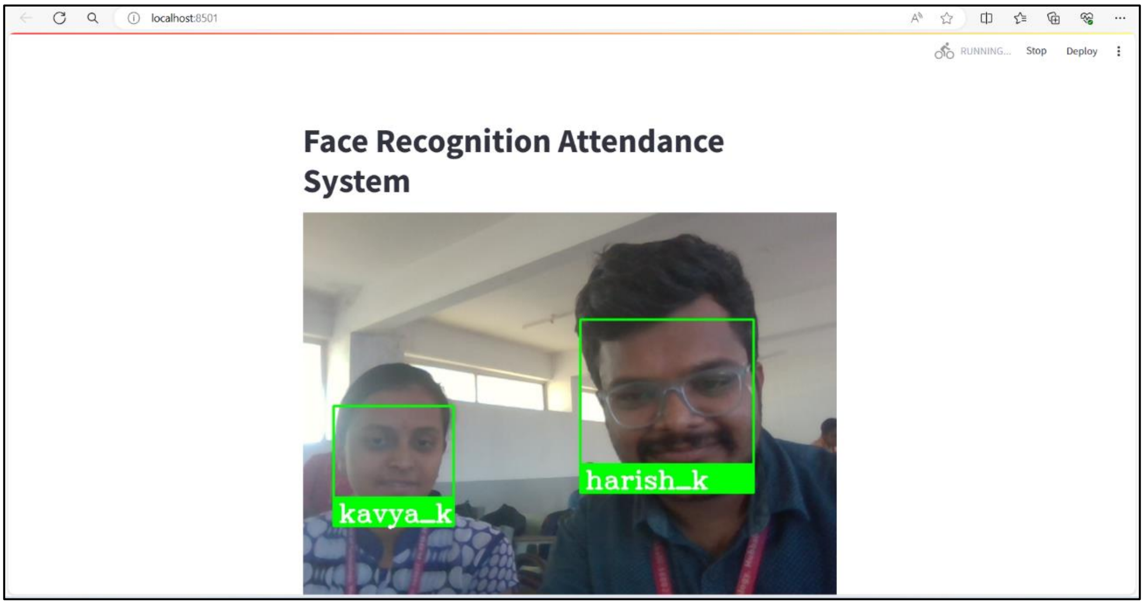

Fig 5.1 Screenshot of attendance marked as kavya_k and harish_k

The Fig 5.1 shows the screenshot captured after detecting the faces of the people in the frame as "kavya-k" and "harish-k". The screenshot also shows the faces being marked with their names after being recognised. Once the faces are annotated by the model the attendance is marked in the .csv file.

