AUTOMATED
ATTENDANCE USING FACE RECOGNITION

A MINI PROJECT

Submitted by

ARO KRISTEN A

JENITTA VINY J

DILLON D

ROYAL RIVALDO R

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

LOYOLA-ICAM COLLEGE OF ENGINEERING AND TECHNOLOGY

CHENNAI – 600034

ANNA UNIVERSITY: 600025

APRIL 2023

ANNA UNIVERSITY CHENNAI: 600025

BONAFIDE CERTIFICATE

Certified that this project report **"Automated Attendance using face recognition"** is the bonafide work of **ARO KRISTEN A (311120104009), JENITTA VINY J (311120104022), DILLON (311120104016) and ROYAL RIVALDO R (311120104050)** who carried out the project work under my supervision.

SIGNATURE

Dr. K. Gopalakrishnan

HEAD OF THE DEPARTMENT

Department of Computer Science and Engineering
Loyola-ICAM College of Engineering and Technology
Loyola Campus,
Nungambakkam,
Chennai-600034

SIGNATURE

Ms. Jeevitha A

SUPERVISOR

Assistant Professor
Department of Computer Science and Engineering
Loyola-ICAM College of Engineering and Technology
Loyola Campus,
Nungambakkam,
Chennai-600034

Submitted for the project viva voce held on …….

INTERNAL EXAMINER

EXTERNAL EXAMINER

Department Vision

To form competent computing engineers who are motivated towards innovation and collaborative research, to serve the ever-changing needs of the society. **Department Mission**

- To inculcate learnability skills in the student to acquire knowledge in the fundamentals of Computer Science and Engineering.
- To bring out the creativeness in the mind of the student leading to innovation and research.
- To develop professional skills and adaptiveness through collaborative activities.
- To create awareness on ethical values and involve the student to serve the societal needs.

Program Outcomes (PO)

PO 1 – Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialisation for the solution of complex engineering problems.

PO 2 – Problem analysis: Identify, formulate, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO 3 – Design/Development of Solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, and cultural, societal, and environmental considerations

PO 4 – Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO 5 – Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

PO 6 – The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO 7 – Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and the need for sustainable development.

PO 8 – Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO 9 – Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO 10 – Communication: Communicate effectively on complex engineering activities with the engineering community and with the society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO 11 – Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO 12 – Life-long learning: Recognise the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Educational Objectives (PEO)

- Graduates apply the acquired mathematical, scientific and engineering skills to meet the growing challenges of industry, pursue higher education and research.
- Graduates provide solutions to contemporary engineering problems in the fields of Electronics and Communication by employing modern techniques and tools.
- Graduates possess the capacity to embrace leadership and team work opportunities and also entrepreneurial ventures to design, develop and implement innovative systems for societal, global, and environmental issues.
- Graduates engage in lifelong learning, continuous improvement and adapt to trending technologies

Program Specific Outcomes (PSO)

PSO 1 – Apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 2 – Adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing novel problems.

PSO 3 – Attain excellence in emerging fields of Machine Learning and Data Science.

Course Outcomes (CO)

CO No.	Course Outcomes
CO1	Map the technical knowledge acquired in the previous semesters for solving real world problems
CO2	Apply new technologies and design techniques (platform, database, etc.) concerned with devising a solution for a given problem statement
CO3	Apply project management skills (scheduling work, procuring parts and documenting expenditures and working within the confines of a deadline
CO4	Work with team mates, sharing due and fair credits and collectively apply effort for making project successful
CO5	Communicate technical information by means of written and oral reports

ACKNOWLEDGEMENT

Firstly, we are deeply obliged to the almighty for this opportunity and for leading us to the successful completion of the project. The experience has been very fruitful.

We express our sincere gratitude to the beloved Director Rev Dr S SEBASTIAN SJ and the Dean of Students Dr. D Caleb Chanthi Raj for their continuous support and Encouragement.

We are deeply obliged to the Principal Dr. L Antony Michael Raj for his inspiring guidance and invaluable advice during the mini project. We also express our sincere thanks to the Head of the Department and Project Coordinator Dr K Gopalakrishnan, the teaching and non-teaching faculty of our department for their advice, support, and guidance.

We extend our gratitude to our project guide Ms. Jeevitha A for providing valuable insights and resources. Without her supervision and constant help, this project would not have been possible.

Last but not the least, we are extremely grateful to our family and friends, who have been a constant source of support during the preparation of this mini project.

ABSTRACT

The proposed attendance system is a computer vision-based solution that automates the process of recording attendance. The system utilizes a webcam to capture images of individuals, detects and recognizes their faces, and records their attendance in a structured manner.

The system is built using Python and incorporates various libraries such as tkinter, OpenCV, numpy, PIL, pandas, datetime, and time. The algorithm starts by importing the required libraries and checking for the presence of the necessary haar cascade file. It then creates the required directories and sets up functions for displaying the current time, handling contact information, and changing the system password. Before accessing the system, the user is prompted for a password to ensure authentication. The algorithm provides functions for capturing images, training the system with the captured images, and retrieving the images and their corresponding labels.

The attendance tracking functionality is implemented by detecting faces in the webcam feed, recognizing them using a trained model, and marking their attendance. The system maintains attendance records by creating and updating CSV files. The main GUI window is created with appropriate widgets and layouts, including buttons for capturing images, saving profiles, tracking attendance, and clearing input fields. Additionally, a clock widget is displayed to show the current time.

The algorithm concludes by handling the event loop for the GUI application. Overall, the proposed attendance system offers an efficient and automated approach to attendance management, eliminating the need for manual recording and reducing the chances of errors.

Keywords: *python, tkinter, openCV, Haar cascades, OpenCV, CSV, GUI, attendance system,*

TABLE OF CONTENT

TABLE OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 PROBLEM DEFINITION

The process of attendance management is a critical aspect of various domains, including educational institutions, workplaces, and events. Traditional methods of manual attendance management are often time-consuming, prone to errors, and susceptible to misuse. To overcome these challenges, there is a growing demand for automated attendance systems that offer improved efficiency and accuracy.

This project proposes the implementation of an automated attendance system using face recognition techniques, developed using the Python programming language. Face recognition, a subfield of computer vision and machine learning, has made significant advancements in recent years, enabling the development of robust and reliable systems for identifying individuals based on their facial features.

The objective of this project is to leverage the power of face recognition algorithms and Python's capabilities to create an efficient and automated attendance management system. By employing computer vision techniques, the system can detect and recognize individuals in real-time, eliminating the need for manual record-keeping and reducing administrative efforts.

1.2 EXISTING APPLICATIONS

a) Truein Face recognition Time & Attendance

This one is a Face Recognition Attendance System, that gives you touchless cloud-based software to help you handle the HRMS. You can use this software and its app on Android or iOS devices. It doesn't require any complicated hardware. Moreover, with such an attendance management software you can handle office attendance, contract staff attendance, etc.

b) Saral PayPack

HRMS offers you a comprehensive payroll and face recognition attendance system, for all your payroll needs. It is a one-stop destination for your search as it possesses the most sought requirements of the payroll management system. This is the right choice for automating your payroll.

c) Darwinbox HR

An end-to-end face recognition attendance software built to empower small and medium-sized enterprises in their journey towards a truly digitized form of business management. It is a value-adding and impactful framework for HR management. The platform tries to meet all employee needs right from hiring, and onboarding, to performance management and exit.

1.3 NEED FOR THE SYSTEM

The existing automated attendance system using face recognition in the market, each system has its own unique features and limitations, and there is still room for improvement in terms of scalability, accessibility, and security. So, the need for an automated attendance system using face recognition arises from various factors and challenges associated with traditional attendance tracking methods. Some of the key reasons for implementing such a system include:

➔***Efficiency and Time Savings:*** Manual attendance tracking methods, such as paper registers or roll call, can be time-consuming and tedious, especially in large classrooms or organizations. Automating the process with face recognition technology significantly reduces the time required to record attendance, allowing teachers or administrators to focus on more productive tasks.

➔ ***Accuracy and Elimination of Errors:*** Human errors, intentional or unintentional, can occur during manual attendance recording, leading to inaccuracies and discrepancies in attendance data. Automated systems using face recognition technology offer a high level of accuracy by relying on biometric identifiers unique to individuals, minimizing the chances of errors or fraudulent practices such as buddy punching or proxy attendance.

➔***Security and Fraud Prevention:*** Face recognition technology enhances security measures by verifying the identity of individuals based on their unique facial features. This helps prevent unauthorized access or impersonation, ensuring that only authorized individuals are marked as present. By eliminating fraudulent practices, such as sharing ID cards or

signing in on behalf of others, the system maintains the integrity of attendance records.

➔Data Analysis and Insights: Automated attendance systems can generate detailed reports and analytics, providing valuable insights into attendance patterns, trends, and overall performance. This data can be used for identifying areas of improvement, optimizing resource allocation, or addressing attendance-related issues promptly.

CHAPTER 2

LITERATURE SURVEY

2.1 Student's attendance management using deep facial recognition

This paper [1] proposes an attendance management system based on facial recognition. Face detection and recognition are performed by convolutional neural network models MTCNN and FaceNet, respectively. Based on the results, it can be concluded that the proposed architecture presents a good solution for managing the attendance of students in classrooms.

