# ATTENDANCE SYSTEM USING FACE RECOGNITION

**PROJECT REPORT**

**DIPLOMA**

IN

**COMPUTER AND ENGINEERING**

Submitted by

**S. Shaiksha Vali (20022-CM-049)**

Under the supervision of

**Mr. M. Vijaya Kumar, M.Tech**

**Head of the Department & Project Guide**

**Dept. of Computer Engineering,**

**Government Polytechnic Proddatur**

GOVERNMENT POLYTECHNIC PRODDATUR
DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

This is to certify that the project report entitled “Attendance System using Face Recognition” is a bonafide record work of **S.Shaiksha Vali (20022-CM-049)** in particular fulfilment of the requirement of award of Diploma in Computer Engineering by Department of Technical Education during the academic year 2022-2023 in Government Polytechnic Proddatur, Kadapa.

| | |
|---|---|
| **HEAD OF THE DEPARTMENT** | **PROJECT GUIDE** |
| M. VIJAYA KUMAR, M.Tech | M. VIJAYA KUMAR, M.Tech |
| Department of Computer Engineering | Department of Computer Engineering |

| | |
|---|---|
| **External Examiners** | **Internal Examiners** |
| **1.** | **1.** |
| **2.** | **2.** |

# ACKNOWLEDGEMENT

All endeavors over a long period can be successful only with device and the support of many well wishers. I take the opportunity to express my gratitude and appreciation to all of them.

While doing the project many difficulties came in between but finally it was with the guidance and suggestion from the following intelligence who gave generously of their time and expertise.

I express profound gratitude to my guide M.VIJAYA KUMAR M.Tech Head of the Department,department of Computer Engineering for his valuable guidance and perspicacious analytical and suggestions throughout this project.And also I am grateful to him to providing suggestions and constant encouragement that led to a roaring success.

I am highly indebted to the P. RAJESH, Lab Incharge department of computer engineering,who provided the lab to the work on the computer and completion of Project.

I would like to thank our honorable principal M.V.Ch. JAGADEESWARUDU,

who has inspired us a lot through his speeches.He has given the meaning to technical studies and told us how to survive in this competitive world.

I am eternally grateful to my parents for their affectionate co- operation and blessings for this great achievement.

This acknowledgement transcends the reality of formality when I would like to express deep gratitude and respect to all those who guided, inspired and helped me for the completion of this project work.

## ABSTRACT

Face identification has been considered an interesting research domain in the past few years as it plays a major biometric authentication role in several applications including attendance management and access control systems. Attendance management systems are very important to all organizations though they are complex and time-consuming for managing regular attendance log. There are many automated human identification techniques such as biometrics, RFID, eye tracking, and voice recognition. Face is one of the most broadly used biometrics for human identity authentication. This paper presents a facial recognition attendance system based on deep learning convolutional neural networks. We utilize transfer learning by using three pre-trained convolutional neural networks and training them on our data. The three networks showed very high performance in terms of high prediction accuracy and reasonable training time.

FEATURES

1. Face detection in Real-time.
2. This can be used for attendance of a person.
3. This can be used for identification of Live person.

CONTENTS

# 1.INTRODUCTION

Traditional method of attendance marking is a tedious task in many colleges and schools. And an extra burden for faculties and it consumes the time. So many institutes has started deploying many other techniques like Fingerprint Recognition, iris recognition, etc. Face Recognition has set an important biometric feature which is easily acquirable and non-intrusive. Attendance using Face Recognition is an important area of computer vision research and applications. The goal of the Face recognition is to uniquely identify individuals during login, attendance events. Its applications are to identify individuals using their face. Most everyday human tasks of taking attendance of students in colleges, employees in industries, officers in government can be simplified if they can be recognized through the face recognizing system. Generally, the human activity recognition system may or may not be supervised. The recognition of Face is mainly used in security and law enforcement. But we used to register their face and take attendance of a person in this project.

## 1.1 PROBLEM STATEMENT

People had to take attendance in the attendance register book, it was hectic, time consuming and prone to human errors and negligence. So that Attendance using Face Recognition was developed.