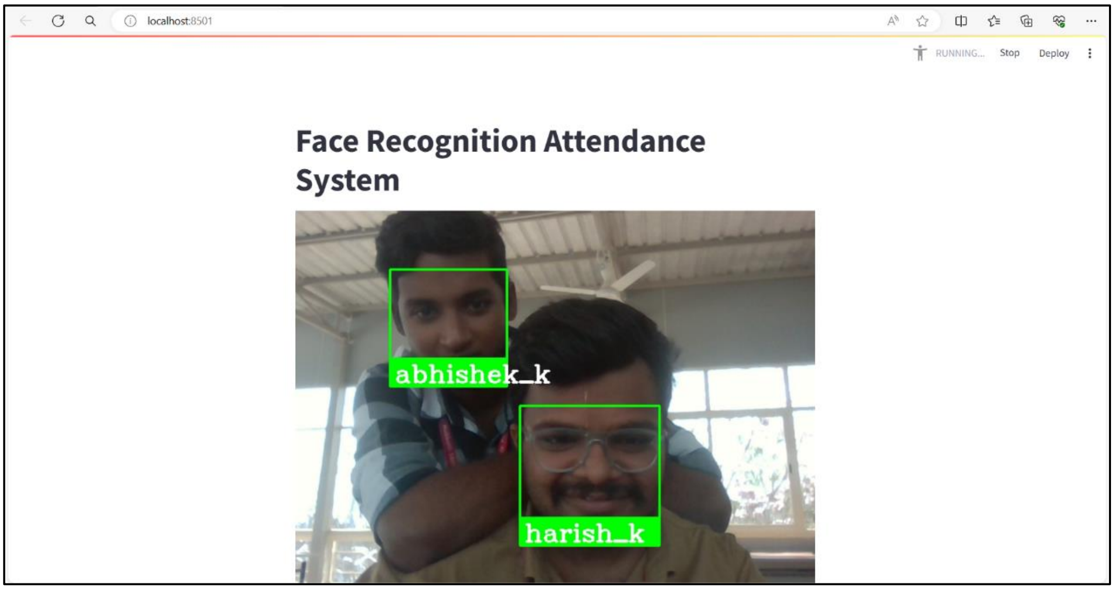

Fig 5.2 Screenshot of attendance marked as abhishek_k and harish_k

The Fig. 5.2 shows the screenshot captured after detecting the faces of the people in the frame as "abhishek-k" and "harish-k". The screenshot also shows the faces being marked with their names after being recognised. Once the faces are annotated by the model the attendance is marked in the .csv file.

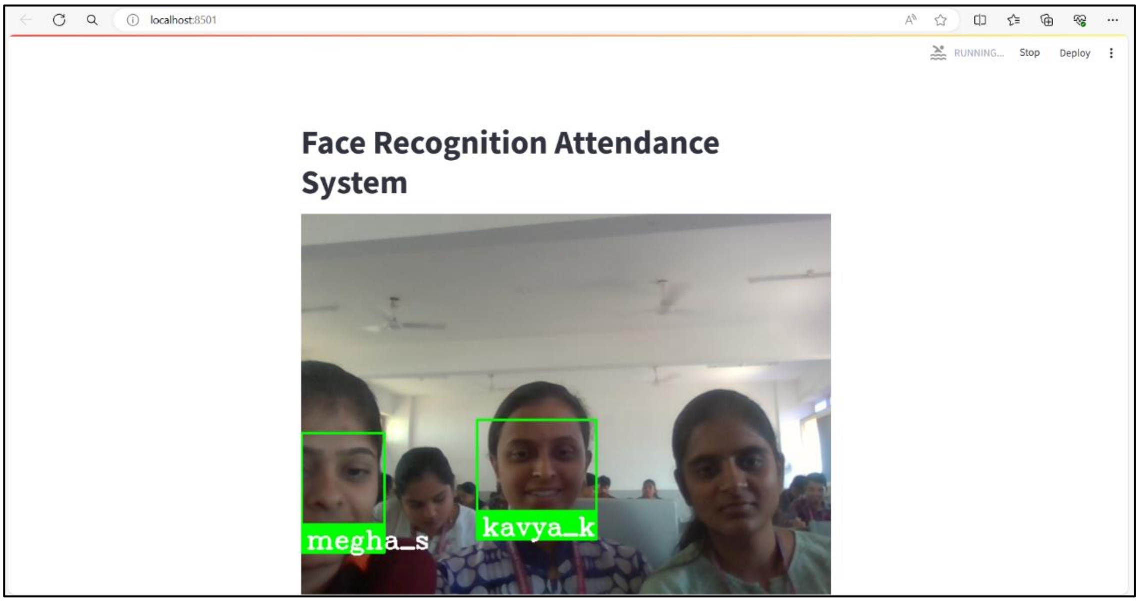

Fig 5.3 Screenshot of attendance marked as megha_s and kavya_k

The Fig 5.3 shows the screenshot captured after detecting the faces of the people in the frame as "kavya-k" and "megha_s". The screenshot also shows the faces being marked with their names after being recognised. Once the faces are annotated by the model the attendance is marked in the .csv file.