2.2 Deep-learning based group-photo Attendance System using One Shot Learning

This paper [2] has attempted to present a solution to make the attendance marking procedure in schools and colleges easy and hassle free. A novel system architecture comprising of selftrained face recognition model is presented along with a novel face matching algorithm, a fully functional backend system comprising of all basic attendance functionality with login functionality and an android interface for the same. Our system uses one shot learning based face recognition which makes our system useful for additional students by requiring only just one image per student. Our model achieves accuracy of 97% on LFW dataset and our attendance system achieves accuracy of 85% on a public student class photo dataset. The visual results of our functioning system is presented in a live environment.

2.3 Smart Multiple Attendance System through Single Image

In this paper [3] he designed a smart, single input, multiple outputs, attendance system based on face recognition algorithms. This system overcomes the problems faced while using traditional systems. As those methods are time-consuming and required each person to access the system separately to mark his/her attendance. In this project, we obliterate the above challenges and reveal the automated face recognition attendance system that takes attendance by using a camera that is situated in front of the classroom to receive images of a whole class in real-time, detect the faces in the image and crop the image and then compare with the database. Once a student is recognized, it marks him/her present. The process is repeated a few times to increase system efficiency and the final results are recorded in the excel file. This automated attendance system saves the precious study time of the students as it runs in the background and needs little to no interaction from the teachers or the students. This system also keeps down the manual work and burden on the lecturers for accurate marking of attendance and improves security.

2.4 IAAS: IoT-Based Automatic Attendance System with Photo Face Recognition in Smart Campus

This paper[4] proposes an IoT-based Automatic Attendance System (called IAAS). Our goal is to provide a reliable and improved automatic attendance checking system with face recognition technologies. This goal is achieved by constructing an improved system structure with suitable face

recognition technologies and using training data with Haar-cascade to extract a user's face data. With this system, it is expected that that our IAAS system can provide convenience for attendance checking to both students and professors. As future work, we will explore the impact of postures or appearance changes of users on the performance and develop a solution to deal with such factors.

2.5 Development of an Automatic Class Attendance System using CNN-based Face Recognition

In this paper [5] , a system for automatically marking and storing the attendance of a class has been implemented. Implementation process includes entering data of the students, training dataset, recognizing faces and marking attendance automatically. The CNN model used in this study can detect and recognize a person by their facial features even if they are not staring exactly straight into the camera. The proposed system can detect and recognize the students of the class with maximum accuracy of about 92% and saves the teachers' time and hassle by automatically marking and storing the attendance of the present students. For the system to be most effective, it has to contain a satisfactory and consistent amount of images of each person during the training stage. Also, the camera has to be positioned in a way that it has clear view of all the students. Furthermore, this system can be used in any organization for automatic attendance recording of staffs.

CHAPTER 3

SYSTEM ANALYSIS

3.1 SYSTEM REQUIREMENTS

The following specifications were those required by the system for the software's successful implementation and functioning

3.1.1 HARDWARE REQUIREMENTS

➔ Computer or laptop with a webcam

➔ Sufficient memory and processing power to handle image processing tasks

➔ Stable internet connection (optional, depending on the desired functionality)

3.1.2 SOFTWARE REQUIREMENTS

➔ Operating System: Windows, macOS, or Linux

➔Python programming language (version 3.x)

➔Python libraries: tkinter, OpenCV, numpy, PIL, pandas

➔Haarcascade XML file for face detection (e.g., haarcascade_frontalface_default.xml)

➔Integrated Development Environment (IDE) for Python development (e.g., PyCharm, Jupyter Notebook, Visual Studio Code)

➔CSV file reader/writer (typically available with Python installations)

➔Basic text editor (e.g., Notepad, Sublime Text) for modifying configuration file

CHAPTER 4

SYSTEM DESIGN

4.1 OBJECT ORIENTED DESIGN

Identifying the objects in a system would be what OO (Object Oriented) analysis and design have always been about identifying their relationships. Generating a design that could be transformed into executables utilizing object oriented programming languages.

UML (Unified Modelling Language) is a standard language that uses, designs, constructs, and documents software components. The Unified Modelling Language (UML) is an effective instrument enabling Object Oriented Analysis and Design. The Object Management Group (OMG) created it, and in January 1997, the OMG proposed UML 1.0 as a specification draft. It started as a way to capture the behavior of vast software and non-software systems and has since grown into such an international standard. UML is created to be process generic, indicating it could be used in a variety of circumstances. It can be used for a spectrum of uses. Business analysts, software architects, and developers are all using UML as a common language. It can be used to describe, specify, build, and document the system's business processes, including its structural and behavioural artifacts.

UML is a modelling language used to create software blueprints. Diagrams are categorized into two divisions, which are further divided into subcategories:

- **Structural Diagrams**
- **Behavioural Diagrams**

4.2 STRUCTURAL DIAGRAMS

The static aspect of the system is portrayed by the structural diagrams. These static aspects are the components of a diagram that describes the main structure and are thus stable. Classes, interfaces, objects, components, and nodes are often used to represent them.

Some of the structural diagrams are:

- Class diagram
- Object diagram
- Component diagram
- Deployment diagram

4.3 BEHAVIOURAL DIAGRAMS

Behavioural diagrams illustrate a system's complex nature. The changing/moving parts of a system are usually known for the dynamic aspect. The following five types of behavioural diagrams are supported in UML:

- Use case diagram
- Sequence diagram
- Class diagram
- Statechart diagram
- Deployment diagram

4.4 USE CASE DIAGRAM

The use case diagrams show the system's behavior concerning the deployment environment. It illustrates the proposed system's users. A use case is a system analysis methodology for finding, explaining, and monitoring program needs.

A use case is a collection of conceivable sequences of interactions between systems and users in a specific environment, all of which are tied to a specific purpose. A use case is represented by an ellipse with the name of the use case. A stick figure with a name is used to represent an actor.

