Face Recognition Based Attendance Monitoring System

Real-Time Project Report

Submitted in partial fulfillment of the requirements for the award of the Degree

of

Bachelor of Technology (B.Tech)

in

INFORMATION TECHNOLOGY

By

A. Karunakar	22AG1A1203
B. Rakesh Naidu	22AG1A1204
G. Sindhu Vasavi	22AG1A1221

Under the Esteemed Guidance of
Mr. G. Prasad
Assistant Professor

Department of Information Technology

ACE ENGINEERING COLLEGE

An Autonomous Institution

All the courses are Accredited by NBA and NAAC with A Grade

(Affiliated to Jawaharlal Nehru Technological University, Hyderabad, Telangana)

Ankushapur, Ghatkesar, Medchal,Hyderabad - 501 301

JULY 2024

ACE

Engineering College

An Autonomous Institution

All the courses are Accredited by NBA and NAAC with A Grade

(Affiliated to Jawaharlal Nehru Technological University, Hyderabad, Telangana)

Website: www.aceec.ac.in E-mail: info@aceec.ac.in

CERTIFICATE

This is to certify that the Real-Time Project work entitled **"Face Recognition Based Attendance Monitoring System"** is being submitted by **A. Karunakar (22AG1A1203), B. Rakesh Naidu (22AG1A1204), G. Sindhu Vasavi (22AG1A1221),** in partial fulfillment for the award of Degree of **BACHELOR OF TECHNOLOGY** in **INFORMATION TECHNOLOGY** to the Jawaharlal Nehru Technological University, Hyderabad during the academic year 2023-24 is a record of bonafide work carried out by them under our guidance and supervision.

The results embodied in this report have not been submitted by the student to any other University or Institution for the award of any degree or diploma.

Internal Guide	**Head of the Department**
Mr. G. Prasad	**Prof. K. JAYA BHARATHI**
Assistant Professor	**Professor and Head**
	Dept. of IT

ACKNOWLEDGEMENT

We would like to express our gratitude to all the people behind the screen who have helped us transform our idea into a real time application.

We would like to express our heart-felt gratitude to our parents without whom we would not have been privileged to achieve and fulfill our dreams.

A special thanks to our Secretary, **Prof. Y. V. GOPALA KRISHNA MURTHY** and Joint Secretary, **Mrs. M. PADMAVATHI** for having founded such an esteemed institution. We are also grateful to our beloved principal, **Dr. B.L. RAJU** for permitting us to carry out this project.

We profoundly thank **Prof. K. JAYA BHARATHI**, Head of the Department of Information Technology who has been an excellent guide and also a great source of inspiration to our work.

We are very thankful to our internal guide **Mr. G. Prasad**, **Assistant Professor** who has been given continuous support for the completion of our project work.

The satisfaction and euphoria that accompany the successful completion of the task would be great, but incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crown all the efforts with success. In this context, we would like to thank all the other staff members, both teaching and non-teaching, who have extended their timely help and eased our task.

A. Karunakar (22AG1A1203)
B. RakeshNaidu (22AG1A1204)
G. SindhuVasavi (22AG1A1221)

DECLARATION

We hereby declare that Real-Time Project entitled "**Face Recognition Based Attendance Monitoring System**" which is being submitted in the partial fulfillment of the requirements for the award of degree of Bachelor of Technology in Information Technology. This dissertation is our original work and the project has not formed the basis for the award of any degree, associate ship, fellowship or any other similar titles and no part of it has been published or sent for the publication at the time of submission.

A. Karunakar	(22AG1A1203)
B. Rakesh Naidu	(22AG1A1204)
G. Sindhu Vasavi	(22AG1A1221)

ABSTRACT

The "**Face Recognition Based Attendance Monitoring System** " project aims to streamline and automate the process of student attendance in educational institutions. The system leverages advanced face recognition technology to accurately identify and record the presence of students.

During registration, the system captures and stores many images of each student, associating them with their respective roll numbers and names. A secure password, known only to the faculty, protects the registration process. For daily attendance, the system compares real-time images of students with the stored data to verify their identity. Attendance records, including date, time, roll number, and name, are automatically logged.

All student details and face images are securely maintained in an Excel sheet and project file. This approach enhances accuracy, saves time, and minimizes manual errors, making the attendance process more efficient and reliable.

The outcomes uncover that the proposed framework can be utilized for face detection even from low quality pictures and shows incredible execution level. At last, the data that will be shown alongside recognized photograph has been put away on database. This concept has a higher scope on security and surveillance projects and various operation.

INDEX

LIST OF FIGURES

LIST OF ABBREVATIONS

1. OpenCV : Open-Source Computer Vision
2. PIL: Python Imaging Library
3. GUI: Graphical User Interface.
4. CSV: Comma Separated Values

CHAPTER 1

INTRODUCTION

The "Attendance using Face Recognition" project addresses the need for an efficient, accurate, and automated attendance management system in educational institutions. Traditional methods of taking attendance are often time-consuming, prone to errors, and susceptible to manipulation. By leveraging advanced face recognition technology, this project aims to modernize the attendance process, ensuring reliable identification of students with minimal manual intervention.

1.1. Purpose:

The purpose of the "Attendance using Face Recognition" project is to develop an automated, efficient, and accurate system for managing student attendance in educational institutions.
The primary objectives are:

1. Efficiency: To streamline the attendance process, reducing the time and effort required by faculty and staff.

2. Accuracy: To eliminate errors associated with manual attendance tracking, ensuring reliable and precise records.

3. Automation: To leverage face recognition technology for seamless, real-time identification and logging of student attendance.

4. Security: To enhance the security and privacy of attendance data through secure registration and data storage practices.

5. Scalability: To create a system that can be easily scaled and adapted to institutions of various sizes and needs.

6. User-Friendliness: To provide an intuitive and accessible interface for both administrators and users, facilitating easy adoption and use.

1.2. Scope:

The scope of the "Attendance using Face Recognition" project encompasses the development, implementation, and maintenance of a comprehensive attendance management system tailored for educational institutions. Key aspects include:

1. **System Development**:

- **Registration Module:** Capture and store multiple images of each student, associating them with unique identifiers such as roll numbers and names.
- **Attendance Module:** Real-time image capture and face recognition to verify student identities and log attendance details (date, time, roll number, and name).
- **User Interface:** Design a user-friendly interface for both administrators (faculty) and users (students) for easy registration and attendance tracking.