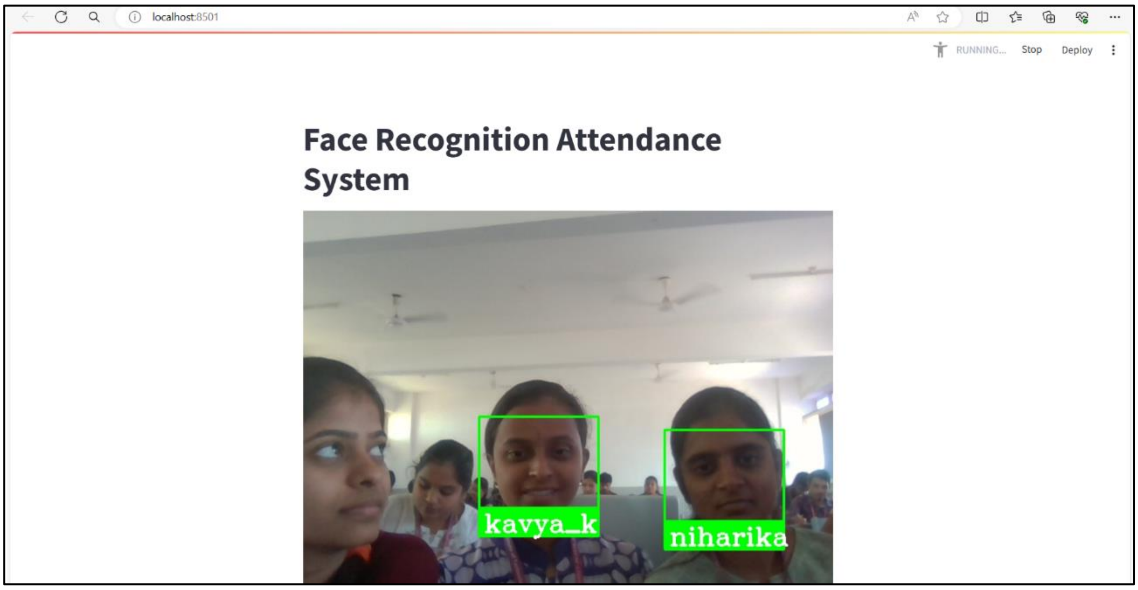

Fig 5.4 Screenshot of attendance marked as kavya_k and niharika

The Fig 5.4 shows the screenshot captured after detecting the faces of the people in the frame as "kavya-k" and "niharika". The screenshot also shows the faces being marked with their names after being recognised. Once the faces are annotated by the model the attendance is marked in the .csv file.