## 1.2 MOTIVATION

We seek to provide a valuable attendance service for both teachers and students. Reduce manual process errors by providing automated and reliable attendance using face recognition technology.

## 1.3 AIMS AND OBJECTIVES

- The main objective of our application is to detect faces.
- To Reduce Human errors in taking attendance.

# 2. REQUIREMENT ANALYSIS:

The project involved recognising various faces which can be used in surveillance systems and attendance systems.

## 2.1 REQUIREMENT SPECIFICATION:

A Requirement Specification is a collection of the set of all requirements that are to be imposed on the design and verification of the product.

### 2.1.1 USER REQUIREMENTS:

User Requirements, often referred to as user needs, describe what the user does with the system

- The User need to register with name and ID then train their face to detect the face while taking attendance.

### 2.1.2 FUNCTIONAL REQUIREMENTS:

- System Interface with the user.
- User should be able to register.

### 2.1.3 SOFTWARE REQUIREMENTS

For developing the website, the following are the software requirements needed:

- Operating System: Windows 10
- Language: Python
- Tools: Anaconda Prompt
- Libraries used: NumPy, tkinter, CV2, OS, Pillow, Pandas, datetime, time

For running the website, the following are the software requirements needed:

- Browser: Google Chrome, fire-fox, internet Explorer 🎬 Network: Wi-Fi Internet or Cellular Network.

## 2.1.4 HARDWARE REQUIREMENTS

For developing the Attendance System using FRS, the following are the Hardware requirements needed:

- Processor speed: 2 GHz and more.
- RAM: 4 GB
- Space or Hard Disk: 128 GB

For running the website, the following are the requirements needed:

- Minimum space to execute : 1 GB for Program and N GB according to the data set.
- Device: Device having Anaconda Prompt.

## 2.1.5 NON-FUNCTIONAL REQUIREMENTS:

It defines the system attributes such as Security, Reliability, Performance, Scalability, Usability.

- Security: Password authentication is provided before saving a profile for new registrations.
- Reliability and Performance: Our Project can be reliable and can have consistent-by-time performance.
- Scalability: We can add as many numbers of registrations according to the need.

# 3. SYSTEM DESIGN

System design is the process of designing the elements of a system such as the architecture, modules and components, the different interfaces of those components and the data that goes through that system. The purpose of the System Design process is to provide sufficient detailed data and information about the system and its system elements to enable the implementation consistent with architectural entities as defined in models and views of the system architecture.