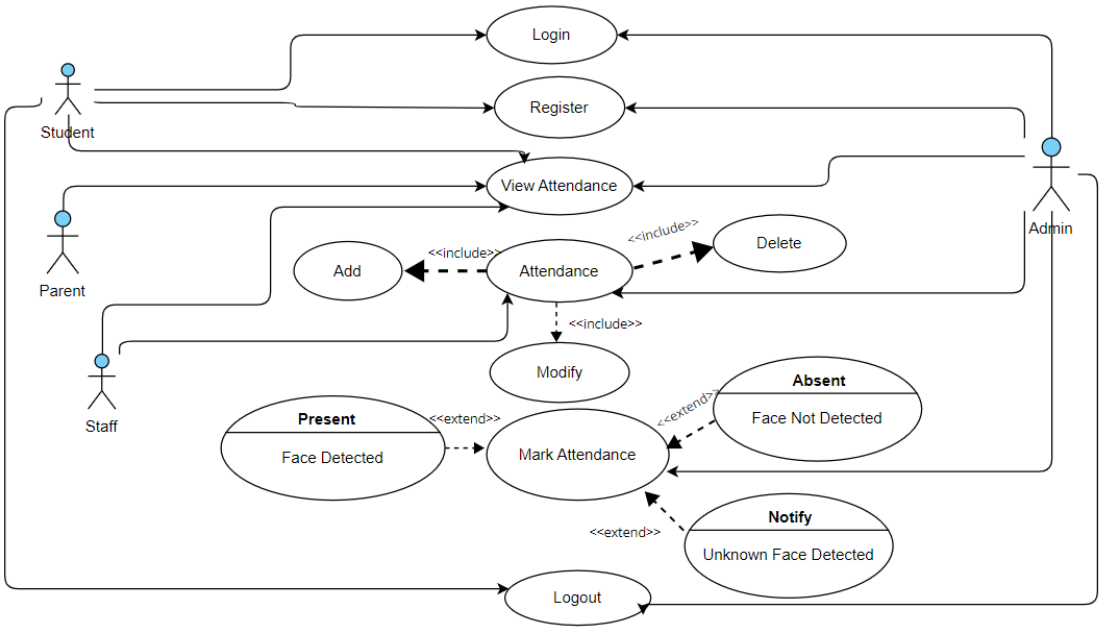

Fig 1.1 Use Case diagram

4.5 SEQUENCE DIAGRAM

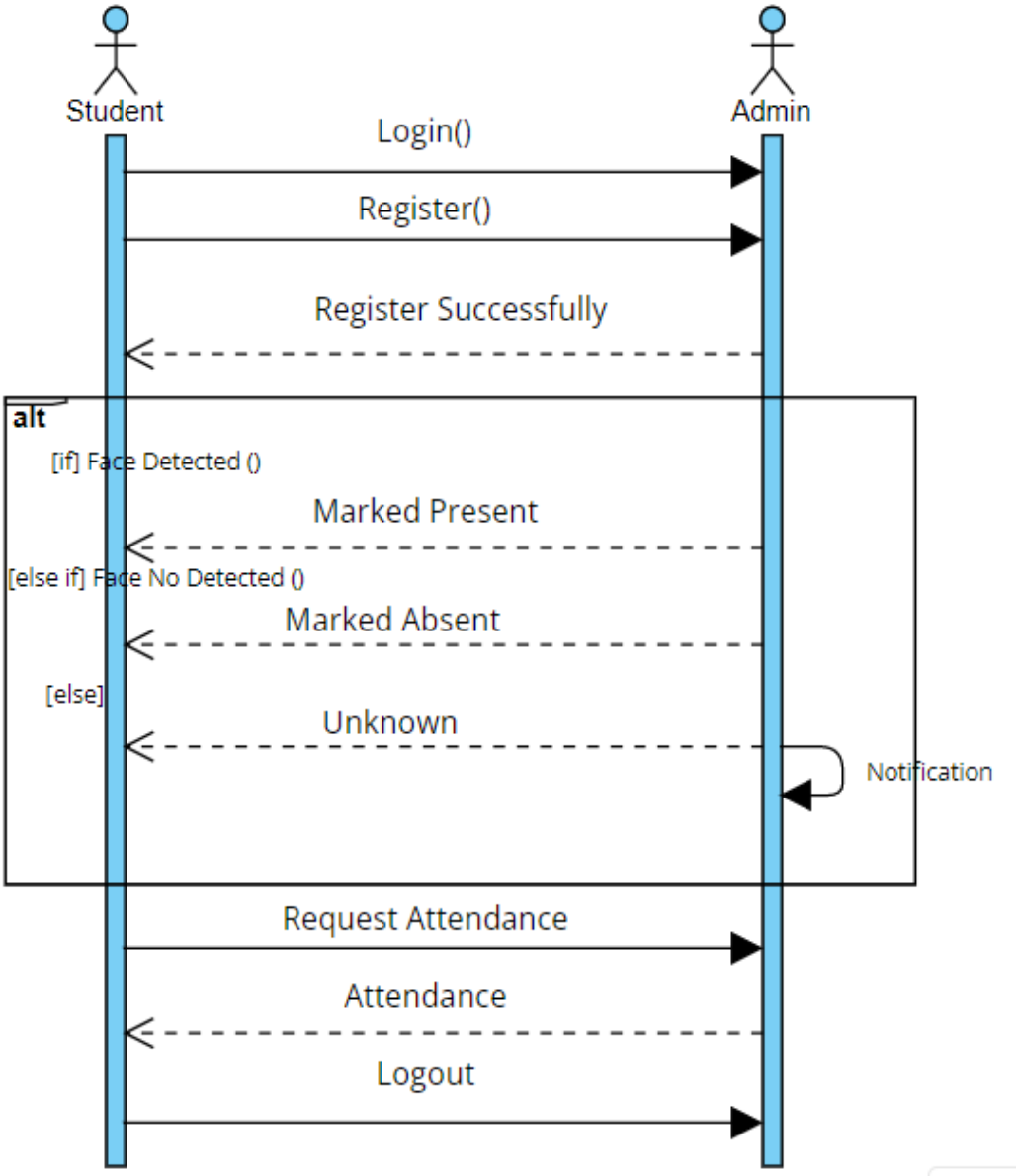

Fig 1.2 Sequence diagram

4.6 CLASS DIAGRAM

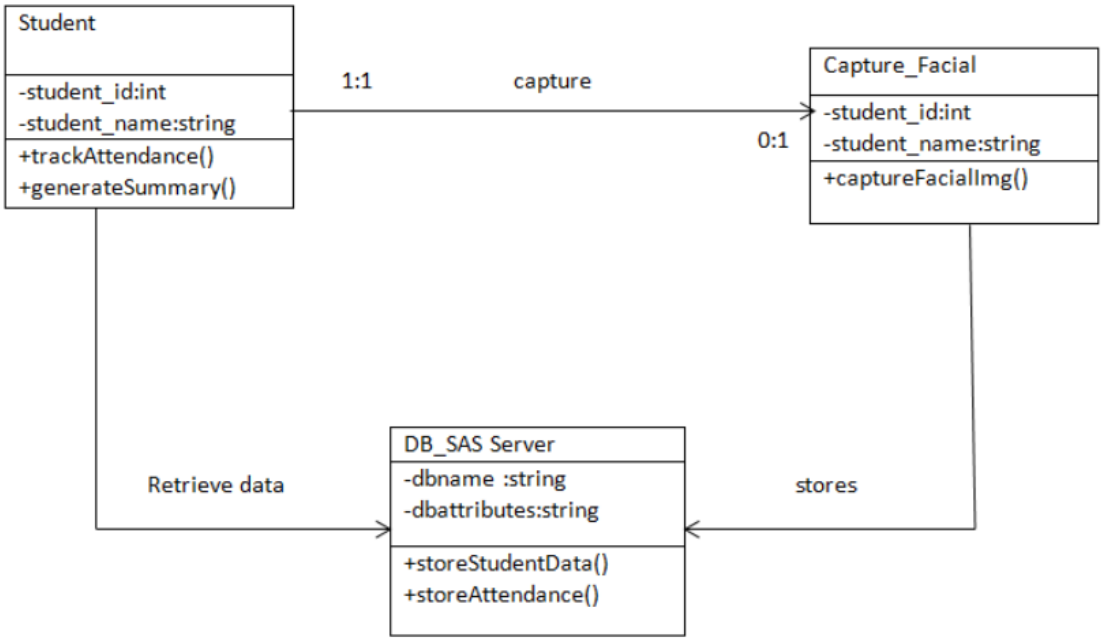

Fig 1.3 Class diagram

CHAPTER 5

SYSTEM IMPLEMENTATION

5.1 OPEN CV in PYTHON

. OpenCV is an open-source software library for computer vision and machine learning. The OpenCV full form is Open Source Computer Vision Library. It created to provide a shared infrastructure for applications for computer vision and to speed up the use of machine perception in consumer products. OpenCV, as a BSD-licensed software, makes it simple for companies to use and change the code. There are some predefined packages and libraries that make our life simple and OpenCV is one of them.

The library has more than 2500 optimised algorithms, including an extensive collection of computer vision and machine learning algorithms, both classic and state-of-the-art.Using OpenCV it becomes easy to do complex tasks such as identify and recognise faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D object models, generate 3D point clouds from stereo cameras, stitch images together to generate an entire scene with a high resolution image and many more.

Python is a user friendly language and easy to work with but this advantage comes with a cost of speed, as Python is slower to languages such as C or C

5.2 IMAGE CLASSIFICATION

Image classification is the task of assigning a label or class to an input image, based on its visual content. This is a common problem in computer vision and is used in a wide range of applications such as object recognition, facial recognition, and self-driving cars.

The process of image classification typically involves several steps:

- **Pre-processing:** This step involves preparing the input image for classification, which may include resizing, cropping, and normalizing the image.
- **Feature extraction:** This step involves extracting relevant features from the pre-processed image, which are then used as input to the classifier.
- **Classification:** This step involves using a machine learning model to make a prediction about the class of the input image, based on the extracted features.

There are various types of image classification, such as binary classification, multi-class classification, and multi-label classification. Deep learning models such as convolutional neural networks (CNNs) are commonly used for image classification due to their ability to automatically learn features from the input images.

5.3 PROPOSED ARCHITECTURE

The system architecture for the Automated Attendance Using Face Recognition project involves several components and stages working together to achieve the desired functionality. Here is an overview of the system architecture:

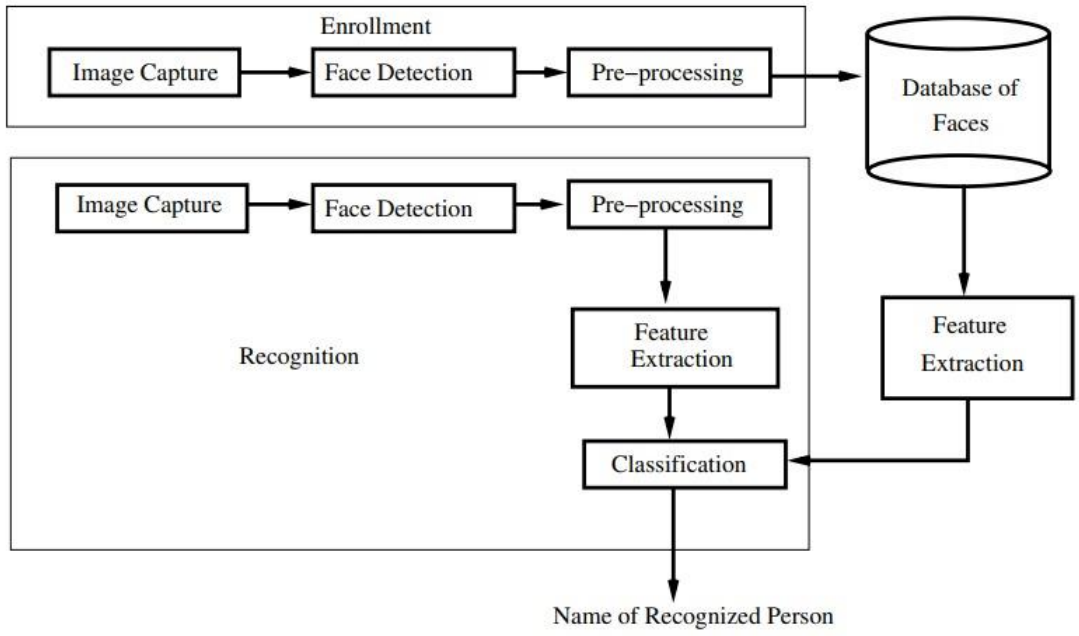

Fig 1.4 System architecture

Input Source:

Camera:

The system takes input from a camera, which can be a webcam or any other connected camera device. The camera captures images or video frames of individuals to be processed for attendance.