2. Data Management:

- **Storage:** Secure storage of student images and attendance records in databases (e.g., MySQL, MongoDB) and Excel sheets.
- **Security:** Implement password protection and encryption to ensure data privacy and integrity.

3. Integration:

- **Compatibility:** Ensure compatibility with existing administrative systems and software used by the institution.
- **Scalability:** Design the system to handle a large number of students and expand as needed.

4. Maintenance and Support:

- **Regular Updates:** Provide system updates to enhance functionality and security.
- **Technical Support:** Offer ongoing technical support to address any issues and ensure smooth operation.

5. Potential Extensions:

- **Analytics:** Incorporate features for generating attendance reports and analytics.
- **Notifications:** Add real-time notifications for students and faculty regarding attendance status.

1.3. Real-Time usage & Applications:

The "Attendance using Face Recognition" system has broad real-time usage and applications, particularly in educational institutions and other organizational settings:

1. **Educational Institutions:**
    - **Schools and Colleges:** Automates daily attendance tracking, reducing the time teachers spend on roll calls and minimizing errors.

- **Examination Halls:** Verifies student identities during exams to prevent impersonation and ensure security.
- **Events and Activities:** Tracks attendance for extracurricular activities, meetings, and other events.

2. **Workplaces:**
    - **Employee Attendance:** Monitors employee attendance and timekeeping, providing accurate records for payroll and compliance.
    - **Access Control:** Enhances security by using face recognition for access to restricted areas within an organization.
3. **Conferences and Seminars:**
    - **Participant Tracking:** Registers and tracks attendance of participants in real time, providing organizers with accurate attendance data.
4. **Healthcare Facilities:**
    - **Staff Attendance:** Manages attendance of healthcare workers, ensuring proper staffing and compliance with labor regulations.
    - **Patient Identification:** Verifies patient identities to prevent mix-ups and enhance security.
5. **Public and Government Services:**
    - **Conference and Seminar Planners:** Professionals who need accurate attendance tracking for participants in real-time.
    - **Visitor Management:** Tracks and manages visitors in government buildings, enhancing security and record-keeping

1.4. Target Audience:

The primary target audience for the "Attendance using Face Recognition" system includes:

1. **Educational Institutions:**
    - **Schools, Colleges, and Universities:** Administrators and faculty looking to streamline and automate student attendance processes.
    - **Training Centers and Institutes:** Organizations that conduct regular classes and need efficient attendance tracking systems.

2. **Corporate Sector:**
    - **Human Resources Departments:** Companies seeking to improve employee attendance tracking and time management.
    - **Security Teams:** Organizations aiming to enhance workplace security and access control through automated face recognition.
3. **Event Organizers:**
    - **Conference and Seminar Planners:** Professionals who need accurate attendance tracking for participants in real-time.
    - **Workshop and Training Event Coordinators:** Facilitators requiring efficient registration and attendance management.
4. **Healthcare Facilities:**
    - **Hospitals and Clinics:** Administrators looking to manage staff attendance and enhance patient identification processes.
5. **Government Agencies:**
    - **Public Service Departments:** Offices that manage large numbers of visitors and employees, requiring efficient attendance and access control.
6. **Other Organizations:**
    - **Non-profits and NGOs:** Organizations needing to track attendance for events, training sessions, and meetings.

- **Recreational and Sports Facilities:** Clubs and centers that require efficient management of member attendance and access control.

CHAPTER 2

LITERATURE REVIEW

- Real-Time Face Recognition for Attendance Monitoring

Author**:** N. Kumar, V. Ravi, A. Tripathi
Year**:** 2017
Technologies: Dlib, CNN, Python
Summary: Develops a real-time attendance monitoring system leveraging Convolutional Neural Networks (CNN) and the Dlib library for high accuracy in face recognition.
By leveraging advanced face recognition algorithms, the system can detect and recognize faces in real-time, ensuring immediate and precise attendance logging. This approach minimizes errors and reduces the possibility of fraud.

- Automated Attendance System Using Face Recognition

Author: P. Arul, K. Devi
Year: 2019
Technologies: LBPH , OpenCV, Raspberry Pi
Summary: It captures images of individuals, detects their faces, and matches them with stored facial images in a database to mark attendance automatically. This approach not only automates the attendance process but also ensures high accuracy and convenience, eliminating the need for manual sign-ins or physical interaction, such as swiping cards or fingerprint scanning. The system is particularly useful in educational institutions and workplaces, offering a non-intrusive and efficient method for maintaining real-time attendance records. However, challenges such as initial setup costs and environmental factors like lighting conditions can affect performance.

Exisiting System:

Smart attendance systems utilizing face recognition technology integrate hardware, software, and AI algorithms for robust attendance tracking. The hardware includes high-resolution cameras equipped with infrared sensors and wide-angle lenses, ensuring clear image capture in varying lighting and angles. Additional components like infrared illuminators enhance performance in challenging conditions.

On the software side, facial recognition algorithms analyze live video feeds to extract unique facial features, creating digital faceprints for each individual. These faceprints are compared with pre-registered templates stored in a database for identification. Advanced image processing techniques like normalization and feature extraction enhance accuracy by compensating for lighting variations, facial expressions, and occlusions (e.g., glasses or hats).

The system provides real-time monitoring, attendance logs, and analytics through a centralized interface. Supervisors can monitor entry and exit events, generate reports, and gain insights into attendance patterns and anomalies. Biometric authentication options such as fingerprint or iris scanning add layers of security, ensuring the integrity of attendance records.

From a user perspective, enrollment is user-friendly, allowing individuals to easily register their facial biometrics and link them to unique identifiers like employee IDs or student numbers. Subsequent attendance tracking requires minimal effort as users simply present their face for recognition, eliminating the need for physical badges or manual entry.

Overall, smart attendance systems using face recognition technology offer efficient, secure, and scalable solutions for attendance management across educational institutions, corporate offices, and public facilities.

CHAPTER 3

SOFTWARE REQUIREMENT ANALYSIS

Conducting a software requirement analysis (SRA) is a pivotal step in the development of a face recognition-based attendance monitoring system. This process involves gathering and defining both functional and non-functional requirements, ensuring that the system meets user needs and performs efficiently. By providing a comprehensive understanding of what the system should do, how it should perform, and the constraints under which it must operate, the SRA lays the foundation for a successful project.