## 3.1DATA FLOW DIAGRAM

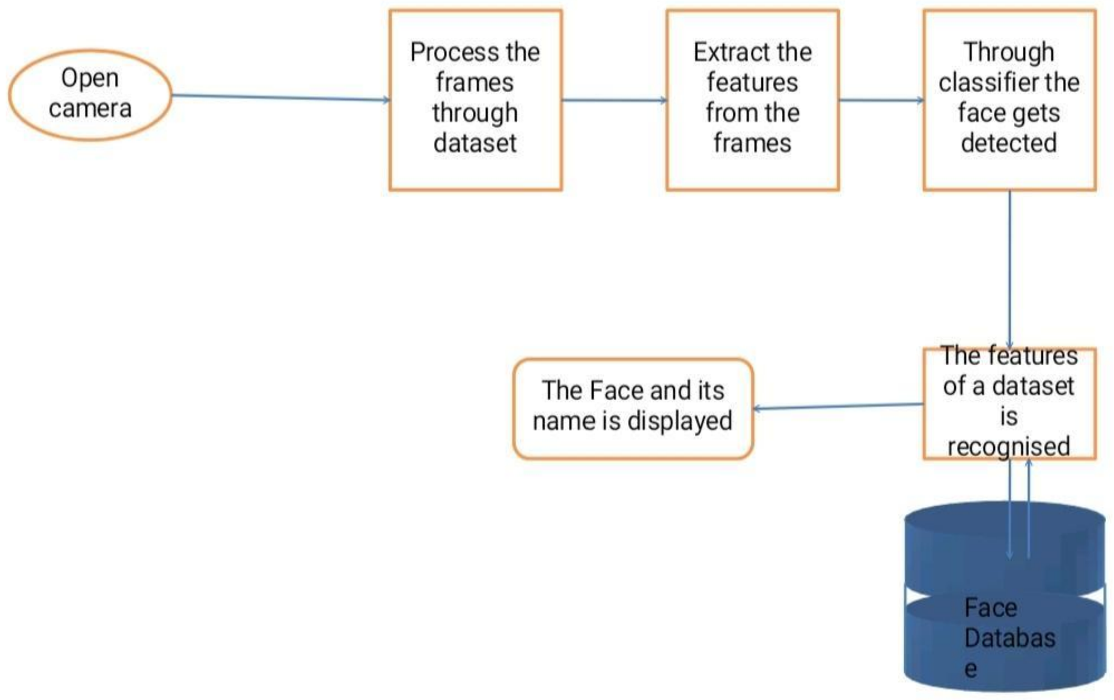
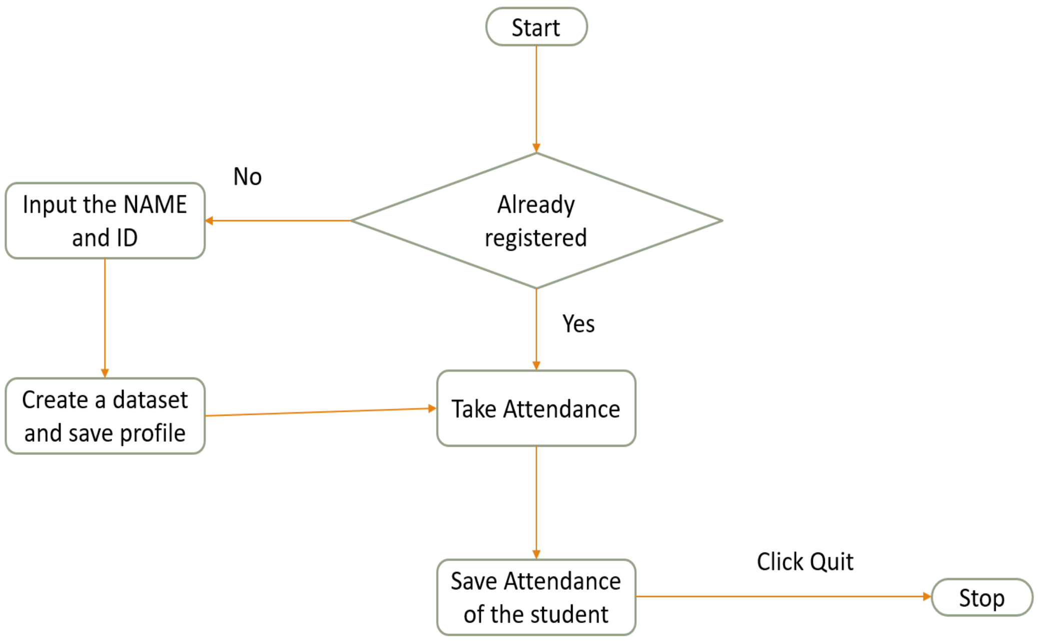

Open camera
Process the frames through dataset
Extract the features from the frames
Through classifier the face gets detected
The features of a dataset is recognised
The Face and its name is displayed
Face Databas e

Figure1:Data Flow Diagram

Figure 1 represents the Data Flow of our project where the user gives an input and how the model processes the data and how faces are detected and then talking attendance.

## 3.2.ACTIVITY DIAGRAM :

Activity Diagrams to illustrate the flow of control in a system and refer to the steps involved in the execution of a use case. So, we basically depict workflows visually using an activity diagram. An activity diagram focuses on the condition of flow and the sequence in which it happens. We describe or depict what causes a particular event using an activity diagram.