Face Detection:

Face detection is the first stage of the system architecture. It involves using computer vision algorithms to locate and extract faces from the input images or video frames. OpenCV library provides pre-trained face detection models like Haar cascades.

Feature Extraction:

Once the faces are detected, the system extracts features from the detected faces. These features capture the unique characteristics of each individual's face, allowing for accurate identification. Feature extraction techniques like Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP), or deep learning-based facial recognition models such as VGGFace or FaceNet can be used for this stage. The dlib library, along with pre-trained models, can be utilized for facial landmark detection and feature extraction.

Face Recognition:

The extracted features are then used for face recognition. The system compares the extracted features of the detected face with the features of pre-registered faces stored in the database. Machine learning algorithms such as Support Vector Machines (SVM), Convolutional Neural Networks (CNNs), or k-Nearest Neighbors (k-NN) can be employed for face recognition. The scikit-learn library provides implementations of these algorithms for training and classification.

Attendance Recording:

Once the face is recognized and matched with a pre-registered face, the system records the attendance for that individual along with a timestamp. The attendance information can be stored in a **database** or a file for further processing or analysis.

Output Display:

The system can provide real-time feedback by displaying the detected faces, recognized individuals, and attendance information on a graphical user interface (GUI). Libraries like OpenCV or GUI frameworks like Tkinter can be used to build the GUI and display the output.

5.4 PROPOSED METHODOLOGIES AND TECHNOLOGIES

The following section outlines the methodology and software development tools and technologies used in our system.

A. ***Data Collection:*** Gather a dataset of facial images from students to create a reliable face recognition model. Capture images in various lighting conditions and angles to enhance the model's accuracy.

B. ***Pre-processing:*** Perform pre-processing techniques such as face detection, alignment, and normalization to standardize the facial images and remove noise or inconsistencies.

C. ***Feature Extraction:*** Utilize deep learning techniques, such as Convolutional Neural Networks (CNN), to extract discriminative features from the facial images. Popular CNN architectures like VGG, ResNet, or Inception can be employed for this purpose.

D. ***Model Training:*** Train the face recognition model using the extracted features and a suitable algorithm, such as Siamese networks or Triplet Loss. The model should learn to differentiate between different individuals accurately.

E. ***Face Detection:*** Implement a face detection module that can identify faces from a live video stream or images captured by a camera. Libraries like OpenCV can be used for this task.

F. ***Face Recognition:*** Utilize the trained model to recognize and verify faces detected in real-time. Compare the extracted features of the detected face with the stored features of registered individuals to identify them.

G. ***Attendance Management:*** Maintain a database or record system to store the attendance information of individuals. Update the attendance records based on the recognized faces in real-time.

H. ***Graphical User Interface (GUI):*** Develop a user-friendly GUI using frameworks like tkinter, PyQt, or Electron, allowing users to interact with the system easily.

5.5 ALGORITHM

Face Recognition Algorithm:

The algorithm for the face recognition used for biometric authentication of users are as follows.

Step 1: Import the necessary libraries and modules for the system, including tkinter, OpenCV, numpy, PIL, pandas, datetime, and time.

Step 2: Create a function to check if the required haarcascade file exists.

Step3: Create a function to create the necessary directories if they don't already exist.

Step4: Create a function to display and update the current time using a clock widget.

Step5: Create a function to handle the "Contact us" functionality, displaying a messagebox with contact information.

Step 6: Create a function to handle changing the password for the system.

Step 7: Create a function to save the new password.

Step 8: Create a function to handle the password prompt and authentication before accessing the system.

Step 9: Create a function to clear the input fields for ID and name.

Step 10: Create a function to capture images from the webcam and save them with the corresponding ID and name.

Step 11: Create a function to train the system by creating a face recognizer and training it with the captured images.

Step 12: Create a function to retrieve the captured images and their corresponding labels from the training directory.

Step 13: Create a function to handle the attendance tracking functionality.

Step 14: Create a function to detect faces in the webcam feed, recognize them using the trained model, and mark their attendance.

Step 15: Create a function to create and update the attendance records in CSV files.

Step 16: Create the main GUI window with appropriate widgets and layouts.

Step 17: Implement the "Take Images" button to capture and save images.

Step 18: Implement the "Save Profile" button to train the system with the captured images.

Step 19: Implement the "Track Images" button to start the attendance tracking process.

Step 20: Implement the "Clear" button to clear the input fields.

Step 21: Start the clock widget to display the current time.

Step 21: Handle the main event loop for the GUI application.

CHAPTER 6

CONCLUSION AND FUTURE WORKS

6.1 CONCLUSION

In conclusion, the development of an automated attendance system using face recognition technology in Python offers a transformative solution for organizations and educational institutions. By leveraging computer vision algorithms and facial analysis techniques, the system provides a reliable, accurate, and efficient method for managing attendance records.The use of face recognition technology eliminates the shortcomings of traditional manual attendance tracking methods, such as time-consuming processes, human errors, and proxy attendance. With automated attendance, individuals can be quickly identified and logged in real-time, saving valuable time for both administrators and attendees. The implementation of face recognition technology in Python enables seamless integration with existing systems and provides a user-friendly interface for administrators to manage attendance records, generate reports, and maintain the database of individuals. Furthermore, the automated attendance system enhances security by ensuring that only authorized individuals can mark their attendance, reducing the risk of identity fraud or proxy attendance. By adopting this automated attendance system, organizations can streamline their attendance management processes, improve efficiency, and allocate resources to more valuable tasks. It offers a cost-effective solution by eliminating the need for paper-based systems or physical identification cards. Overall, the automated attendance system using face recognition technology in Python provides an advanced, accurate, and reliable method for attendance tracking, significantly benefiting organizations and educational institutions in their daily operations.

6.2 FUTURE WORK

In the future, the development of python, automated attendance using face recognition, can focus on several important areas. Firstly, We will Explore and implement more advanced face recognition algorithms, such as deep learning-based methods like Convolutional Neural Networks (CNNs), to achieve better accuracy and robustness in recognizing faces.

Secondly, We will Modify the system to recognize and track multiple faces simultaneously. This is especially useful in scenarios where there are group activities or multiple individuals present in the frame.

Thirdly, We will develop a mobile application for students and faculty to view their attendance records, receive notifications, and access attendance-related information. This can enhance convenience and accessibility for all stakeholders.

Fourthly, We ensure the compliance with privacy regulations and implement measures to protect the personal data of students and faculty. This can include data encryption, access controls, and anonymization techniques.

Lastly, continuously improve the user interface of the attendance system to make it intuitive, easy to use, and adaptable to different environments and user preferences.