3.1 Functional Requirements

The functional requirements for a face recognition-based attendance monitoring system cover several key areas to ensure the system performs its intended tasks efficiently and accurately.

1.**Face Recognition**:

- The system should capture and recognize faces in real-time.
- It should accurately identify registered users and differentiate between them.
- The system should handle multiple users simultaneously, especially during class start times.

2.**Attendance Logging**:

- The system should automatically log attendance when a face is recognized.
- It should store the date, time, and identity of the recognized person.
- There should be an option for manual attendance marking in case of system failure.

3.**Database Management**:

- The system should maintain a secure database of user information, attendance records, and facial data.

4.**Reporting**:

- The system should generate attendance reports based on various criteria (e.g., date, user, class).

- It should support exporting reports in multiple formats (e.g., PDF, Excel).

3.2 Non-Functional Requirements:

Non-functional requirements ensure that the system not only functions correctly but also performs efficiently, securely, and is user-friendly.

1.**Security**:

- The system should ensure data encryption for stored user information and attendance records.
- It should implement secure authentication mechanisms to prevent unauthorized access.

2.**Scalability**:

- The system should be scalable to accommodate an increasing number of users and records without performance degradation.
- It should support both horizontal and vertical scaling options.

3.Maintainability:

- The system should have a modular architecture to facilitate easy updates and maintenance.
- It should include detailed documentation for users and developers.

CHAPTER 4

SOFTWARE DESIGN

The software design aspects of a Face Recognition Attendance System. This innovative system leverages facial recognition technology to automate attendance tracking, making it efficient and accurate. Here are the key points:

Project Overview:

The System replaces manual attendance processes with computer vision technology. It captures attendance data in real-time, eliminating the need for manual note-taking and paperwork. The system identifies individuals by analyzing their facial features and records attendance details.

Face Detection:

Face detection is the initial step in recognizing faces. Algorithms (such as those from the dlib library) analyze images or video frames to locate and extract facial attributes (eyes, nose, mouth, face shape). Computer vision techniques ensure accurate face detection even in complex scenarios.

Face Recognition:

After detecting faces, the system proceeds to face recognition. It compares detected faces with pre-existing data (such as a database of known individuals) to determine identity. The face recognition library combines OpenCV for image processing.

Encoding and Learning:

The system generates encodings for each face. These encodings represent unique features extracted from facial images. Training involves feeding labeled images to the algorithm to learn patterns and relationships between facial features and encodings.

Implementation:

Use Python and OpenCV for building facial recognition algorithms. Create face databases to pump data into the recognizer algorithm. During attendance sessions, compare detected faces against the database to identify individuals. Automatically record attendance details (names, date, time) in an Excel sheet.

CHAPTER 5

SOFTWARE AND HARDWARE REQUIREMENT

The "Attendance using Face Recognition" system requires specific software and hardware components for optimal performance:

5.1 Software Requirements:

- Programming language: Python 3.10
- Face Recognition Library: OpenCV, PIL
- Data analysis libraries: pandas, numPy
- Operating System : windows 10 Pro

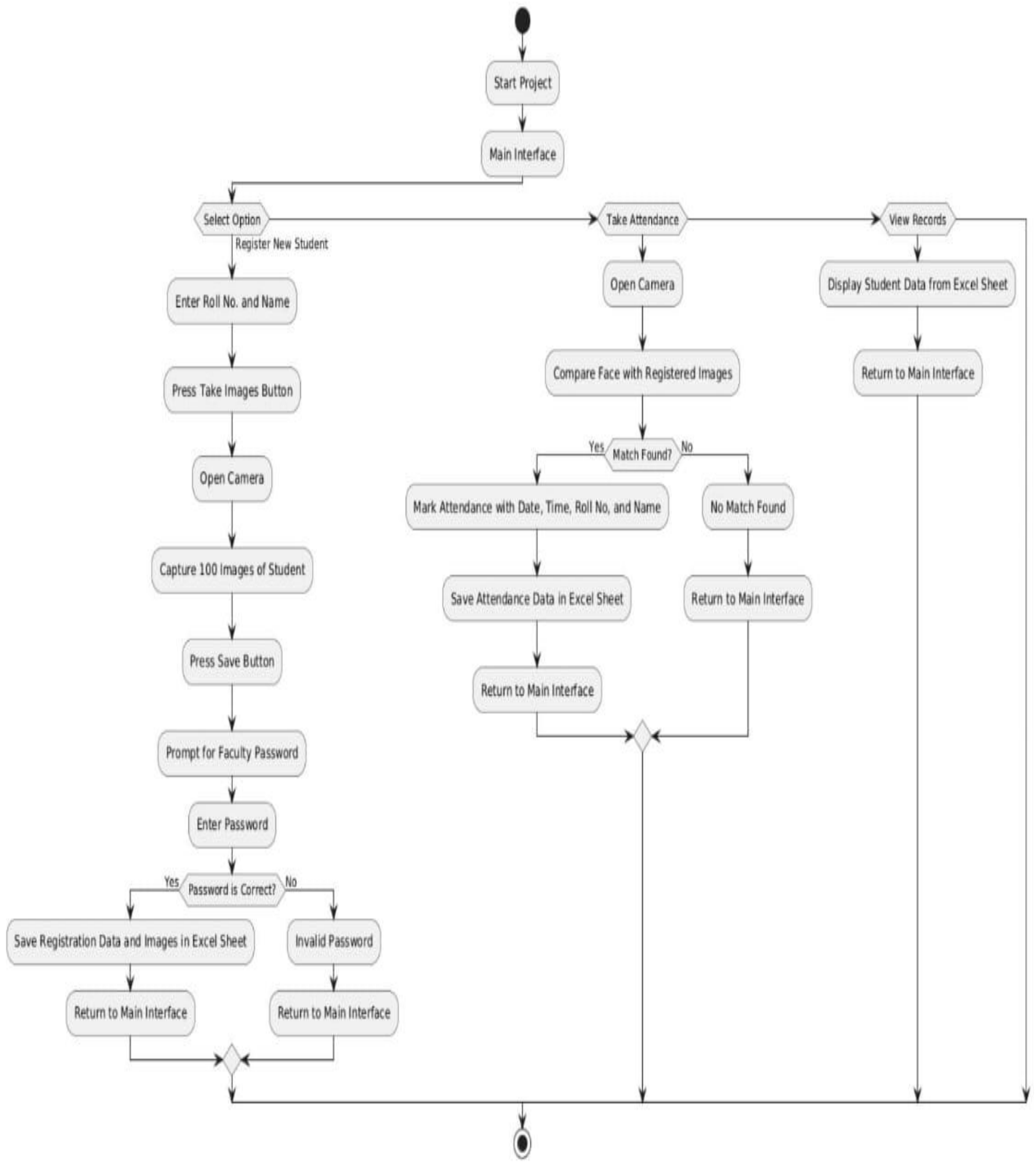
5.2 Hardware Requirements:

- Processor: Intel core i7
- RAM: 8GB
- Storage: 2 TB HDD

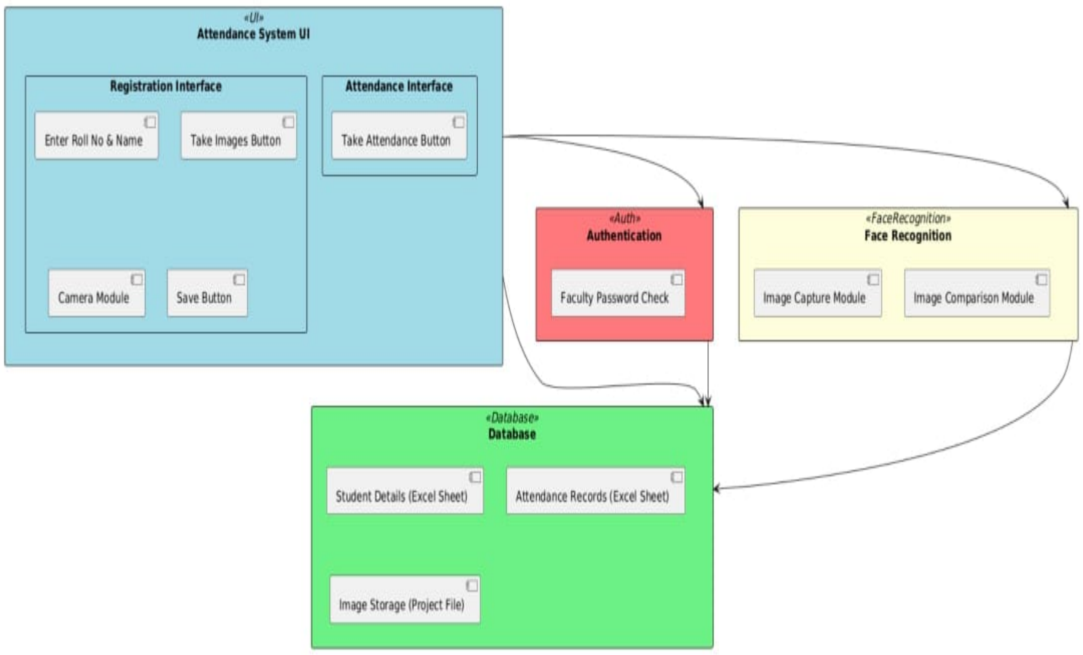
CHAPTER 6

ARCHITECTURE AND FLOW DIAGRAM

6.1 Flow Diagram:



6.2 Architecture Diagram:



CHAPTER 7

IMPLEMENTATION (Main Code)

```
import tkinter as tk
from tkinter import ttk
from tkinter import messagebox as mess
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 :
'sindhuvasavi@gmail.com' ")


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=('comic',12,' bold '))
    lbl4.place(x=10,y=10)
    global old

old=tk.Entry(master,width=25,fg="black",relief='solid',font=('comic',12,'
bold '),show='*')
    old.place(x=180,y=10)
    lbl5 = tk.Label(master, text='   Enter New Password', bg='white',
font=('comic',12,' bold '))
    lbl5.place(x=10, y=45)
    global new
    new = tk.Entry(master, width=25, fg="black",relief='solid',
font=('comic',12,' bold '),show='*')
    new.place(x=180, y=45)
    lbl6 = tk.Label(master, text='Confirm New Password', bg='white',
font=('comic',12,' bold '))
    lbl6.place(x=10, y=80)
    global nnew
    nnew = tk.Entry(master, width=25, fg="black",
relief='solid',font=('comic',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=('comic',10,' bold '))
    cancel.place(x=200, y=120)
    save1 = tk.Button(master, text="Save", command=save_pass, fg="black",
bg="#00fcca", height = 1,width=25, activebackground="white",
font=('comic',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 (' 'inname)):
        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='#B0C4DE')

frame1 = tk.Frame(window, bg="#B2BEB5")
frame1.place(relx=0.11, rely=0.17, relwidth=0.39, relheight=0.80)

frame2 = tk.Frame(window, bg="#B2BEB5")
frame2.place(relx=0.51, rely=0.17, relwidth=0.38, relheight=0.80)

message3 = tk.Label(window, text="Face Recognition Based Attendance
Monitoring System",fg="black",width=55,height=1,font=('comic',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="#ff61e5",bg="#2d420a",width=55,height=1,font=('comic',22,' bold '))
```

```
datef.pack(fill='both',expand=1)

clock =
tk.Label(frame3,fg="#ff61e5",bg="#2d420a",width=55,height=1,font=('comic',2
2,' bold '))
clock.pack(fill='both',expand=1)
tick()

head2 = tk.Label(frame2, text="                       For New Registrations
", fg="black",bg="#00fcca",font=('comic',17,' bold ') )
head2.grid(row=0,column=0)

head1 = tk.Label(frame1, text="                       For Already
Registered                       ",
fg="black",bg="#00fcca",font=('comic',17,' bold ') )
head1.place(x=0,y=0)

lbl = tk.Label(frame2, text="Enter
ID",width=20,height=1,fg="black",bg="#B2BEB5",font=('comic',17,' bold ') )
lbl.place(x=80, y=55)

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

lbl2 = tk.Label(frame2, text="Enter
Name",width=20,fg="black",bg="#B2BEB5",font=('comic',17,' bold '))
lbl2.place(x=80, y=140)

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

message1 = tk.Label(frame2, text="1)Take Images  >>>  2)Save
Profile",bg="#c79cff",fg="black",width=39,height=1, activebackground =
"#3ffc00",font=('comic',15,' bold '))
message1.place(x=7, y=230)

message = tk.Label(frame2,
text="",bg="#c79cff",fg="black",width=39,height=1, activebackground =
"#3ffc00",font=('comic',16,' bold '))
message.place(x=7, y=450)

lbl3 = tk.Label(frame1,
text="Attendance",width=20,fg="black",bg="#c79cff",height=1,font=('comic',1