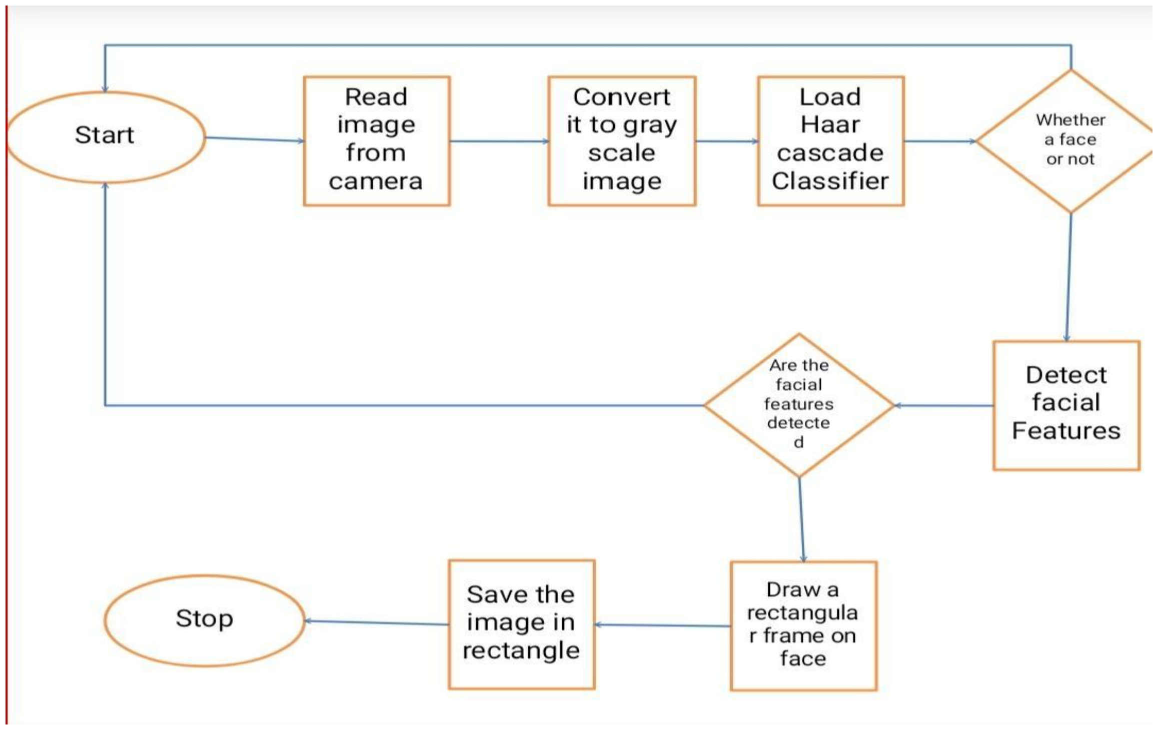

Start
Read image from camera
Convert it to gray scale image
Load Haar cascade Classifier
Whether a face or not
Detect facial Features
Are the facial features detecte d
Draw a rectangula r frame on face
Save the image in rectangle
Stop

Figure2: Activity Diagram

Figure 2 represents all the activities which are performed while the activity is being detected.

# 4. SOFTWARE ENVIRONMENT

Software environment is the term commonly used to refer to support an application. A software environment for a particular application includes the operating system, the database system, specific development tools or compiler.

4.1 Python Libraries and methods

4.1.1 **OS**: Python OS module provides the facility to establish the interaction between the user and the operating system. We have used this module to work with files and directories in the drive.

4.1.2 **Numpy**: Numpy is a general-purpose array-processing package. It provides a high performance multidimensional array object, and tools for working with these arrays. We have used Numpy library to convert lists to arrays and use them.

4.1.3 **tkinter**: It is used for widgets, buttons and output frames. **Ttk** comes with 18 widgets, twelve of which already existed in tkinter: Button, Check button, Entry, Frame, Label, Label Frame, Menu button, Paned Window, Radio button, Scale, Scrollbar, and Spin box. The **messagebox** module is used to display the message boxes in the python applications which are already present in the tkinter module.

4.1.4 **Pandas Library**: Pandas module runs on top of NumPy and it is popularly used for data science and data analytics.