APPENDIX

APPENDIX A - SOURCE CODE

Main.py

```
####################IMPORTING###################################

import tkinter as tk
from tkinter import ttk
from tkinter import messagebox as mess
from tkinter import *
import tkinter.simpledialog as tsd
import cv2,os
import csv
import numpy as np
from PIL import Image
import pandas as pd
import datetime
import time

###########################FUNCTIONS##############################

def assure_path_exists(path):
dir = os.path.dirname(path)
if not os.path.exists(dir):
os.makedirs(dir)
def tick():
time_string = time.strftime('%H:%M:%S')
clock.config(text=time_string)
clock.after(200,tick)


def contact():
mess._show(title='Contact us', message="Please contact us on :
'admin@licet.ac.in' ")
```

```
def check_haarcascadefile():
    exists = os.path.isfile("haarcascade_frontalface_default.xml")
    if exists:
        pass
    else:
        mess._show(title='Some file missing', message='Please contact us for
help')
        window.destroy()



def save_pass():
    assure_path_exists("TrainingImageLabel/")
    exists1 = os.path.isfile("TrainingImageLabel\psd.txt")
    if exists1:
        tf = open("TrainingImageLabel\psd.txt", "r")
        key = tf.read()
    else:
        master.destroy()
        new_pas = tsd.askstring('Old Password not found', 'Please enter a new
password below', show='*')
        if new_pas == None:
            mess._show(title='No Password Entered', message='Password not
set!! Please try again')
        else:
            tf = open("TrainingImageLabel\psd.txt", "w")
            tf.write(new_pas)
            mess._show(title='Password Registered', message='New password
was registered successfully!!')
            return
    op = (old.get())
    newp= (new.get())
    nnewp = (nnew.get())
    if (op == key):
        if(newp == nnewp):
```

```
            txf = open("TrainingImageLabel\psd.txt", "w")
            txf.write(newp)
        else:
            mess._show(title='Error', message='Confirm new password again!!!')
            return
    else:
        mess._show(title='Wrong Password', message='Please enter correct old
password.')
        return
    mess._show(title='Password Changed', message='Password changed
successfully!!')
    master.destroy()

def change_pass():
    global master
    master = tk.Tk()
    master.geometry("400x160")
    master.resizable(False,False)
    master.title("Change Password")
    master.configure(background="white")
    lbl4 = tk.Label(master,text='    Enter Old
Password',bg='white',font=('times', 12, ' bold '))
    lbl4.place(x=10,y=10)
    global old
    old=tk.Entry(master,width=25 ,fg="black",relief='solid',font=('times', 12, '
bold '),show='*')
    old.place(x=180,y=10)
    lbl5 = tk.Label(master, text='   Enter New Password', bg='white',
font=('times', 12, ' bold '))
    lbl5.place(x=10, y=45)
    global new
    new = tk.Entry(master, width=25, fg="black",relief='solid', font=('times',
12, ' bold '),show='*')
    new.place(x=180, y=45)
    lbl6 = tk.Label(master, text='Confirm New Password', bg='white',
font=('times', 12, ' bold '))
```

```
    lbl6.place(x=10, y=80)
    global nnew
    nnew = tk.Entry(master, width=25, fg="black", relief='solid',font=('times',
12, ' bold '),show='*')
    nnew.place(x=180, y=80)
    cancel=tk.Button(master,text="Cancel", command=master.destroy
,fg="black"  ,bg="red" ,height=1,width=25 , activebackground = "white"
,font=('times', 10, ' bold '))
    cancel.place(x=200, y=120)
    save1 = tk.Button(master, text="Save", command=save_pass, fg="black",
bg="#3ece48", height = 1,width=25, activebackground="white",
font=('times', 10, ' bold '))
    save1.place(x=10, y=120)
    master.mainloop()


def psw():
    assure_path_exists("TrainingImageLabel/")
    exists1 = os.path.isfile("TrainingImageLabel\psd.txt")
    if exists1:
        tf = open("TrainingImageLabel\psd.txt", "r")
        key = tf.read()
    else:
        new_pas = tsd.askstring('Old Password not found', 'Please enter a new
password below', show='*')
        if new_pas == None:
            mess._show(title='No Password Entered', message='Password not
set!! Please try again')
        else:
            tf = open("TrainingImageLabel\psd.txt", "w")
            tf.write(new_pas)
            mess._show(title='Password Registered', message='New password
was registered successfully!!')
            return
    password = tsd.askstring('Password', 'Enter Password', show='*')
    if (password == key):
```

```
        TrainImages()
    elif (password == None):
        pass
    else:
        mess._show(title='Wrong Password', message='You have entered wrong
password')


def clear():
    txt.delete(0, 'end')
    res = "1)Take Images  >>>  2)Save Profile"
    message1.configure(text=res)


def clear2():
    txt2.delete(0, 'end')
    res = "1)Take Images  >>>  2)Save Profile"
    message1.configure(text=res)


def TakeImages():
    check_haarcascadefile()
    columns = ['SERIAL NO.', '', 'ID', '', 'NAME']
    assure_path_exists("StudentDetails/")
    assure_path_exists("TrainingImage/")
    serial = 0
    exists = os.path.isfile("StudentDetails\StudentDetails.csv")
    if exists:
        with open("StudentDetails\StudentDetails.csv", 'r') as csvFile1:
            reader1 = csv.reader(csvFile1)
            for l in reader1:
                serial = serial + 1
        serial = (serial // 2)
        csvFile1.close()
    else:
        with open("StudentDetails\StudentDetails.csv", 'a+') as csvFile1:
```

```
            writer = csv.writer(csvFile1)
            writer.writerow(columns)
            serial = 1
        csvFile1.close()
    Id = (txt.get())
    name = (txt2.get())
    if ((name.isalpha()) or (' ' in name)):
        cam = cv2.VideoCapture(0)
        harcascadePath = "haarcascade_frontalface_default.xml"
        detector = cv2.CascadeClassifier(harcascadePath)
        sampleNum = 0
        while (True):
            ret, img = cam.read()
            gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            faces = detector.detectMultiScale(gray, 1.3, 5)
            for (x, y, w, h) in faces:
                cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
                # incrementing sample number
                sampleNum = sampleNum + 1
                # saving the captured face in the dataset folder TrainingImage
                cv2.imwrite("TrainingImage\ " + name + "." + str(serial) + "." + Id
+ '.' + str(sampleNum) + ".jpg",
                            gray[y:y + h, x:x + w])
                # display the frame
                cv2.imshow('Taking Images', img)
            # wait for 100 miliseconds
            if cv2.waitKey(100) & 0xFF == ord('q'):
                break
            # break if the sample number is morethan 100
            elif sampleNum > 100:
                break
        cam.release()
        cv2.destroyAllWindows()
        res = "Images Taken for ID : " + Id
        row = [serial, '', Id, '', name]
        with open('StudentDetails\StudentDetails.csv', 'a+') as csvFile:
```

```
            writer = csv.writer(csvFile)
            writer.writerow(row)
        csvFile.close()
        message1.configure(text=res)
    else:
        if (name.isalpha() == False):
            res = "Enter Correct name"
            message.configure(text=res)


def TrainImages():
    check_haarcascadefile()
    assure_path_exists("TrainingImageLabel/")
    recognizer = cv2.face_LBPHFaceRecognizer.create()
    harcascadePath = "haarcascade_frontalface_default.xml"
    detector = cv2.CascadeClassifier(harcascadePath)
    faces, ID = getImagesAndLabels("TrainingImage")
    try:
        recognizer.train(faces, np.array(ID))
    except:
        mess._show(title='No Registrations', message='Please Register someone
first!!!')
        return
    recognizer.save("TrainingImageLabel\Trainner.yml")
    res = "Profile Saved Successfully"
    message1.configure(text=res)
    message.configure(text='Total Registrations till now  : ' + str(ID[0]))


def getImagesAndLabels(path):
    # get the path of all the files in the folder
    imagePaths = [os.path.join(path, f) for f in os.listdir(path)]
    # create empth face list
    faces = []
    # create empty ID list
    Ids = []
```

```
    # now looping through all the image paths and loading the Ids and the
images
    for imagePath in imagePaths:
        # loading the image and converting it to gray scale
        pilImage = Image.open(imagePath).convert('L')
        # Now we are converting the PIL image into numpy array
        imageNp = np.array(pilImage, 'uint8')
        # getting the Id from the image
        ID = int(os.path.split(imagePath)[-1].split(".")[1])
        # extract the face from the training image sample
        faces.append(imageNp)
        Ids.append(ID)
    return faces, Ids


def TrackImages():
    check_haarcascadefile()
    assure_path_exists("Attendance/")
    assure_path_exists("StudentDetails/")
    for k in tv.get_children():
        tv.delete(k)
    msg = ''
