Face Recognition based attendance system using Python

COURSE CODE (23UPCSC4P01)

A mini project submitted in partial fulfilment of the requirements for the degree of

MASTER OF SCIENCE

IN

DATA SCIENCE

BY

SASIKUMAR S

(REG NO: U23PG507DTS028)

DEPARTMENT OF COMPUTER SCIENCE
(Supported by UGC-SAP)

PERIYAR UNIVERSITY

(NAAC 'A++' Grade with CGPA 3.61 (Cycle - 3)

State University - NIRF Rank 59 - NIRF Innovation Band of 11-50)

PERIYAR PALKALAI NAGAR, Salem-636011,

APRIL 2024

DR.K. SASIREKHA,

Teaching Assistant*,*

Department of Computer Science,

Periyar University,

Salem-11.

CERTIFICATE

This is to certify that report of Mini Project entitled" **Face Recognition based attendance system using python"**submitted in partial fulfilment of the requirement for the degree of master of science in data science to the periyar university, salem is a record of bonafide work carried out by **SASIKUMAR S (Reg.No:U23PG507DTS028)**under my supervision and guidance.

Signature of the Guide Signature of the HOD

Submitted of the Viva-Voce Examination held on…….

Internal Examiner External Examiner

DECLARATION

I hereby, declare that the project work entitled **"Face Recognition based attendance system using python"** submitted to Periyar University in partial fulfilment of the requirement for the award of the Degree of Master of Science in Data Science is the record work carried out by me, under the supervision **DR.*K.SASIREKHA*, Teaching Assistant of Computer Science, Periyar University**. To the best of my knowledge, the work reported here is not a part of any other work on the basis of which a degree or award was conferred on an earlier to one or any other candidate.

Place: Salem Signature of Student

Date: [SASIKUMAR S]

ACKNOWLEDGEMENT

First, I would like to thank the almighty for providing mw with everything that I requested to completing the project.

I would like to extend my sincere thanks to **Prof Dr. R. JAGANNATHAN,** Vice Chancellor Periyar University, Salem who has been an invaluable source of providing the facility to complete the mini project successfully.

I would like to extend my sincere thanks to **Prof. Dr, C. CHANDRASEKAR**, Head of the Department, Department of Computer Science, Periyar University, Salem For the support and encourage.

I express my sincere thanks to my guide **Dr. K. SASIREKHA, Teaching Assistant**, Department of Computer Science, Periyar University, Salem for the motivation and kind suggestion given in every step throughout this dissertation work and for valuable support in finishing this dissertation.

I extend my thanks to my parents and friends for their constant support and encouragement.

Place: Salem *Signature of the Student*

Date: *[SASIKUMAR S]*

ABSTRACT

In this digital era, face recognition system plays a vital role in almost every sector. Face recognition is one of the mostly used biometrics. It can used for security, authentication, identification, and has got many more advantages. Despite of having low accuracy when compared to iris recognition and fingerprint recognition, it is being widely used due to its contactless and non-invasive process. Furthermore, **face recognition system can also be used for attendance marking in schools, colleges, offices, etc.** This system aims to build a class attendance system which uses the concept of face recognition as existing manual attendance system is time consuming and cumbersome to maintain. And there may be chances of proxy attendance. Thus, the need for this system increases. **This system consists of four phases**- **database creation, face detection, face recognition, attendance updation**. Database is created by the images of the students in class. Face detection and recognition is performed using Haar-Cascade classifier and Local Binary Pattern Histogram algorithm respectively. Faces are detected and recognized from live streaming video of the classroom. Attendance will be mailed to the respective faculty at the end of the session.

2 Keywords:

- Face Recognition;
- Face Detection;
- Haar-Cascade classifier;
- Local Binary Pattern Histogram;
- attendance system;

INDAX

CHAPTER-I

INRODUCTION

Traditional method of attendance marking is a tedious task in many schools and colleges. It is also an extra burden to the faculties who should mark attendance by manually calling the names of students which might take about 5 minutes of entire session. This is time consuming. There are some chances of proxy attendance. Therefore, many institutes started deploying many other techniques for recording attendance like use of **Radio Frequency Identification** (RFID), iris recognition , fingerprint recognition, and so on. However, these systems are queue based which might consume more time and are intrusive in nature.

Face recognition has set an important biometric feature, which can be easily acquirable and is non-intrusive. Face recognition based systems are relatively oblivious to various facial expression. Face recognition system consists of two categories: verification and face identification. Face verification is an 1:1 matching process, it compares face image against the template face images and whereas is an 1:N problems that compares a query face images.

The purpose of this system is to build a attendance system which is based on face recognition techniques. Here face of an individual will be considered for marking attendance. Nowadays, face recognition is gaining more popularity and has been widely used. In this paper, we proposed a system which detects the faces of students from live streaming video of classroom and attendance will be marked if the detecte face is found in the database. This new system will consume less time than compared to traditional methods.

CHAPTER-II

SYSTEM CONFIGURATION

System configuration mainly refers to the specification of a given computer system, from its hardware components to the software and various processes that are run within that system. It refers to what types and models of devices are installed and what specific software is being used to run the various parts of the computer system.

- **Windows:** 10 or newer
- **MAC:** OS X v10.7 or higher
- **Linux:** parrot OS

1 SOFTWARE SPECIFICATION

Operating System	**:** Windows 11.
Coding Language	**:** Python Language.
Software Tool	**:**Python ,Pycharm community edition
System type	: 64-bit operating system, x64-based processor

2 HARDWARE CONFIGURATION

Processor	:11th Gen Intel(R) Core (TM) i3-1115G4 @ 3.00GHz 2.90 GHz
Hard Disk	:512GB
Ram	: 8.00 GB (7.65 GB usable)

CHAPTER-lll

LITERATURE SURVEY

Authors in proposed a model of an automated attendance system. The model focuses on how face recognition incorporated with Radio Frequency Identification (RFID) detect the authorized students and counts as they get in and get out form the classroom. The system keeps the authentic record of every registered student. **The system also keeps the data of every student registered for a particular course in the attendance log and provides necessary information according to the need.**

In this paper , authors have designed and implemented an attendance system which uses iris biometrics. Initially, the attendees were asked to register their details along with their unique iris template. At the time of attendance, the system automatically took class **attendance by capturing the eye image of each attendee, recognizing their iris, and searching for a match in the created database.** The prototype was web based.

In , authors proposed an attendance system based on facial recognition. The algorithms like Viola-Jones and Histogram of Oriented Gradients (HOG) features along with Support Vector Machine (SVM) classifier were used to implement the system. Various real time scenarios such as scaling, illumination, occlusions and pose was considered by the authors. Quantitative analysis was done on the basis of Peak Signal to Noise Ratio (PSNR) values and was implemented in MATLAB GUI.