4.1.5 **Pillow**: Pillow library contains all the basic image processing functionality like image resizing, rotation and transformation. Pillow module allows you to pull some statistics data out of the image using histogram method.

4.1.6 **CV2**: OpenCV has a function to read video, which is cv2. VideoCapture(). We can access our webcam using pass 0 in the function parameter.

4.1.7 **CSV**: The csv module implements classes to read and write tabular data in CSV format. It allows programmers to say, “write this data in the format preferred by Excel,” or “read data from this file which was generated by Excel,” without knowing the precise details of the CSV format used by Excel.

4.1.8 **Simpledialog**: simpledialog module contains convenience classes and functions for creating simple modal dialogs to get a value from the user.

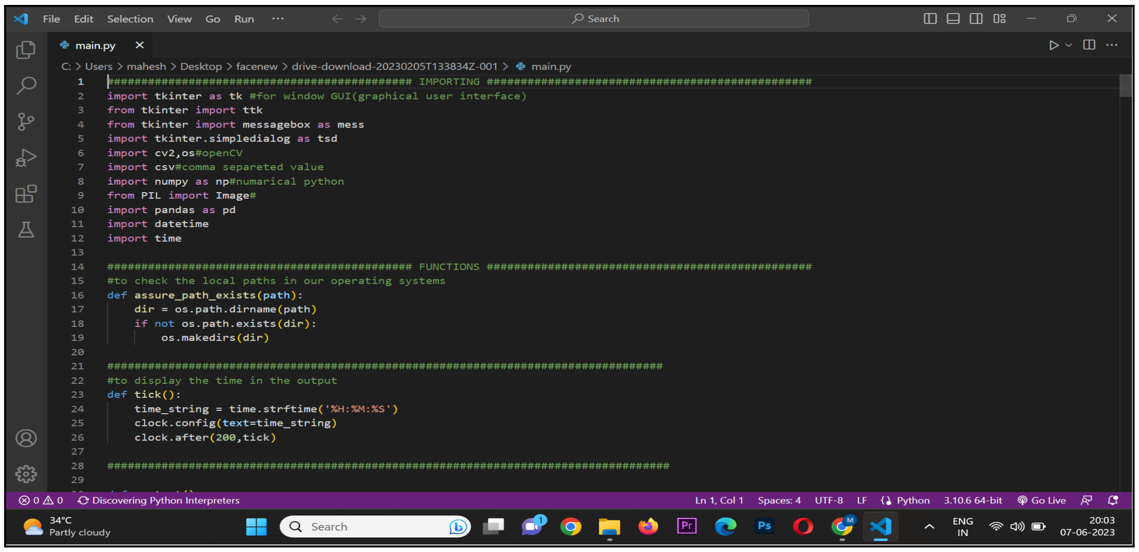
# 5.IMPLEMENTATION

Software implementation refers to the process of adopting and integrating a software application into a business workflow. Implementation of new tools and software into an enterprise can be complex, depending on the size of the organization and the software.

The software that we used to work on this project was VS Code.

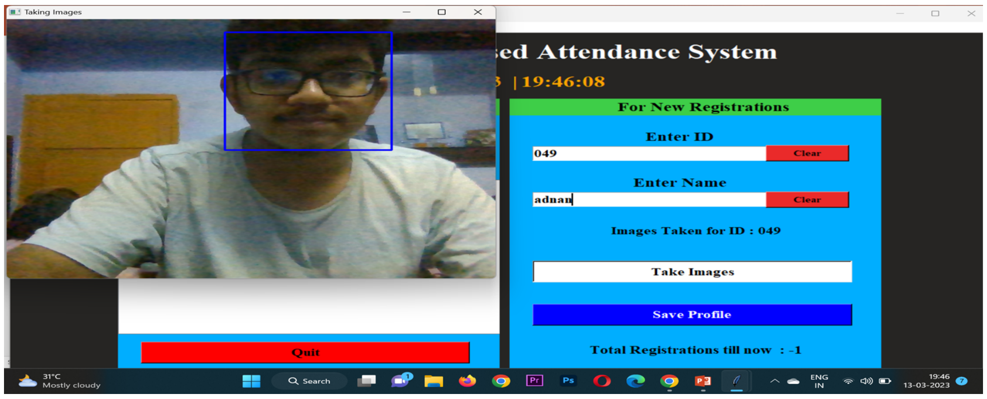
## VS Code:

It is an IDE i.e., Integrated Development Environment which has many features like it supports scientific tools (like matplotlib, numpy, scipy etc) web frameworks (example Django, web2py and Flask) refactoring in Python, integrated python debugger, code completion, code and project navigation etc.


File Edit Selection View Go Run ···
Search
main.py
C: > Users > mahesh > Desktop > facenew > drive-download-20230205T133834Z-001 > main.py
⊗ 0 ⚠ 0 Discovering Python Interpreters
Ln 1, Col 1 Spaces: 4 UTF-8 LF Python 3.10.6 64-bit Go Live
34°C Partly cloudy
Search
ENG IN
20:03 07-06-2023

```python
############################################# IMPORTING ################################################
import tkinter as tk #for window GUI(graphical user interface)
from tkinter import ttk
from tkinter import messagebox as mess
import tkinter.simpledialog as tsd
import cv2,os#openCV
import csv#comma separeted value
import numpy as np#numarical python
from PIL import Image#
import pandas as pd
import datetime
import time

############################################# FUNCTIONS ################################################
#to check the local paths in our operating systems
def assure_path_exists(path):
    dir = os.path.dirname(path)
    if not os.path.exists(dir):
        os.makedirs(dir)

##################################################################################
#to display the time in the output
def tick():
    time_string = time.strftime('%H:%M:%S')
    clock.config(text=time_string)
    clock.after(200,tick)

###################################################################################

```

When the face gets detectd, it shows the detected region of the face by drawing a rectangle around that region