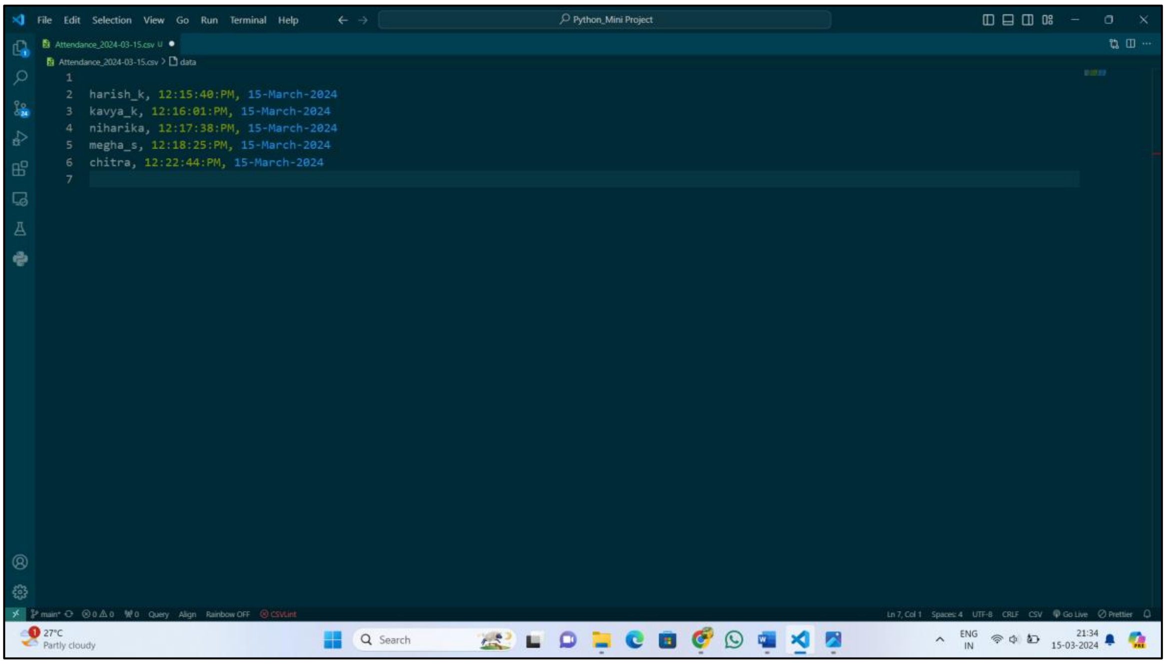

Fig 5.5 Screenshot of the attendance being marked and stored in .csv file

The Fig 5.5 shows the attendance being marked and stored in a .CSV (Comma Seperated Values) file in the format of name, time and date.

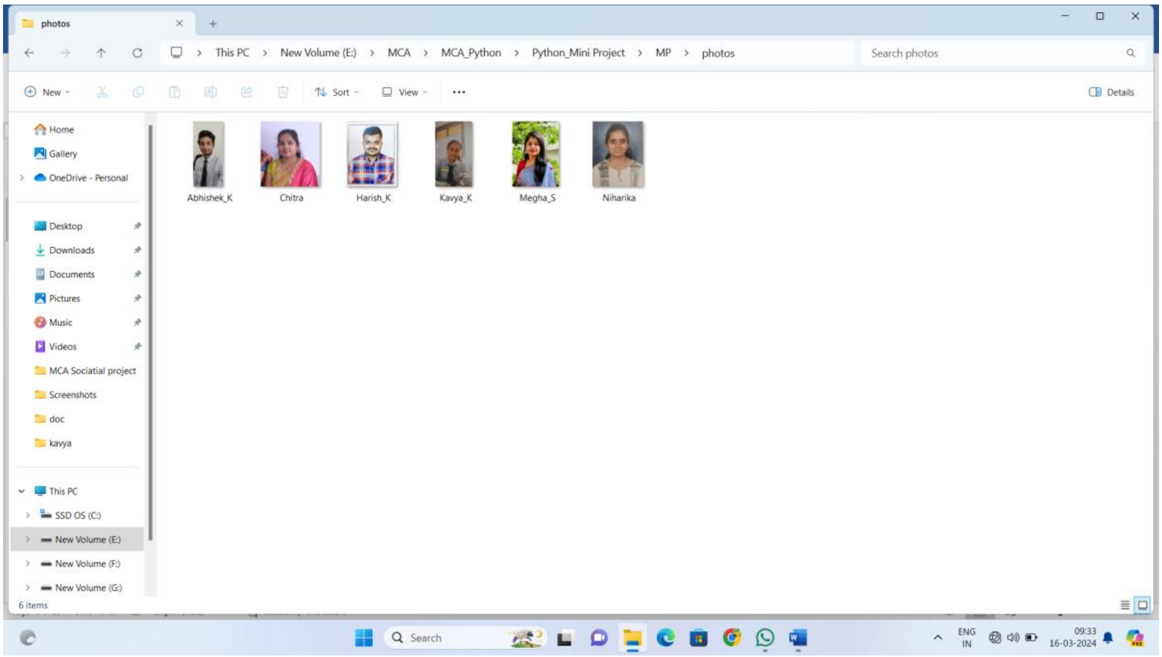

Fig 5.6. Screenshot of the dataset used to recognise the faces of the students

The Fig 5.6 shows the dataset used to label the images to allow the model to recognise the faces of the students.