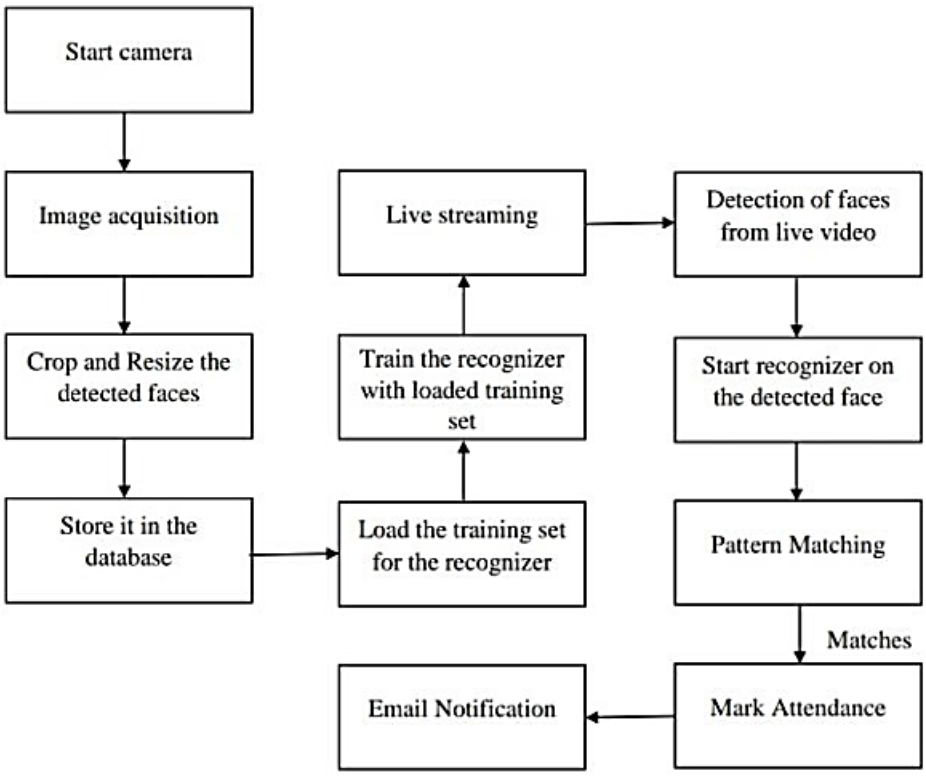
Authors in researches to get best facial recognition algorithm (Eigenface and Fisher face) provided by the Open CV 2.4.8 by comparing the Receiver Operating Characteristics (ROC) curve and then implemented it in the attendance system. Based on the experiments carried out in this paper, the ROC curve proved that, Eigenface achieves better result than Fisher face. System implemented using Eigenface algorithm achieved an accuracy rate of 70% to 90%. In, authors proposed a method for student attendance system in classroom using face recognition technique by combining Discrete Wavelet Transforms (DWT) and Discrete Cosine Transform (DCT). These algorithms were used to extract the features of student's face followed by applying Radial Basis Function (RBF) for classifying the facial objects. This system achieved an accuracy rate of 82%.

CHAPTER-IV

PROPOSED SYSTEM

All the students of the class must register themselves by entering the required details and then their images will be captured and stored in the dataset. During each session, faces will be detected from live streaming video of classroom. The faces detected will be compared with images present in the dataset. If match found, attendance will be marked for the respective student. At the end of each session, list of absentees will be mailed to the respective faculty handling the session. The system architecture of the proposed system is given below,

System Architecture



process can be divided into four stages,

1.Dataset

2.Face Detection

3.Face Recognition

4.Attendance Updation

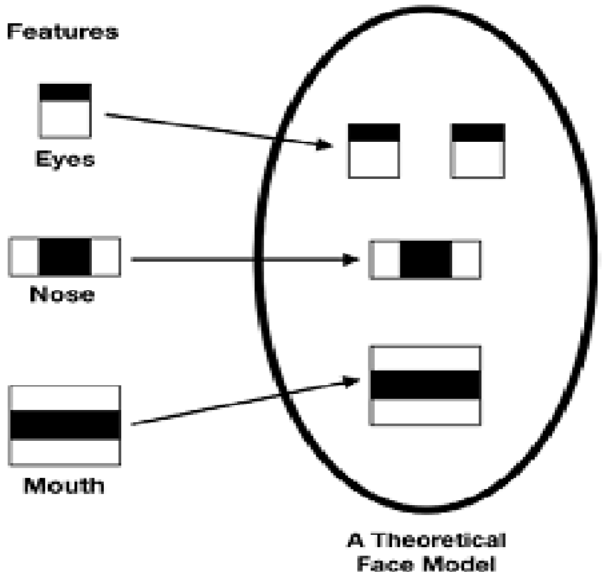
1.Dataset Creation:

Images of students are captured using a web cam. Multiple images of single student will be acquired with varied gestures and angles. These images undergo pre-processing. The images are cropped to obtain the Region of Interest (ROI) which will be further used in recognition process. Next step is to resize the cropped images to particular pixel position. Then these images will be converted from RGB to gray scale images. And then these images will be saved as the names of respective student in a folder.

2. Face Detection:

Face detection here is performed using Haar-Cascade Classifier with OpenCV. Haar Cascade algorithm needs to be trained to detect human faces before it can be used for face detection. This is called feature extraction. The haar cascade training data used is an xml file- haar cascade-frontal face-default. The haar features shown in . will be used for feature extraction.

Haar Features



Here we are using detect Multi Scale module from OpenCV. This is required to create a rectangle around the faces in an image. It has got three parameters to consider- scale Factor, min Neighbors, min Size. Scale Factor is used to indicate how much an image must be reduced in each image scale. Min Neigh bors specifies how many neigh bors each candidate rectangle must have. Higher values usually detects less faces but detects high quality in image. Min Size specifies the minimum object size. By default it is (30,30) . The parameters used in this system is scale Factor and min Neigh bors with the values 1.3 and 5 respectively.

3. Face Recognition:

Face recognition process can be divided into three steps- prepare training data, train face recognizer, prediction. Here training data will be the images present in the dataset. They will be assigned with a integer label of the student it belongs to. These images are then used for face recognition. Face recognizer used in this system is **Local Binary Pattern Histogram**. Initially, the list of local binary patterns (LBP) of entire face is obtained. These LBPs are converted into decimal number and then histograms of all those decimal values are made. At the end, one histogram will be formed for each images in the training data. Later, during recognition process histogram of the face to be recognized is calculated and then compared with the already computed histograms and returns the best matched label associated with the student it belongs to.

4. Attendance Updation:

After face recognition process, the recognized faces will be marked as present in the excel sheet and the rest will be marked as absent and the list of absentees will be mailed to the respective faculties. Faculties will be updated with monthly attendance sheet at the end of every month.