7,' 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=('comic',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="#ff7221",width=11,activebackground =
"white",font=('comic',11,' bold '))
clearButton.place(x=335, y=86)
clearButton2 = tk.Button(frame2, text="Clear", command=clear2
,fg="black",bg="#ff7221",width=11, activebackground =
"white",font=('comic',11,' bold '))
clearButton2.place(x=335, y=172)
takeImg = tk.Button(frame2, text="Take Images", command=TakeImages
,fg="white",bg="#6d00fc",width=34,height=1, activebackground =
"white",font=('comic',15,' bold '))
takeImg.place(x=30, y=300)
trainImg = tk.Button(frame2, text="Save Profile", command=psw
,fg="white",bg="#6d00fc",width=34,height=1, activebackground =
"white",font=('comic',15,' bold '))
trainImg.place(x=30, y=380)
trackImg = tk.Button(frame1, text="Take Attendance", command=TrackImages
,fg="black",bg="#3ffc00",width=35,height=1, activebackground =
"white",font=('comic',15,' bold '))
trackImg.place(x=30,y=50)
quitWindow = tk.Button(frame1, text="Quit", command=window.destroy
,fg="black",bg="#eb4600",width=35,height=1, activebackground =
"white",font=('comic',15,' bold '))
quitWindow.place(x=30, y=450)

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

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

CHAPTER 8
TEST CASES

1. **Unit Testing:**

   Unit tests focus on individual components or functions within the system. For face recognition, test individual modules such as face detection, face recognition, and attendance logging.

   Example test cases:

   - Verify that the face detection module correctly identifies faces in sample images.
   - Ensure that the face recognition module accurately matches known faces from the database.

2. **Integration Testing:**

   Integration tests check how different components work together. Test the interaction between face detection, recognition, and attendance logging.

   Example test cases:

   - Validate that the attendance logging module records accurate details when a face is recognized.
   - Confirm that the system handles real-time recognition during attendance sessions.

3. **Boundary Testing:**

   Boundary tests explore system limits and edge cases.

   Example test cases:

   - Test the system's behavior when it encounters low-light conditions or partial face visibility.
   - Verify how the system handles extreme angles or unusual facial expressions.

4. **Performance Testing:**

   Performance tests assess system responsiveness, scalability, and resource usage.

   Example test cases:

- Measure the system’s response time during face recognition.
- Evaluate memory usage and CPU load during peak attendance sessions.

5. **Accuracy Testing:**

   Accuracy tests focus on the correctness of face recognition. Use a dataset with known faces and compare system results.

   Example test cases:

   - Calculate the system’s accuracy (true positive rate) in recognizing enrolled students.
   - Check the system’s false positive rate (identifying non-enrolled individuals).

6. **Robustness Testing:**

   Robustness tests assess system stability under adverse conditions.

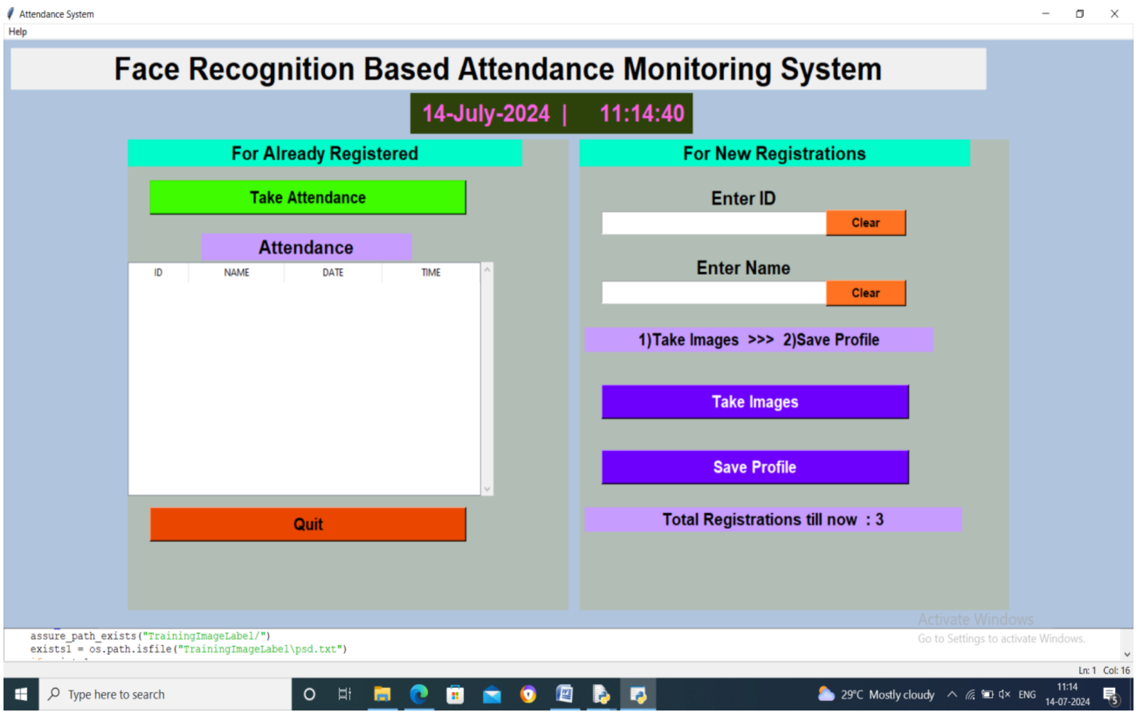
   Example test cases:

   - Introduce noise (e.g., background clutter) and verify if the system still recognizes faces.
   - Test the system’s resilience to variations in lighting and facial appearance.

CHAPTER 9

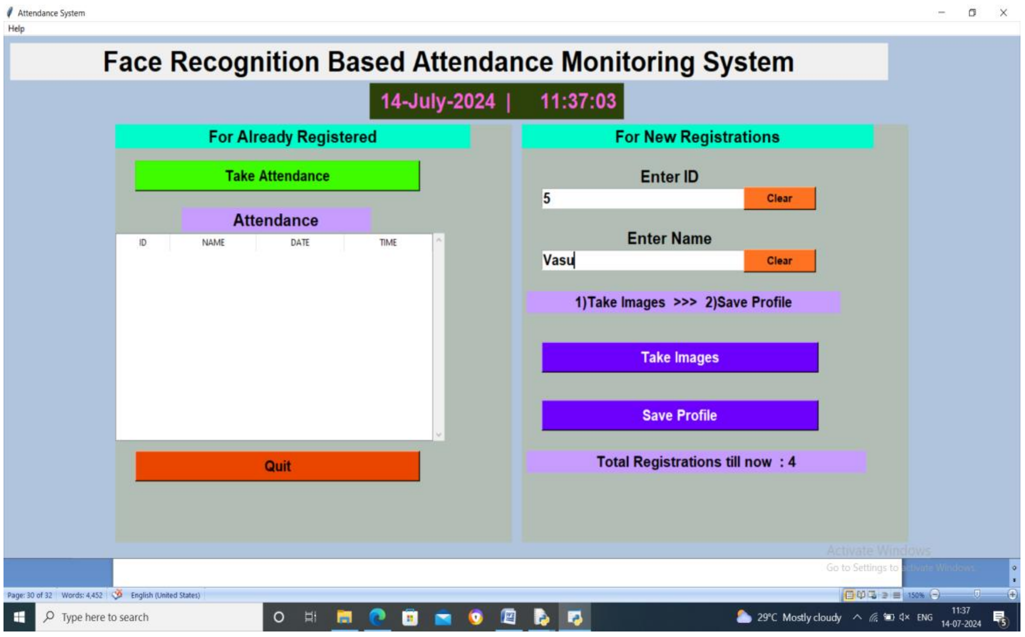
OUTPUT SCREENS

1. Main Interface :