CHAPTER 6
SOFTWARE TESTING

Software testing for the Face Recognition Attendance Management System could be organized into different testing categories along with test descriptions

6.1 Functional Testing

Functional testing ensures individual components of a software system function as expected, isolating and validating their correctness through focused tests.

Table 6.1 Functional Testing

Test Category	Test Description	Expected Outcome	Actual Outcome	Pass/Fail
Image Loading	Test whether the system can load images from the designated folder.	Images are successfully loaded	Images loaded	Pass
Face Encoding	Test whether the system correctly encodes images for identification.	Images are encoded correctly	Images encoded	Pass
Video Capture	Test whether the system can capture a live video stream from the webcam.	Webcam feed is displayed	Webcam feed displayed	Pass
Face Detection	Test whether the system can detect faces within each frame of the video stream.	Faces are detected within each frame	Faces detected	Pass
Face Recognition	Test whether the system correctly identifies faces using pre-trained models.	Matching faces are correctly identified	Faces identified	Pass
Attendance Marking	Test whether the system accurately marks attendance for recognized individuals.	Attendance is marked correctly	Attendance marked	Pass

6.2 Performance Testing

Table 6.2 Performance Testing

Test Category	Test Description	Expected Outcome	Actual Outcome	Pass/Fail

Video Frame Rate	Test whether the system can process video frames at an acceptable frame rate.	Video frames are processed smoothly	Frame rate maintained	Pass
Face Recognition Speed	Test whether the system can recognize faces within a reasonable time frame.	Faces are recognized in a timely manner	Recognition speed adequate	Pass

6.3 Usability Testing

Usability testing ensures individual components of a software system function as expected, isolating and validating their correctness through focused tests.

Table 6.3 Usability Testing

Test Category	Test Description	Expected Outcome	Actual Outcome	Pass/Fail
Interface Navigation	Test whether the user interface is intuitive and easy to navigate.	Users can easily navigate through the interface	Interface navigation smooth	Pass
Recognition Feedback	Test whether the recognition feedback provided to users is clear and informative.	Users can clearly identify recognized faces	Recognition feedback clear	Pass

6.4 Error Handling Testing

Table 6.4 Error Handling Testing

Test Category	Test Description	Expected Outcome	Actual Outcome	Pass/Fail

Absence of Recognized Faces	Test whether the system handles the absence of recognized faces gracefully.	No faces are marked for attendance	No attendance marked	Pass
Duplicate Attendance	Test whether the system prevents duplicate attendance entries within a session.	Attendance is not marked for duplicate recognitions	No duplicate attendance	Pass
Error Handling	Test whether the system properly handles errors and logs them as expected.	Errors are properly handled and logged	Errors handled	Pass

6.5 Security Testing

Table 6.5 Security Testing

Test Category	Test Description	Expected Outcome	Actual Outcome	Pass/Fail
Data Privacy	Test whether the system ensures the privacy and security of attendance data.	Attendance data is securely stored and accessible only to authorized users	Data privacy maintained	Pass

REFERENCES

[1] S. Chintalapati and M. V. Raghunadh, "Automated attendance management system based on face recognition algorithms," *IEEE International Conference on Computational Intelligence and Computing Research*, Enathi, India, 2021, pp. 1-5, doi: 10.1109/ICCIC.2021.6724266.

[2] Kasotiya, R., Raj, S. and Akhtar, S., 2021. Face Recognition Attendance Management System. *Turkish Online Journal of Qualitative Inquiry*, *12*(6).

[3] O. A. R. Salim, R. F. Olanrewaju and W. A. Balogun, "Class Attendance Management System Using Face Recognition," *2018 7th International Conference on Computer and Communication Engineering (ICCCE)*, Kuala Lumpur, Malaysia, 2018, pp. 93-98, doi: 10.1109/ICCCE.2018.8539274