CHAPTER-V

RESULTS AND DISCUSSIONS

The users can interact with the system using a GUI. Here users will be mainly provided with three different options such as, student registration, faculty registration, and mark attendance. The students are supposed to enter all the required details in the student registration form. After clicking on register button, the web cam starts automatically and window as shown in pops up and starts detecting the faces in the frame. Then it automatically starts clicking photos until 60 samples are collected or CRTL+Q is pressed. These images then will be pre-processed and stored in training images folder.

The faculties are supposed to register with the respective course codes along with their email-id in the faculty registration form provided. This is important because the list of absentees will be ultimately mailed to the respective faculties.

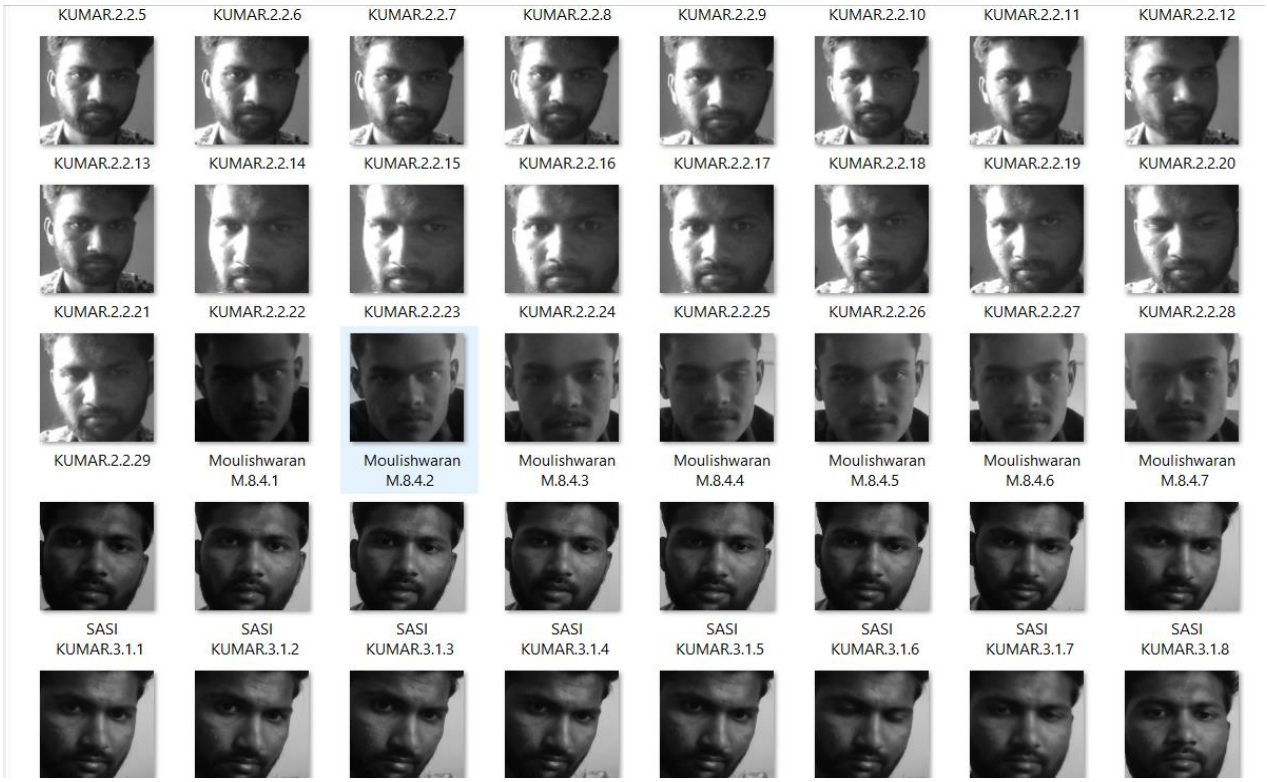
Face Detection

In every session, respective faculty must enter their course code. Then after submitting the course code, the camera will start automatically. The shows the face recognition window where two registered students are recognized and if in case they were not registered it would have shown ‘unknown’. By pressing CTRL+Q, the window will be closed and attendance will be updated in the excel sheet and names of absentees will be mailed to the respective faculty.

Face Recognition

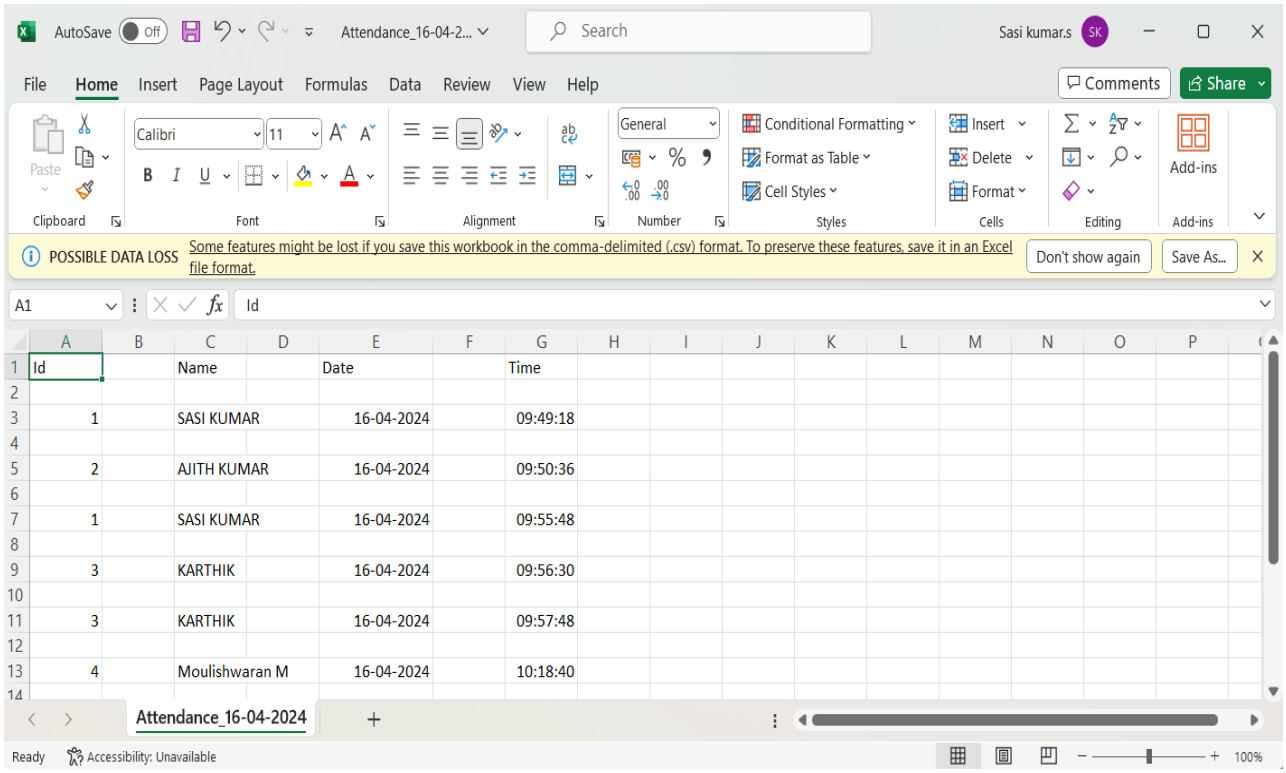
Data entered in the user interface are **stored in the student database** as show in above screen.