```
assure_path_exists("TrainingImageLabel/")
exists1 = os.path.isfile("TrainingImageLabel\psd.txt")
```

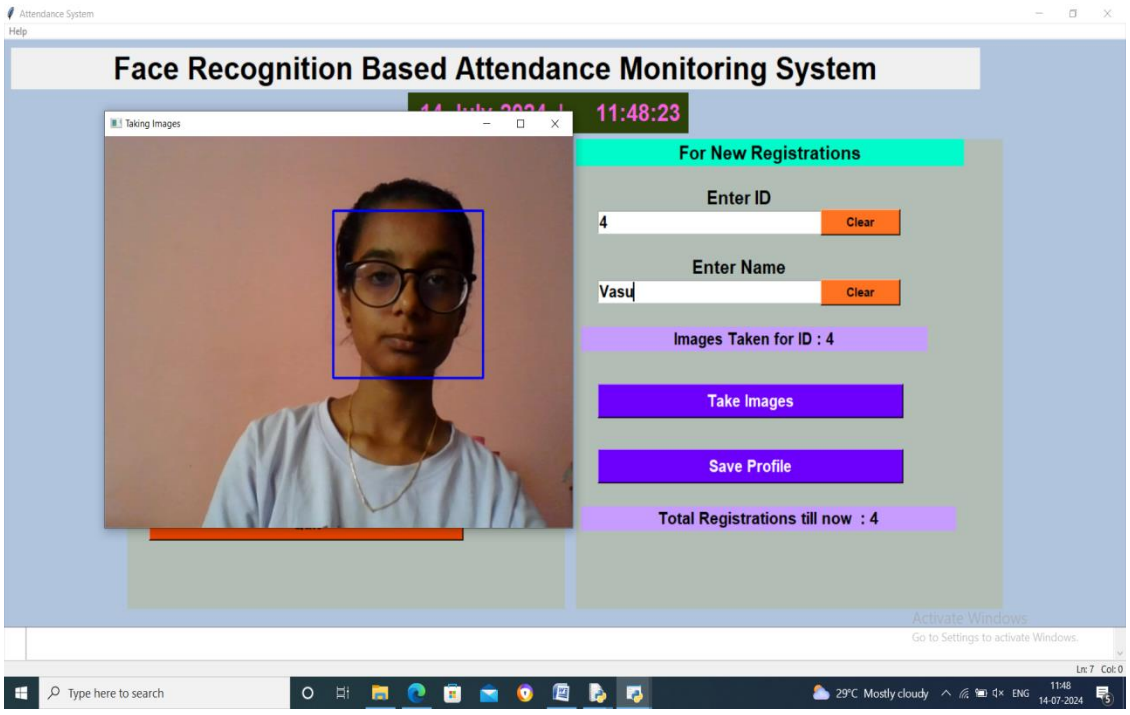
2. For New Registration :



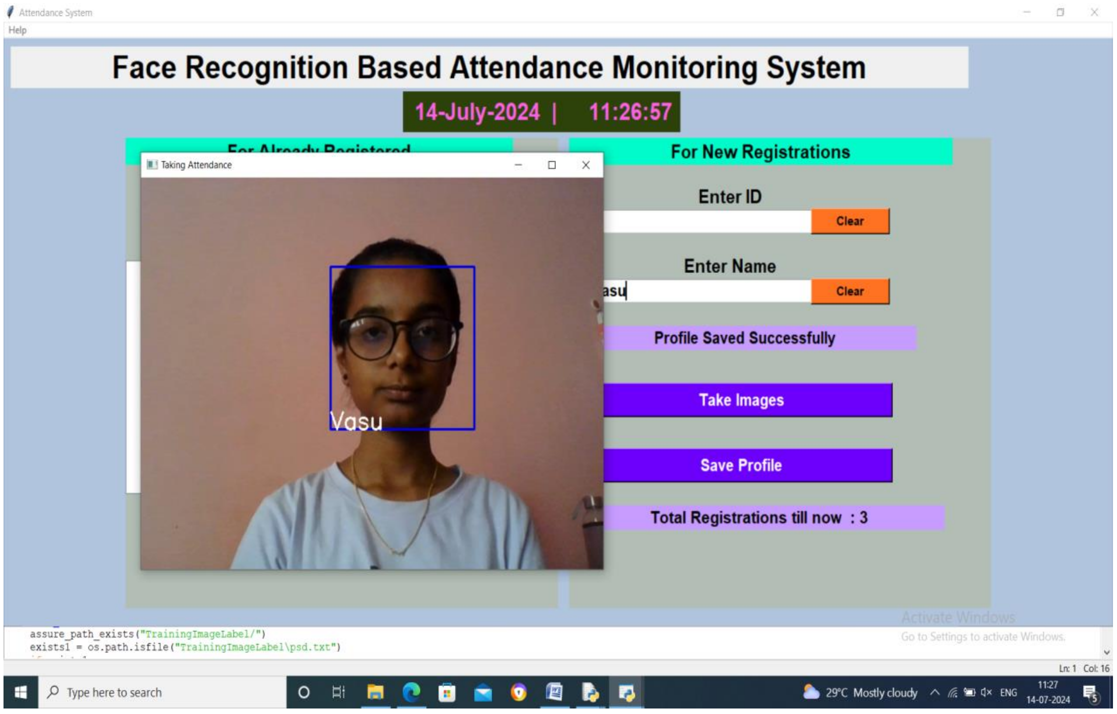
Enter ID: 5

Enter Name: Vasu

3. **Taking images:**

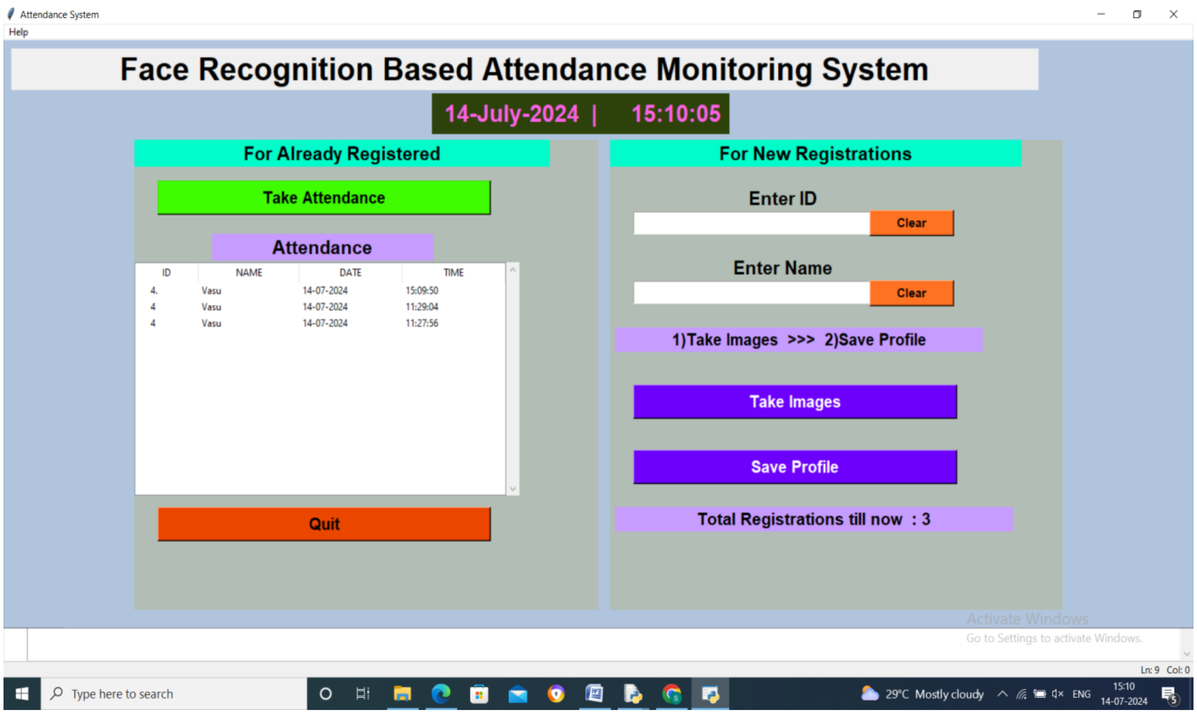


4. **Taking Attendance:**