    i = 0
    j = 0
    recognizer = cv2.face.LBPHFaceRecognizer_create()  #
cv2.createLBPHFaceRecognizer()
    exists3 = os.path.isfile("TrainingImageLabel\Trainner.yml")
    if exists3:
        recognizer.read("TrainingImageLabel\Trainner.yml")
    else:
        mess._show(title='Data Missing', message='Please click on Save Profile
to reset data!!')
        return
    harcascadePath = "haarcascade_frontalface_default.xml"
    faceCascade = cv2.CascadeClassifier(harcascadePath);
```

```
    cam = cv2.VideoCapture(0)
    font = cv2.FONT_HERSHEY_SIMPLEX
    col_names = ['Id', '', 'Name', '', 'Date', '', 'Time']
    exists1 = os.path.isfile("StudentDetails\StudentDetails.csv")
    if exists1:
        df = pd.read_csv("StudentDetails\StudentDetails.csv")
    else:
        mess._show(title='Details Missing', message='Students details are
missing, please check!')
        cam.release()
        cv2.destroyAllWindows()
        window.destroy()
    while True:
        ret, im = cam.read()
        gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
        faces = faceCascade.detectMultiScale(gray, 1.2, 5)
        for (x, y, w, h) in faces:
            cv2.rectangle(im, (x, y), (x + w, y + h), (225, 0, 0), 2)
            serial, conf = recognizer.predict(gray[y:y + h, x:x + w])
            if (conf < 50):
                ts = time.time()
                date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')
                timeStamp =
datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
                aa = df.loc[df['SERIAL NO.'] == serial]['NAME'].values
                ID = df.loc[df['SERIAL NO.'] == serial]['ID'].values
                ID = str(ID)
                ID = ID[1:-1]
                bb = str(aa)
                bb = bb[2:-2]
                attendance = [str(ID), '', bb, '', str(date), '', str(timeStamp)]

            else:
                Id = 'Unknown'
                bb = str(Id)
            cv2.putText(im, str(bb), (x, y + h), font, 1, (255, 255, 255), 2)
```

```
        cv2.imshow('Taking Attendance', im)
        if (cv2.waitKey(1) == ord('q')):
            break
    ts = time.time()
    date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')
    exists = os.path.isfile("Attendance\Attendance_" + date + ".csv")
    if exists:
        with open("Attendance\Attendance_" + date + ".csv", 'a+') as csvFile1:
            writer = csv.writer(csvFile1)
            writer.writerow(attendance)
        csvFile1.close()
    else:
        with open("Attendance\Attendance_" + date + ".csv", 'a+') as csvFile1:
            writer = csv.writer(csvFile1)
            writer.writerow(col_names)
            writer.writerow(attendance)
        csvFile1.close()
    with open("Attendance\Attendance_" + date + ".csv", 'r') as csvFile1:
        reader1 = csv.reader(csvFile1)
        for lines in reader1:
            i = i + 1
            if (i > 1):
                if (i % 2 != 0):
                    iidd = str(lines[0]) + '   '
                    tv.insert('', 0, text=iidd, values=(str(lines[2]), str(lines[4]),
str(lines[6])))
    csvFile1.close()
    cam.release()
    cv2.destroyAllWindows()

############################################ USED STUFFS
############################################

global key
key = ''
```

```
ts = time.time()
date = datetime.datetime.fromtimestamp(ts).strftime('%d-%m-%Y')
day,month,year=date.split("-")

mont={'01':'January',
      '02':'February',
      '03':'March',
      '04':'April',
      '05':'May',
      '06':'June',
      '07':'July',
      '08':'August',
      '09':'September',
      '10':'October',
      '11':'November',
      '12':'December'
      }

##########################################GUI FRONT-END
###############################################

window = tk.Tk()
window.geometry("1280x720")
window.resizable(True,False)
window.title("Attendance System")
window.configure(background='#FFFFFF')

frame1 = tk.Frame(window, bg="#111111")
frame1.place(relx=0.11, rely=0.17, relwidth=0.39, relheight=0.80)

frame2 = tk.Frame(window, bg="#000000")
frame2.place(relx=0.51, rely=0.17, relwidth=0.38, relheight=0.80)

message3 = tk.Label(window, text="Face Recognition Based Attendance
System" ,fg="white",bg="#262523" ,width=55 ,height=1,font=('times', 29, '
bold '))
```

```
message3.place(x=10, y=10)

frame3 = tk.Frame(window, bg="#c4c6ce")
frame3.place(relx=0.52, rely=0.09, relwidth=0.09, relheight=0.07)

frame4 = tk.Frame(window, bg="#c4c6ce")
frame4.place(relx=0.36, rely=0.09, relwidth=0.16, relheight=0.07)

datef = tk.Label(frame4, text = day+"-"+mont[month]+"-"+year+"  |  ",
fg="orange",bg="#262523" ,width=55 ,height=1,font=('times', 22, ' bold '))
datef.pack(fill='both',expand=1)

clock = tk.Label(frame3,fg="orange",bg="#262523" ,width=55
,height=1,font=('times', 22, ' bold '))
clock.pack(fill='both',expand=1)
tick()

head2 = tk.Label(frame2, text="                    For New Registrations
", fg="black",bg="#3ece48" ,font=('times', 17, ' bold ') )
head2.grid(row=0,column=0)

head1 = tk.Label(frame1, text="                    For Already Registered
", fg="black",bg="#3ece48" ,font=('times', 17, ' bold ') )
head1.place(x=0,y=0)

lbl = tk.Label(frame2, text="Enter ID",width=20  ,height=1  ,fg="black"
,bg="#00aeff" ,font=('times', 17, ' bold ') )
lbl.place(x=80, y=55)

txt = tk.Entry(frame2,width=32 ,fg="black",font=('times', 15, ' bold '))
txt.place(x=30, y=88)

lbl2 = tk.Label(frame2, text="Enter Name",width=20  ,fg="black"
,bg="#00aeff" ,font=('times', 17, ' bold '))
lbl2.place(x=80, y=140)
```

```
txt2 = tk.Entry(frame2,width=32 ,fg="black",font=('times', 15, ' bold ')  )
txt2.place(x=30, y=173)

message1 = tk.Label(frame2, text="1)Take Images  >>>  2)Save Profile"
,bg="#00aeff" ,fg="black"  ,width=39 ,height=1, activebackground =
"yellow" ,font=('times', 15, ' bold '))
message1.place(x=7, y=230)

message = tk.Label(frame2, text="" ,bg="#00aeff" ,fg="black"
,width=39,height=1, activebackground = "yellow" ,font=('times', 16, ' bold '))
message.place(x=7, y=450)

lbl3 = tk.Label(frame1, text="Attendance",width=20  ,fg="black"
,bg="#00aeff"  ,height=1 ,font=('times', 17, ' bold '))
lbl3.place(x=100, y=115)

res=0
exists = os.path.isfile("StudentDetails\StudentDetails.csv")
if exists:
    with open("StudentDetails\StudentDetails.csv", 'r') as csvFile1:
        reader1 = csv.reader(csvFile1)
        for l in reader1:
            res = res + 1
    res = (res // 2) - 1
    csvFile1.close()
else:
    res = 0
message.configure(text='Total Registrations till now  : '+str(res))

#####################MENUBAR################################

menubar = tk.Menu(window,relief='ridge')
filemenu = tk.Menu(menubar,tearoff=0)
filemenu.add_command(label='Change Password', command = change_pass)
filemenu.add_command(label='Contact Us', command = contact)
filemenu.add_command(label='Exit',command = window.destroy)
```

```
menubar.add_cascade(label='Help',font=('times', 29, ' bold '),menu=filemenu)

############TREEVIEW ATTENDANCE TABLE####################

tv= ttk.Treeview(frame1,height =13,columns = ('name','date','time'))
tv.column('#0',width=82)
tv.column('name',width=130)
tv.column('date',width=133)
tv.column('time',width=133)
tv.grid(row=2,column=0,padx=(0,0),pady=(150,0),columnspan=4)
tv.heading('#0',text ='ID')
tv.heading('name',text ='NAME')
tv.heading('date',text ='DATE')
tv.heading('time',text ='TIME')

######################SCROLLBAR################################

scroll=ttk.Scrollbar(frame1,orient='vertical',command=tv.yview)
scroll.grid(row=2,column=4,padx=(0,100),pady=(150,0),sticky='ns')
tv.configure(yscrollcommand=scroll.set)

###################### BUTTONS
##################################

clearButton = tk.Button(frame2, text="Clear", command=clear  ,fg="black"
,bg="#ea2a2a"  ,width=11 ,activebackground = "white" ,font=('times', 11, '
bold '))
clearButton.place(x=335, y=86)
clearButton2 = tk.Button(frame2, text="Clear", command=clear2  ,fg="black"
,bg="#ea2a2a"  ,width=11 , activebackground = "white" ,font=('times', 11, '
bold '))
clearButton2.place(x=335, y=172)
takeImg = tk.Button(frame2, text="Take Images", command=TakeImages
,fg="white"  ,bg="blue"  ,width=34  ,height=1, activebackground = "white"
,font=('times', 15, ' bold '))
takeImg.place(x=30, y=300)
```

```
trainImg = tk.Button(frame2, text="Save Profile", command=psw ,fg="white"
,bg="blue"  ,width=34  ,height=1, activebackground = "white" ,font=('times',
15, ' bold '))
trainImg.place(x=30, y=380)
trackImg = tk.Button(frame1, text="Take Attendance",
command=TrackImages  ,fg="black"  ,bg="yellow"  ,width=35  ,height=1,
activebackground = "white" ,font=('times', 15, ' bold '))
trackImg.place(x=30,y=50)
quitWindow = tk.Button(frame1, text="Quit", command=window.destroy
,fg="black"  ,bg="red"  ,width=35 ,height=1, activebackground = "white"
,font=('times', 15, ' bold '))
quitWindow.place(x=30, y=450)

####################END ########################################

window.configure(menu=menubar)
window.mainloop()
```