Track Images Page



On clicking the face detector button, a webcam is started for taking the attendance of the students.

Attendance sheet



Id	Name	Date	Time
1	SASI KUMAR	16-04-2024	09:49:18
2	AJITH KUMAR	16-04-2024	09:50:36
1	SASI KUMAR	16-04-2024	09:55:48
3	KARTHIK	16-04-2024	09:56:30
3	KARTHIK	16-04-2024	09:57:48
4	Moulishwaran M	16-04-2024	10:18:40

hows the attendance sheet updated after recognition process. Recognized students are marked as '1' and absent students are marked as '0'. The list of absentees will be mailed to the respective faculty email-id.

TECHNOLOGY USED

1. tkinter for whole GUI
2. OpenCV for taking images and face recognition (cv2.face.LBPHFaceRecognizer_create())
3. CSV, Numpy, Pandas, datetime etc. for other purposes.

FEATURES

1. Easy to use with interactive GUI support.
2 .Password protection for new person registration.
3. Creates/Updates CSV file for deatils of students on registration.
4. Creates a new CSV file everyday for attendance and marks attendance with proper date and time.
5.Displays live attendance updates for the day on the main screen in tabular format with Id, name, date and time.

Data Base Dictionary

Field Name	Data Type	Description
Rool_no	int	Student roll no
Name	Varchar	Name of Student
Date	Date	Date of the Attendance
Time	time	Time of the Attendance
Attendance	Varchar	Attendance of a student
Images	.pgm	Images of Students

Install All the modules mentioned below :