```
assure_path_exists("TrainingImageLabel/")
exists1 = os.path.isfile("TrainingImageLabel\psd.txt")
```

5. Showing Attendance Taken :

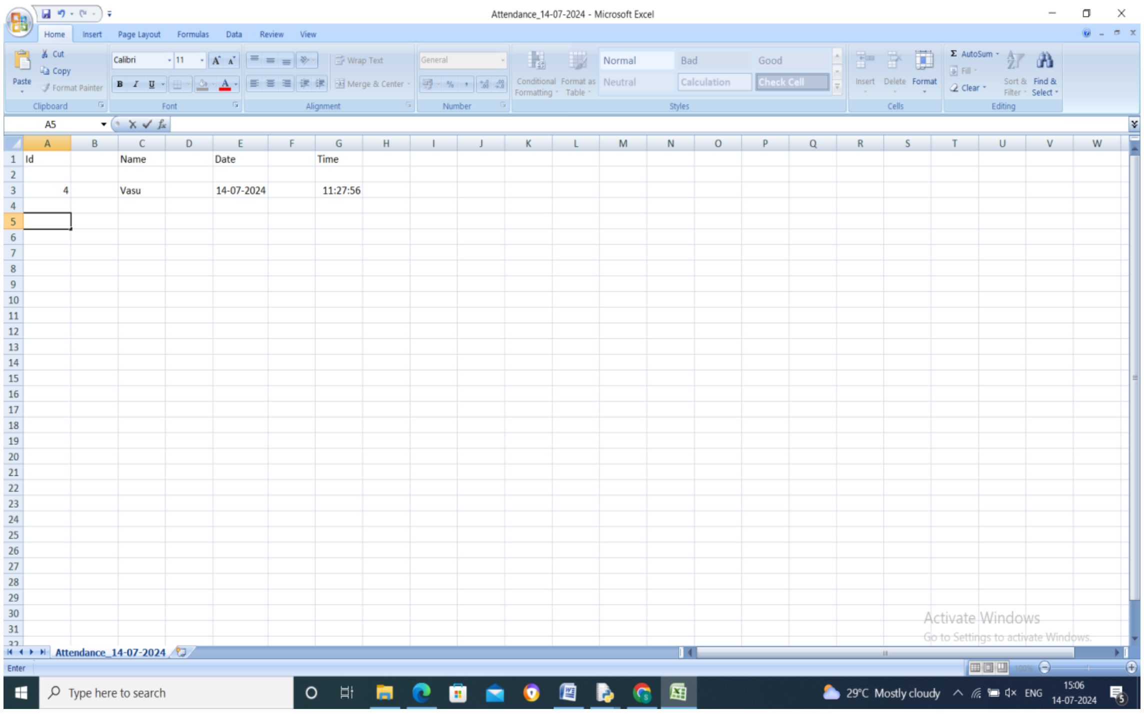


Face Recognition Based Attendance Monitoring System

14-July-2024 | 15:10:05

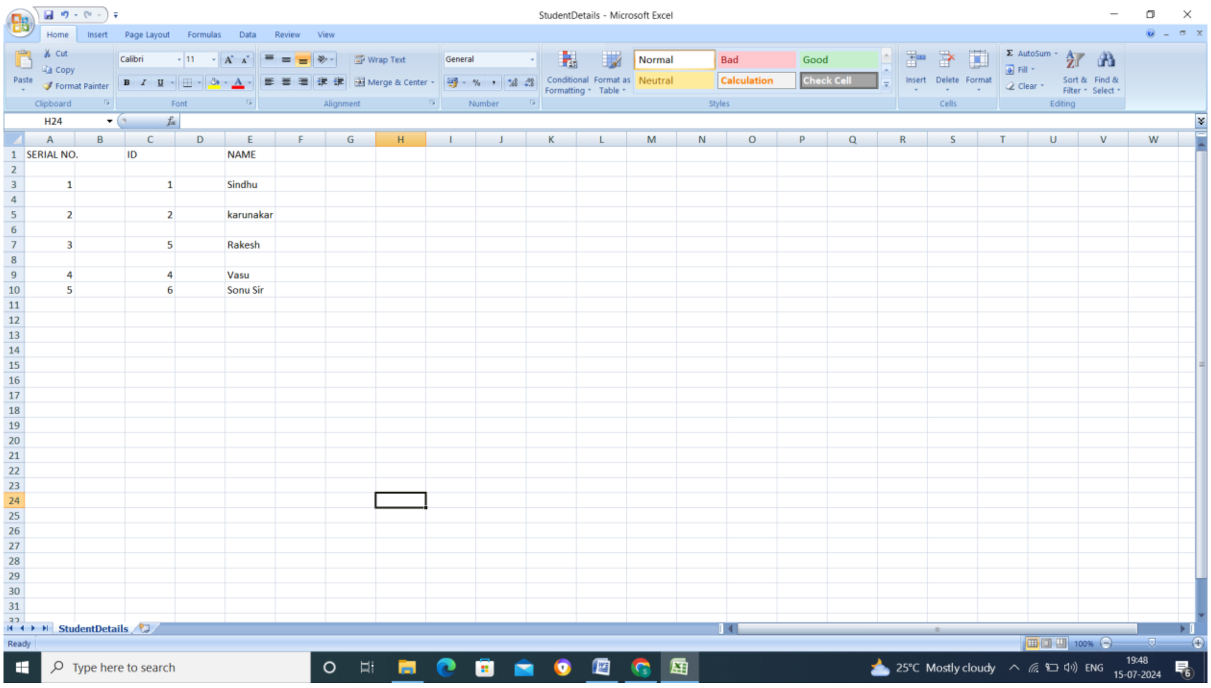
ID	NAME	DATE	TIME
4.	Vasu	14-07-2024	15:09:50
4	Vasu	14-07-2024	11:29:04
4	Vasu	14-07-2024	11:27:56

6. Attendance Stored in ExcelSheet



Id	Name	Date	Time
4	Vasu	14-07-2024	11:27:56

7. Registered People Details:



StudentDetails - Microsoft Excel

SERIAL NO.	ID	NAME
1	1	Sindhu
2	2	karunakar
3	5	Rakesh
4	4	Vasu
5	6	Sonu Sir

CHAPTER 10
CONCLUSION

In conclusion, the implementation of the "**Face Recognition Based Attendance Monitoring System**" real-time project marks a significant advancement in attendance management systems. By harnessing the power of face recognition technology, the project offers a streamlined and accurate method for recording attendance, minimizing errors and enhancing security. The system's capabilities in automating the verification process contribute to operational efficiency, particularly in educational and corporate sectors. This project showcases the potential of innovative solutions to revolutionize traditional practices and pave the way for more efficient and secure attendance management systems.

CHAPTER 11
FUTURE ENHANCEMENTS

As technology evolves, these enhancements promise to make attendance tracking even more efficient and accurate:

1. **Improved Precision:**

   Future systems will focus on enhancing precision. Algorithms will become better at recognizing faces under various conditions, including challenging lighting, partial visibility, and diverse facial expressions. 3D facial recognition methods may be integrated to improve accuracy by capturing depth information.

2. **Faster Processing:**