APPENDIX B

OUTPUT SCREENSHOTS:

MAIN SCREEN

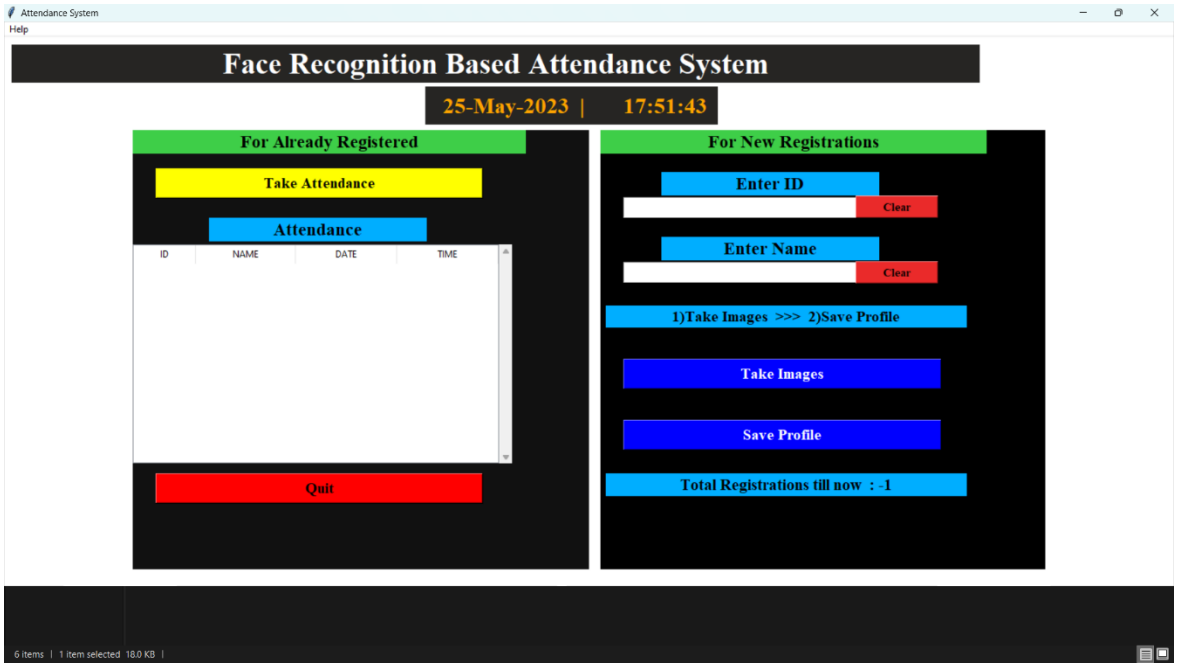

Fig 1.5 main page

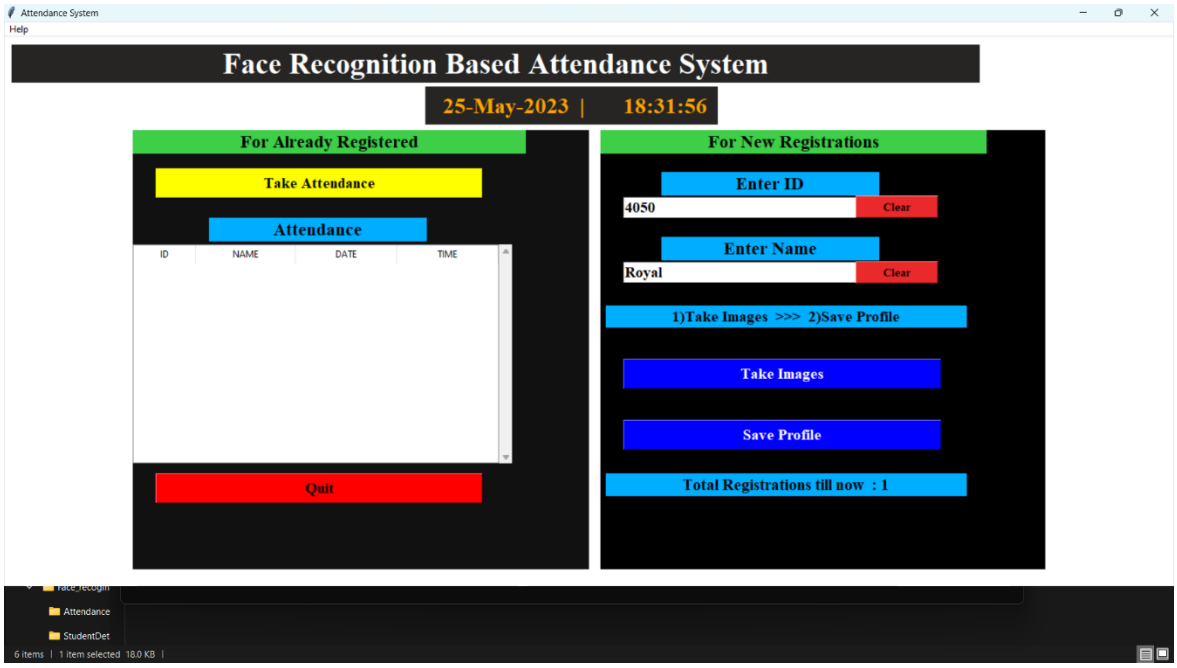

Fig 1.6 Registration for new student

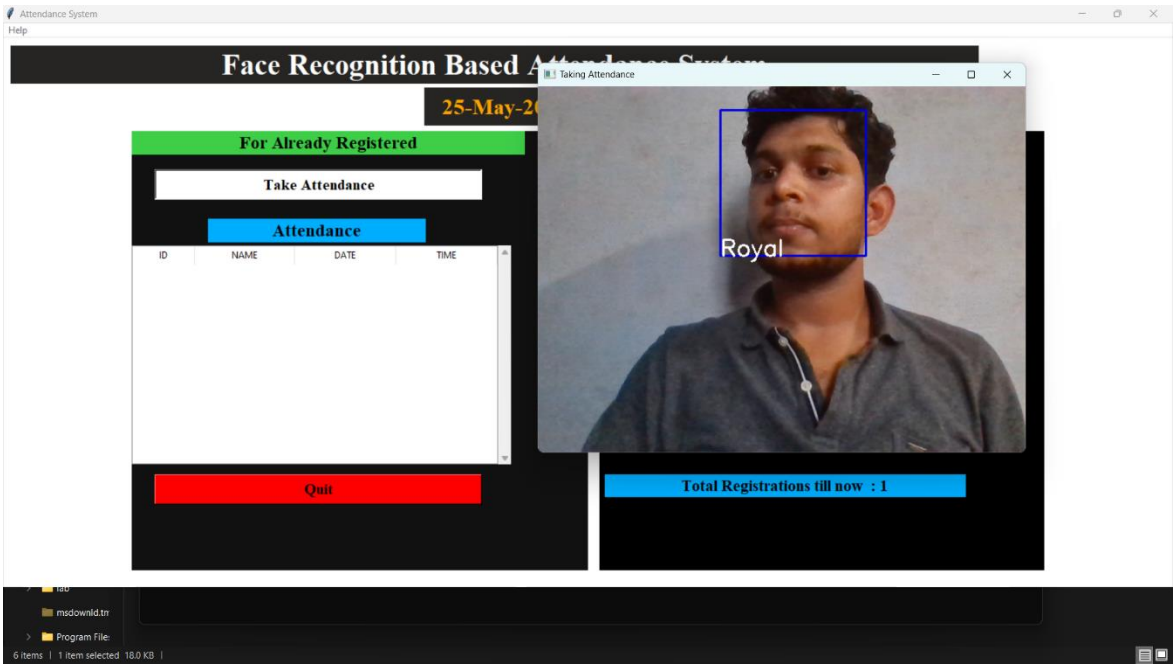

Fig 1.7 Face Recognition in Attendance

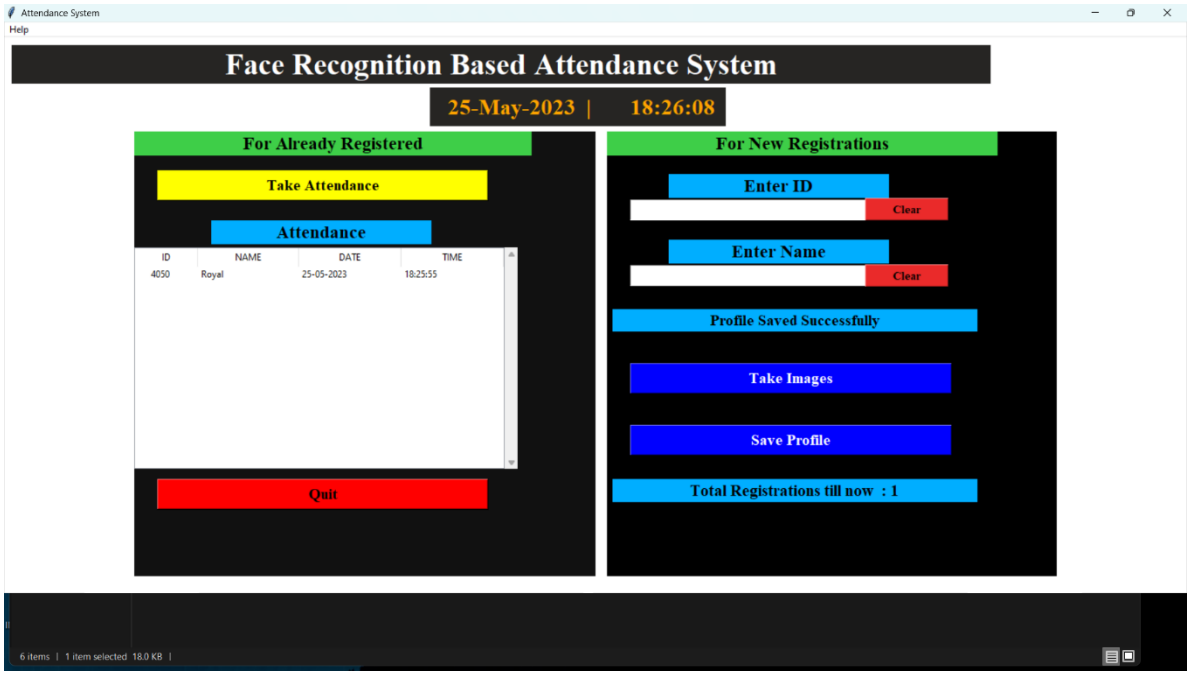

Fig 1.8 Attendance marked

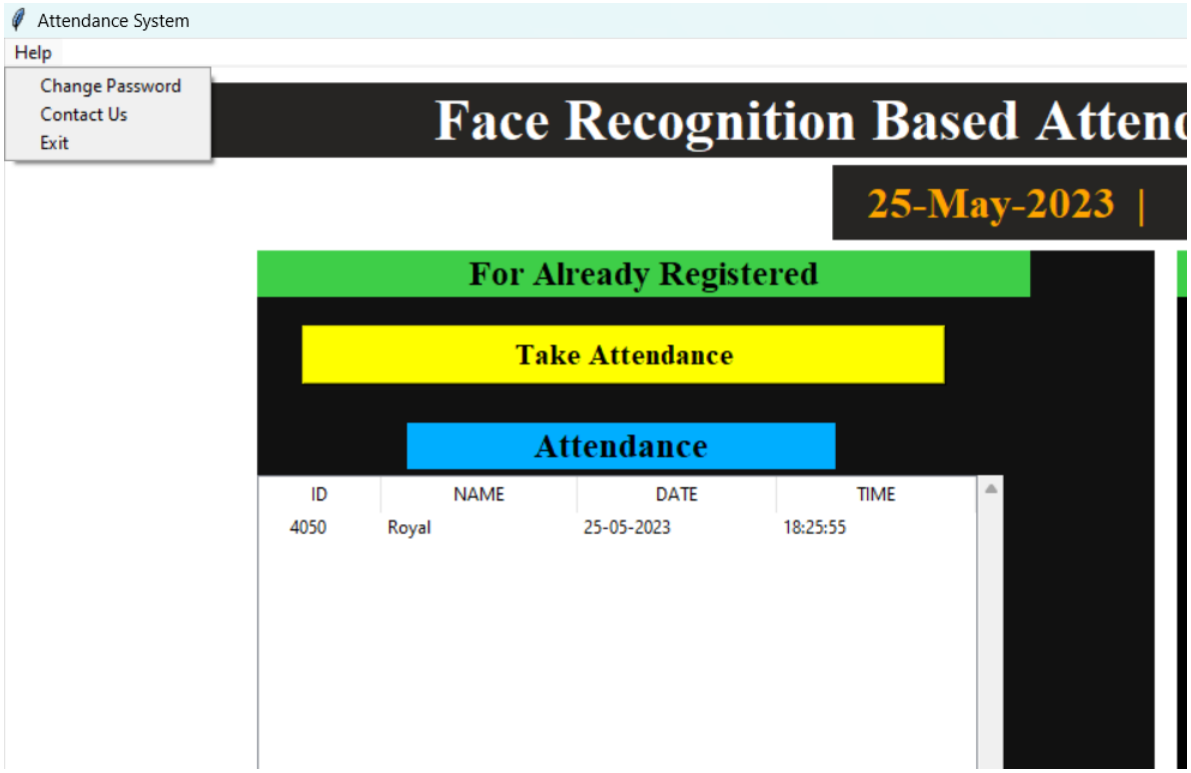

Fig 1.9 Help option in menubar

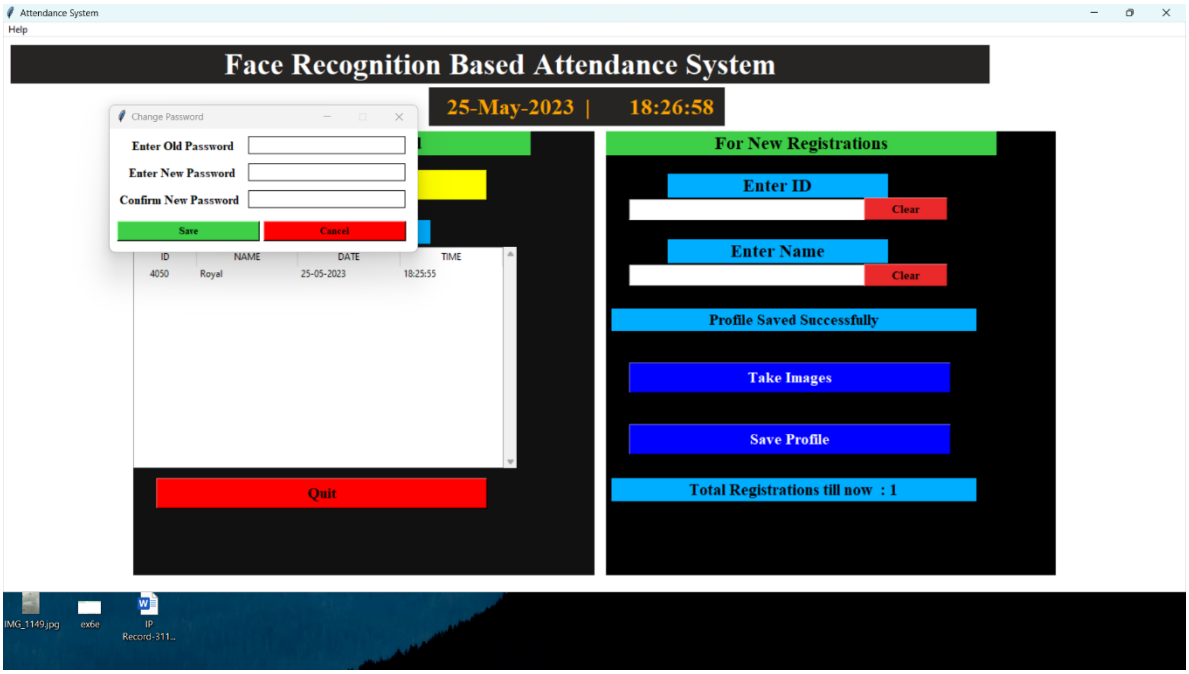

Fig 1.10 Change password option

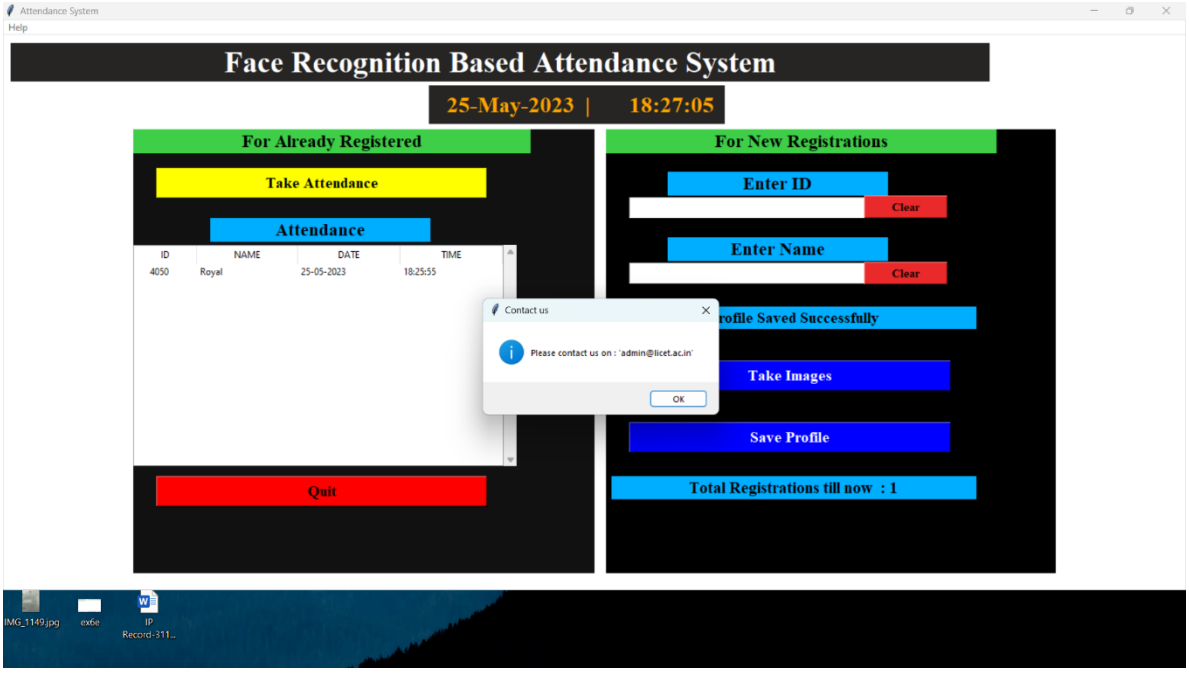

Fig 1.11 Help to contact the admin

REFERENCES

[1] Tarik Hachad, Abdelalim Sadiq, Fadoua Ghanimi. Student's attendance management using deep facial recognition. (January 2021).

[2] Dr Aruna Bhat, Shivam Rustagi, Shivi R Purwaha, Shubhang Singhal. Deep-learning based group-photo Attendance System using One Shot Learning. (July 2020)

[3] Maria Ali, Hafiz Usman Zahoor, Ans Ali, Muhammad Ali Qureshi. Smart Multiple Attendance System through Single Image. (November 2020)