1.pip install tk-tools

2.pip install opencv-contrib-python

3.pip install datetime

4.pip install pytest-shutil

4.pip install python-csv

5.pip install numpy

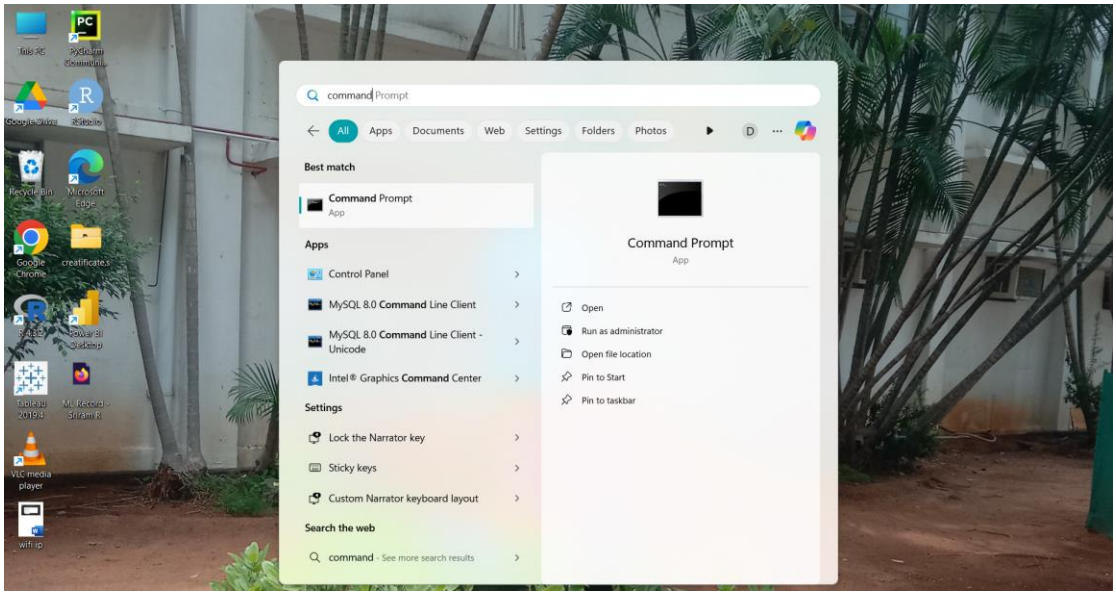
6.pip install pillow

7.pip install pandas

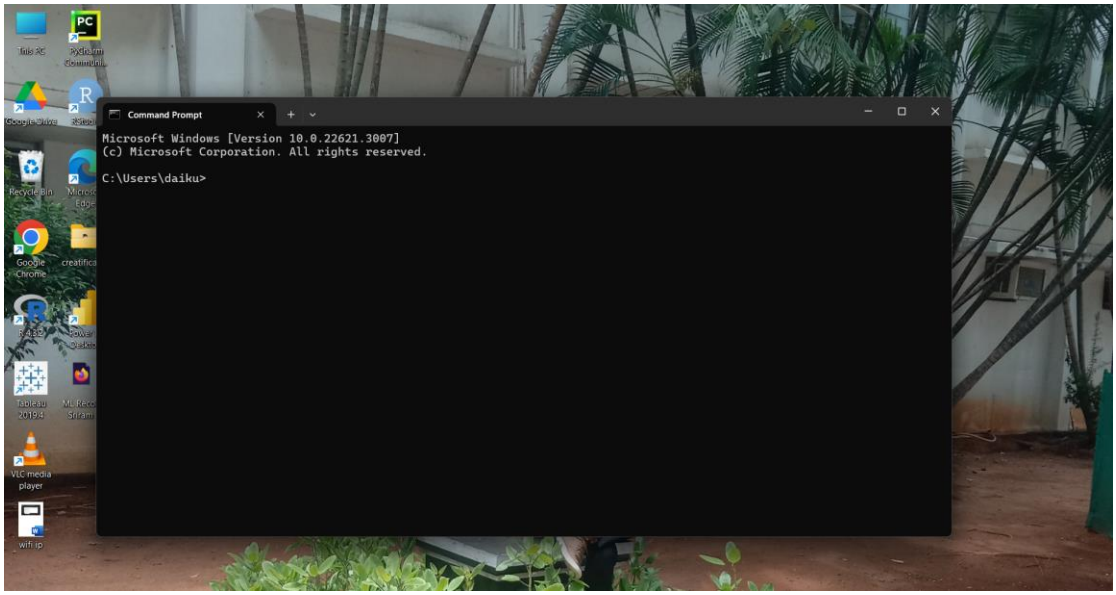
8.pip install times

INSTALL PROCESS

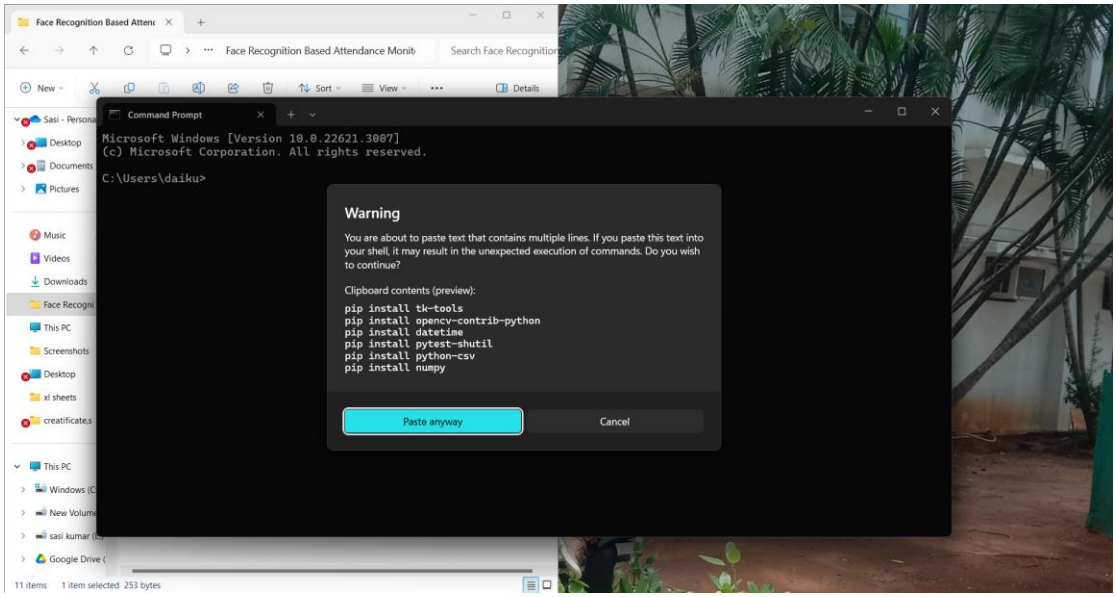
Step_1



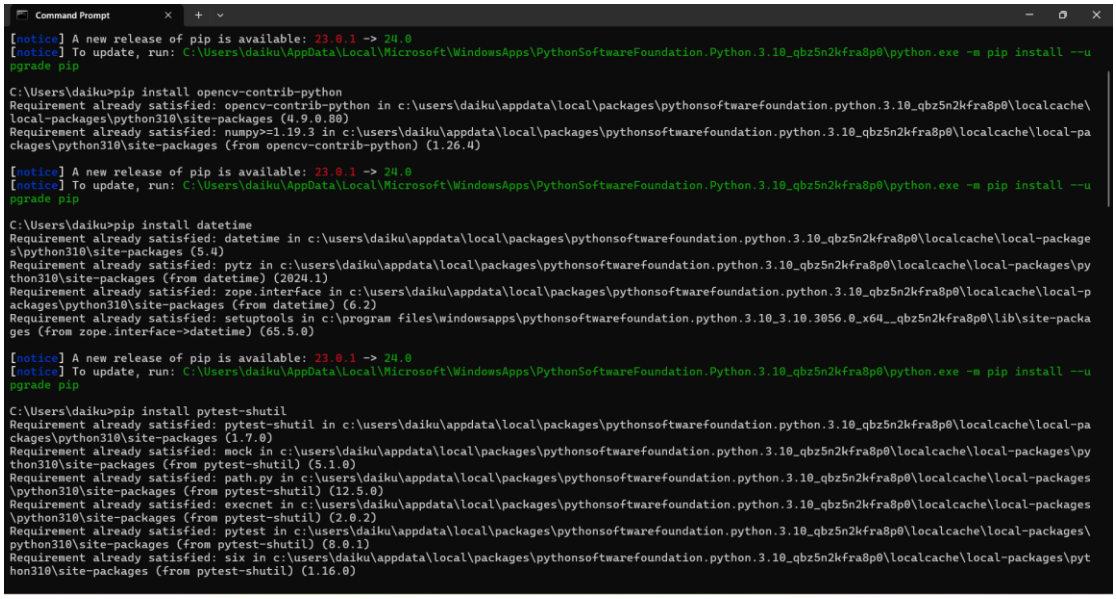
Step_2



Step_3


Face Recognition Based Attenc
Face Recognition Based Attendance Monit
Search Face Recognition
Command Prompt
Microsoft Windows [Version 10.0.22621.3007]
(c) Microsoft Corporation. All rights reserved.
C:\Users\daiku>
Warning
You are about to paste text that contains multiple lines. If you paste this text into your shell, it may result in the unexpected execution of commands. Do you wish to continue?
Clipboard contents (preview):
pip install tk-tools
pip install opencv-contrib-python
pip install datetime
pip install pytest-shutil
pip install python-csv
pip install numpy
Paste anyway
Cancel
11 items
1 item selected 253 bytes