   Speed is crucial, especially in large classrooms or organizations. Future enhancements will optimize face recognition algorithms for real-time processing. Parallel processing and hardware acceleration (e.g., GPUs) will contribute to faster attendance marking.

3. **Enhanced Security Features:**

   To prevent unauthorized access or misuse, future systems will incorporate robust security features. Anti-spoofing techniques (such as detecting photos or videos) will be integrated to ensure the system only recognizes live faces.

4. **Privacy and Ethical Considerations:**

   As face recognition becomes more widespread, addressing privacy concerns is essential. Future systems will prioritize user consent, data protection, and compliance with privacy regulations.

5. **Integration with Student Management Systems:**

   Seamless integration with existing student management systems will be a priority. This allows attendance data to flow seamlessly into other administrative processes.

6. **Scalability and Adaptability:**

   Future systems will be scalable, accommodating varying class sizes and organizational needs. They will also be adaptable to different educational settings (schools, colleges,universities).

CHAPTER 12
REFERENCES

1. **Reference Books**

    - Python Programming – M. SURESH
    - Tkinter Programmming – GeeksforGeeks

2. **References:**

    - Face Recognition Attendance System:
      By Arijit Das in 2023
      https://github.com/Arijit1080/Face-Recognition-Based-Attendance-System

    - Face Recognition based Attendance Management System :
      By Amit in 2024
      https://github.com/Amit950/Face-recognition-based-Attendance-Management-system

    - Smart Attendance System Using Face Recognition Technology:
      https://ijcrt.org/papers/IJCRT2403757.pdf

    - Attendance Monitoring System Using Face Recognition:
      By Krithika Pathak
      http://www.ir.juit.ac.in:8080/jspui/bitstream/123456789/9825/1/Attendance%20Monitoring%20System%20using%20Face%20Recognition.pdf