[4] Jaehoon (Paul) Jeong, Minho Kim, Yeonghyeon Lee, and Patrick Lingga. IAAS: IoT-Based Automatic Attendance System with Photo Face Recognition in Smart Campus. (December 2020)

[5] oumitra Chowdhury , Sudipta Nath , Ashim Dey and Annesha Das. Development of an Automatic Class Attendance System using CNN-based Face Recognition. (December 2020)

[6] Shreyak Sawhney, Karan Kacker ,Samyak Jain, Shailendra Narayan Singh , Rakesh Garg5. Real-Time Smart Attendance System using Face Recognition Techniques. (January 2019)

[7] Royston Dmello, Sai Yerremreddy, Samriddha Basu, Tejas Bhitle, Yash Kokate, and Prachi Gharpure. Automated Facial Recognition Attendance System Leveraging IoT Cameras. (January 2019)

[8] Dr. Jayakumar Kaliappan, Jain Shreyansh, Shanmuga Sundari. P. Surveillance Camera using Face Recognition for automatic Attendance feeder and Energy conservation in classroom. (March 2019)

[9] Raisha Shrestha, Saurav Man Pradhan, Rahul Karn, Suraj Shrestha. Attendance and Security Assurance using Image Processing. (February 2018)

[10] Christopher Chun Ki Chan, Chih-Cheng Chen. Continuous Real-time Automated Attendance System using Robust C2D-CNN. (August 2020)