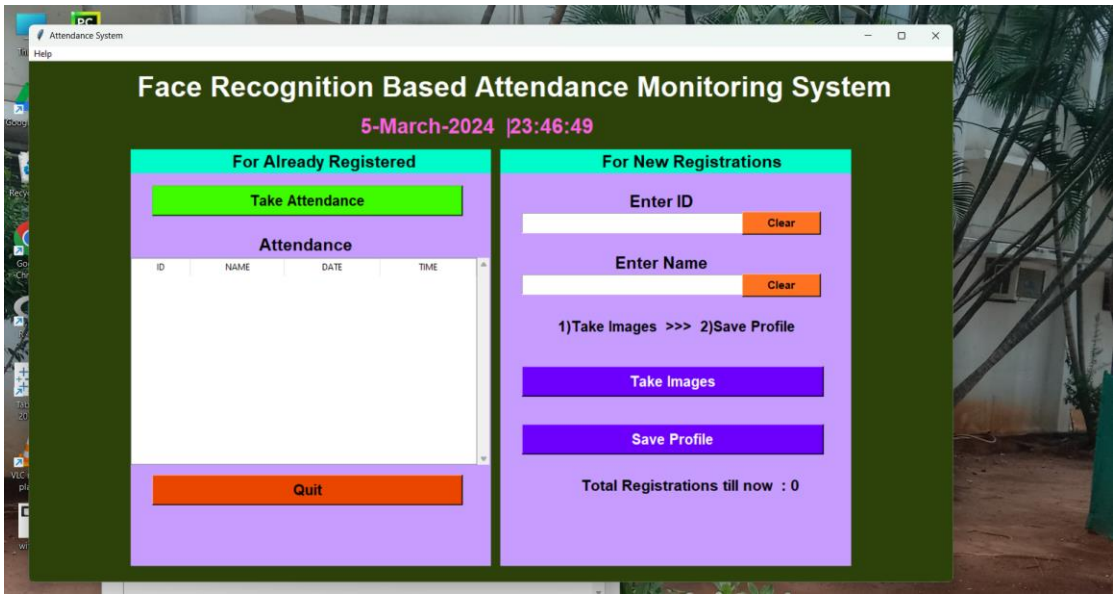
Step_4


Command Prompt
[notice] A new release of pip is available: 23.0.1 -> 24.0
[notice] To update, run: C:\Users\daiku\AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundation.Python.3.10_qbz5n2kfra8p0\python.exe -m pip install --upgrade pip
C:\Users\daiku>pip install opencv-contrib-python
Requirement already satisfied: opencv-contrib-python in c:\users\daiku\appdata\local\packages\pythonsoftwarefoundation.python.3.10_qbz5n2kfra8p0\localcache\local-packages\python310\site-packages (4.9.0.80)
Requirement already satisfied: numpy>=1.19.3 in c:\users\daiku\appdata\local\packages\pythonsoftwarefoundation.python.3.10_qbz5n2kfra8p0\localcache\local-packages\python310\site-packages (from opencv-contrib-python) (1.26.4)
[notice] A new release of pip is available: 23.0.1 -> 24.0
[notice] To update, run: C:\Users\daiku\AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundation.Python.3.10_qbz5n2kfra8p0\python.exe -m pip install --upgrade pip
C:\Users\daiku>pip install datetime
Requirement already satisfied: datetime in c:\users\daiku\appdata\local\packages\pythonsoftwarefoundation.python.3.10_qbz5n2kfra8p0\localcache\local-packages\python310\site-packages (5.4)
Requirement already satisfied: pytz in c:\users\daiku\appdata\local\packages\pythonsoftwarefoundation.python.3.10_qbz5n2kfra8p0\localcache\local-packages\python310\site-packages (from datetime) (2024.1)
Requirement already satisfied: zope.interface in c:\users\daiku\appdata\local\packages\pythonsoftwarefoundation.python.3.10_qbz5n2kfra8p0\localcache\local-packages\python310\site-packages (from datetime) (6.2)
Requirement already satisfied: setuptools in c:\program files\windowsapps\pythonsoftwarefoundation.python.3.10_3.10.3056.0_x64__qbz5n2kfra8p0\lib\site-packages (from zope.interface->datetime) (65.5.0)
[notice] A new release of pip is available: 23.0.1 -> 24.0
[notice] To update, run: C:\Users\daiku\AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundation.Python.3.10_qbz5n2kfra8p0\python.exe -m pip install --upgrade pip
C:\Users\daiku>pip install pytest-shutil
Requirement already satisfied: pytest-shutil in c:\users\daiku\appdata\local\packages\pythonsoftwarefoundation.python.3.10_qbz5n2kfra8p0\localcache\local-packages\python310\site-packages (1.7.0)
Requirement already satisfied: mock in c:\users\daiku\appdata\local\packages\pythonsoftwarefoundation.python.3.10_qbz5n2kfra8p0\localcache\local-packages\python310\site-packages (from pytest-shutil) (5.1.0)
Requirement already satisfied: path.py in c:\users\daiku\appdata\local\packages\pythonsoftwarefoundation.python.3.10_qbz5n2kfra8p0\localcache\local-packages\python310\site-packages (from pytest-shutil) (12.5.0)
Requirement already satisfied: execnet in c:\users\daiku\appdata\local\packages\pythonsoftwarefoundation.python.3.10_qbz5n2kfra8p0\localcache\local-packages\python310\site-packages (from pytest-shutil) (2.0.2)
Requirement already satisfied: pytest in c:\users\daiku\appdata\local\packages\pythonsoftwarefoundation.python.3.10_qbz5n2kfra8p0\localcache\local-packages\python310\site-packages (from pytest-shutil) (8.0.1)
Requirement already satisfied: six in c:\users\daiku\appdata\local\packages\pythonsoftwarefoundation.python.3.10_qbz5n2kfra8p0\localcache\local-packages\python310\site-packages (from pytest-shutil) (1.16.0)

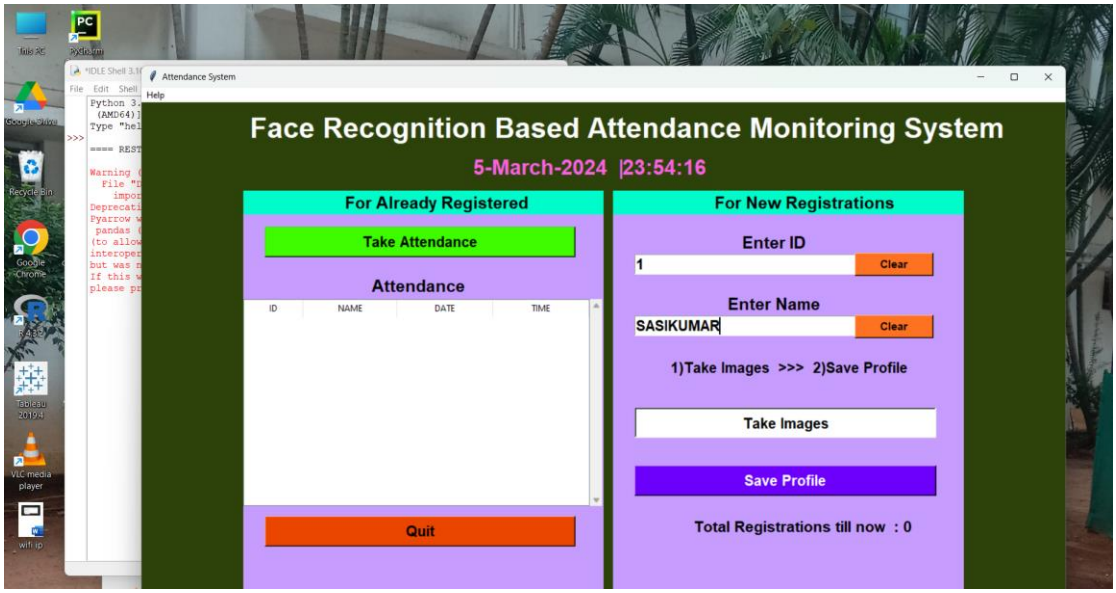
FACE ATTENCE SYSTE:

SCREENSHORTS

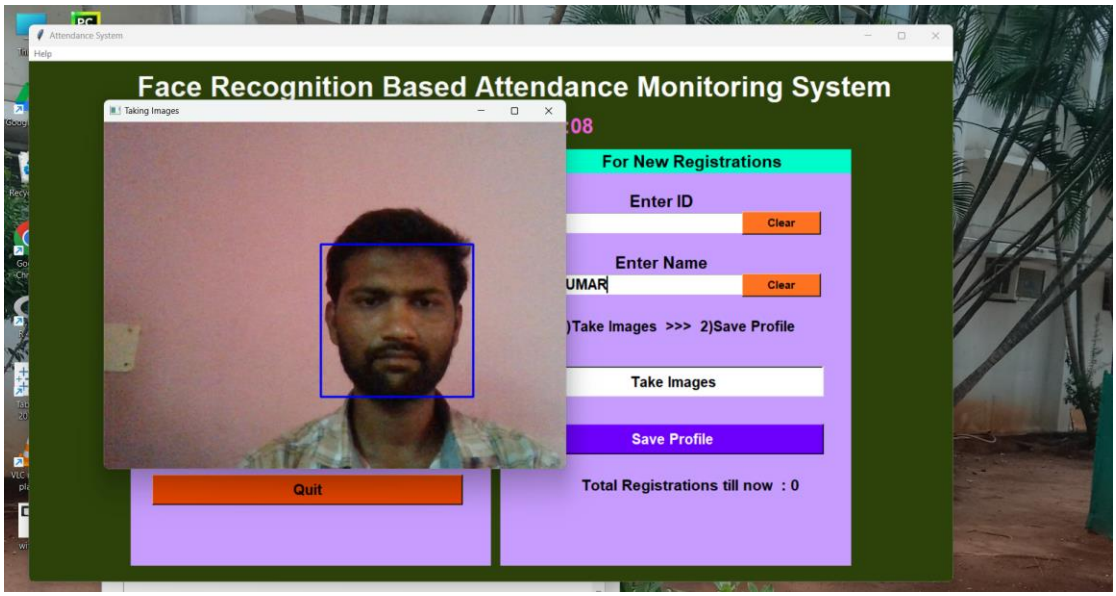
Step_1



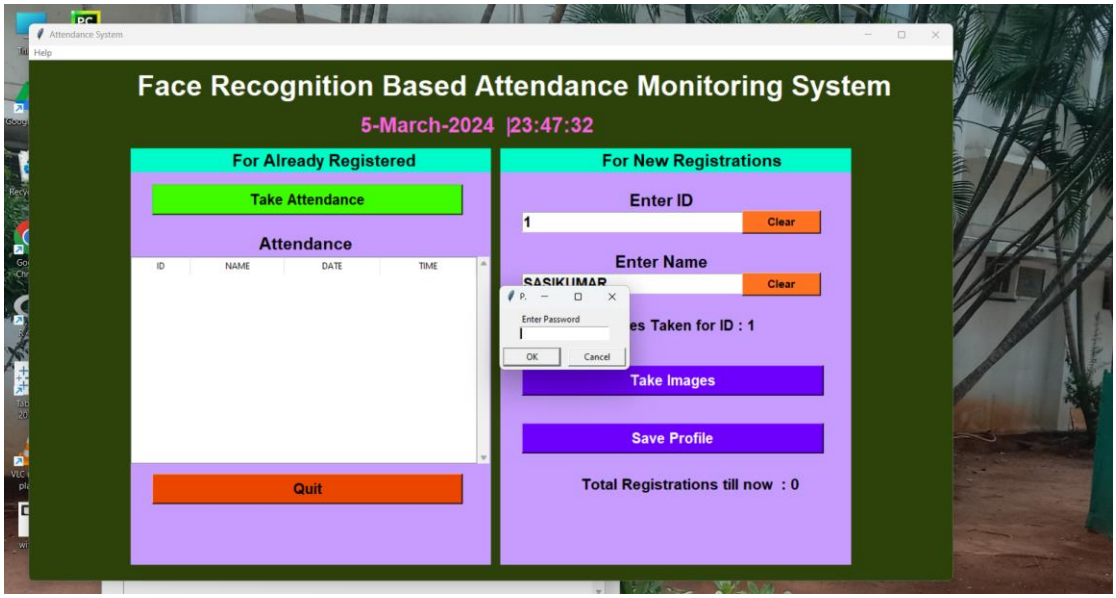
Step_2



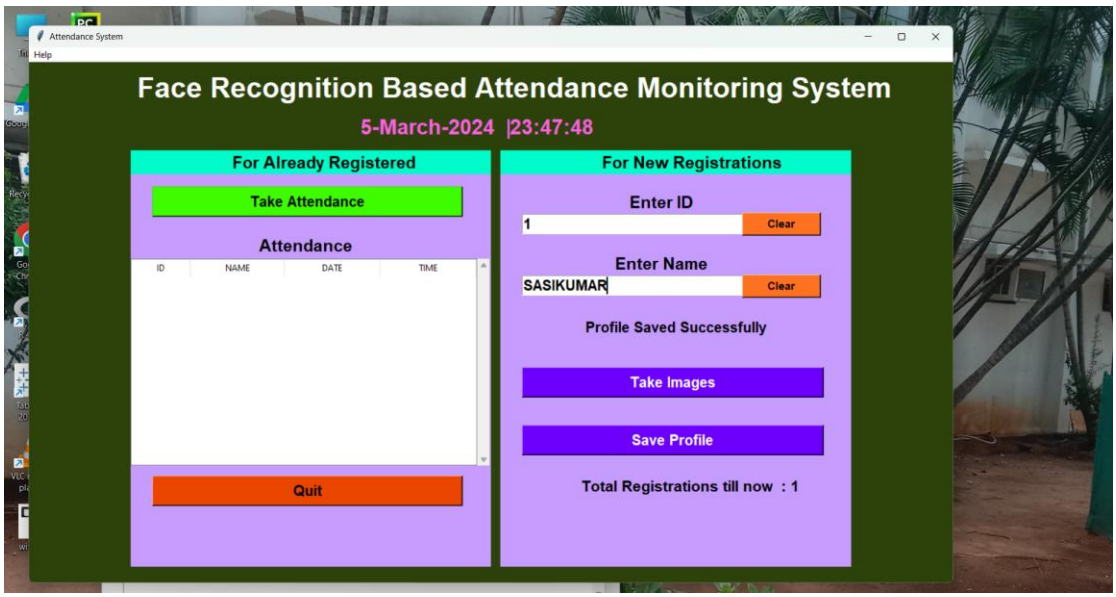
`step_3


Face Recognition Based Attendance Monitoring System
Taking Images
For New Registrations
Enter ID
Clear
Enter Name
Clear
)Take Images >>> 2)Save Profile
Take Images
Save Profile
Quit
Total Registrations till now : 0

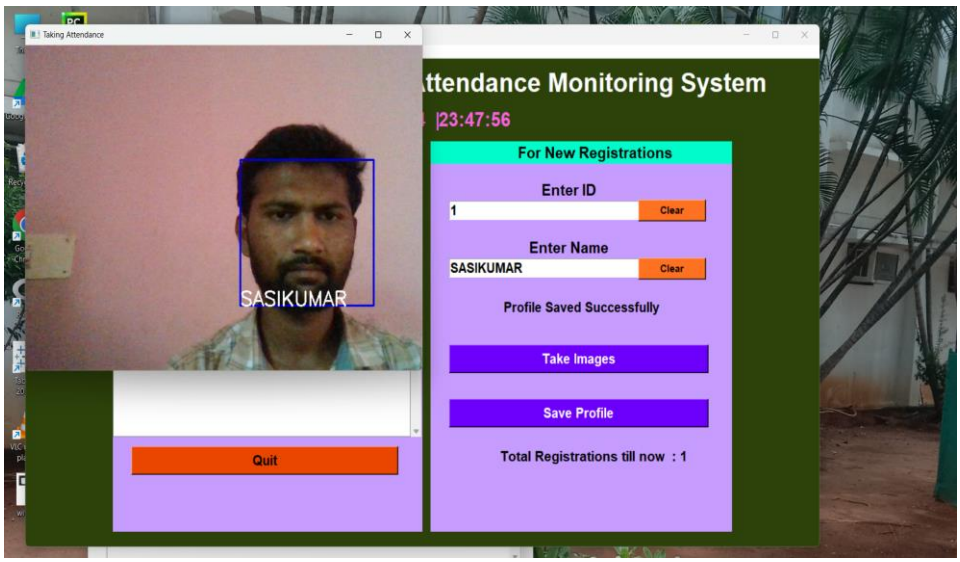
Step_4


Face Recognition Based Attendance Monitoring System
5-March-2024 |23:47:32
For Already Registered
Take Attendance
Attendance
ID
NAME
DATE
TIME
Quit
For New Registrations
Enter ID
1
Clear
Enter Name
Clear
Enter Password
OK
Cancel
Take Images
Save Profile
Total Registrations till now : 0

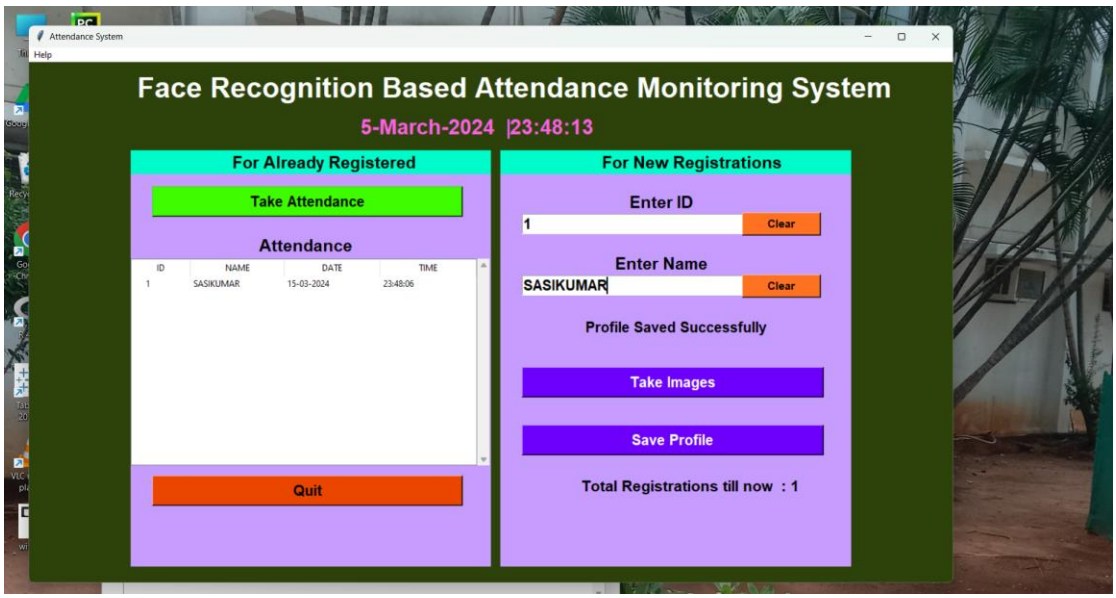
Step_5


Attendance System
Help
Face Recognition Based Attendance Monitoring System
5-March-2024 |23:47:48
For Already Registered
Take Attendance
Attendance
ID
NAME
DATE
TIME
Quit
For New Registrations
Enter ID
1
Clear
Enter Name
SASIKUMAR
Clear
Profile Saved Successfully
Take Images
Save Profile
Total Registrations till now : 1

Step_6


Taking Attendance
SASIKUMAR
Quit
ttendance Monitoring System
|23:47:56
For New Registrations
Enter ID
1
Clear
Enter Name
SASIKUMAR
Clear
Profile Saved Successfully
Take Images
Save Profile
Total Registrations till now : 1

Step_7



Source Code screen short:

```
# IMPORTING
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 : 'shubhamkumar8180323@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 (' ' 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='#2d420a')
```

```
frame1 = tk.Frame(window, bg="#c79cff")
frame1.place(relx=0.11, rely=0.17, relwidth=0.39, relheight=0.80)

frame2 = tk.Frame(window, bg="#c79cff")
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="white",bg="#2d420a" ,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', 22, ' 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="#c79cff" ,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="#c79cff" ,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', 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=('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)

# 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()
```

CONCLUSION

This system aims to build an effective class attendance system using face recognition techniques. The proposed system will be able to mark the attendance via face Id. It will detect faces via webcam and then recognize the faces. After recognition, it will mark the attendance of the recognized student and update the attendance record.

APPENDIX

SOURCE CODE

 # IMPORT libirary

```
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 : 'shubhamkumar8180323@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 (' ' 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("-")
month={
'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='#2d420a')


frame1 = tk.Frame(window, bg="#c79cff")

frame1.place(relx=0.11, rely=0.17, relwidth=0.39, relheight=0.80)


frame2 = tk.Frame(window, bg="#c79cff")

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="white",bg="#2d420a" ,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', 22, ' 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="#c79cff" ,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="#c79cff" ,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', 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=('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()
```

REFERENCES

[1] Hapani, Smit, et al. "Automated Attendance System Using Image Processing." 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA). IEEE, 2018.

[2] Akbar, Md Sajid, et al. "Face Recognition and RFID Verified Attendance System." 2018 International Conference on Computing, Electronics & Communications Engineering (iCCECE). IEEE, 2018.

[3] Okokpujie, Kennedy O., et al. "Design and implementation of a student attendance system using iris biometric recognition." 2017 International Conference on Computational Science and Computational Intelligence (CSCI). IEEE, 2017.

[4] Rathod, Hemantkumar, et al. "Automated attendance system using machine learning approach." 2017 International Conference on Nascent Technologies in Engineering (ICNTE). IEEE, 2017.

[5] Siswanto, Adrian Rhesa Septian, Anto Satriyo Nugroho, and Maulahikmah Galinium. "Implementation of face recognition algorithm for biometrics based time attendance system." 2014 International Conference on ICT For Smart Society (ICISS). IEEE, 2014.

[6] Lukas, Samuel, et al. "Student attendance system in classroom using face recognition technique." 2016 International Conference on Information and Communication Technology Convergence (ICTC). IEEE, 2016.

[7] https://becominghuman.ai/face-detection-using-opencv-with-haar-cascade-classifiers-941dbb25177

[8] https://www.superdatascience.com/blogs/opencv-face-recognition

[9] Salim, Omar Abdul Rhman, Rashidah Funke Olanrewaju, and Wasiu Adebayo Balogun. "Class attendance management system using face recognition." 2018 7th International Conference on Computer and Communication Engineering (ICCCE). IEEE, 2018.