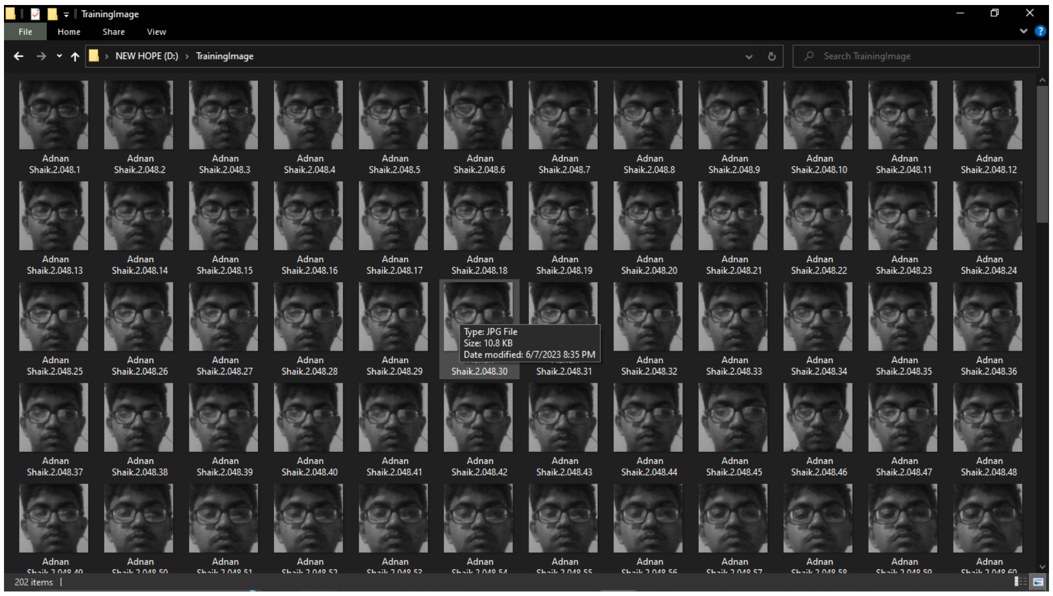
Enter ID
049

Enter Name
adnan

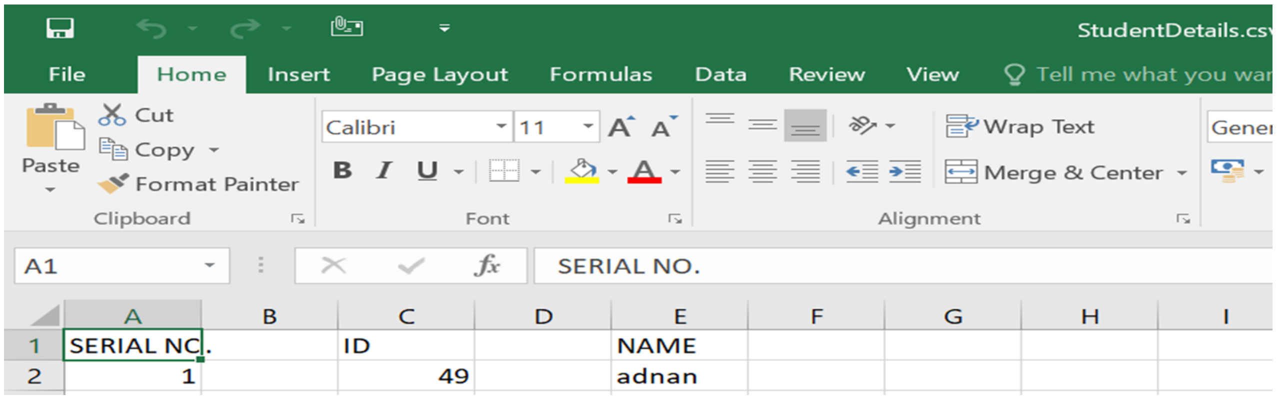
Images Taken for ID : 049

Total Registrations till now : -1

The window popped up shows the face getting detected, but in the background every frame is being captured then stored as a sample data.

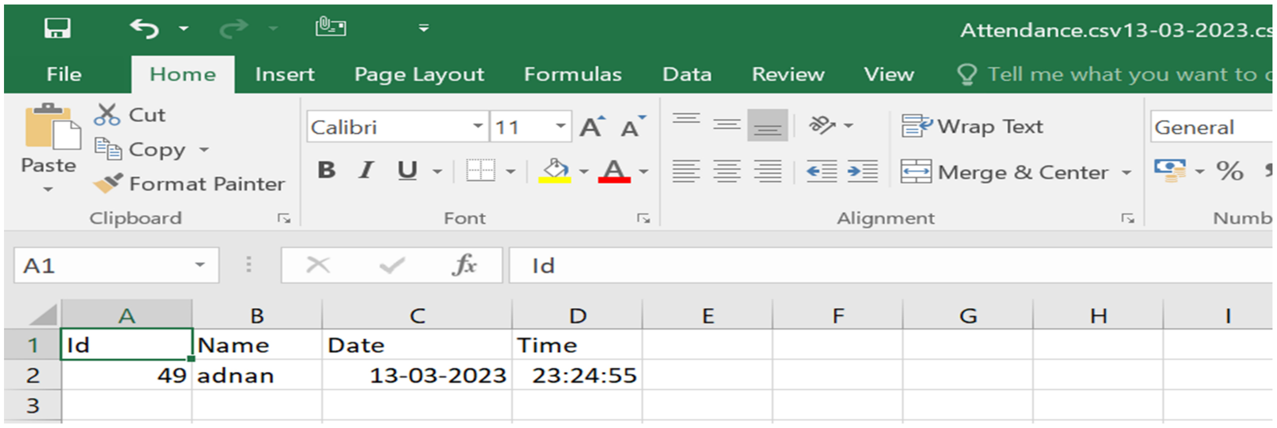


Now the information is stored in theStuden tdetails  CSV file.



| | A | B | C | D | E |
|---|---|---|---|---|---|
| 1 | SERIAL NO. | | ID | | NAME |
| 2 | 1 | | 49 | | adnan |

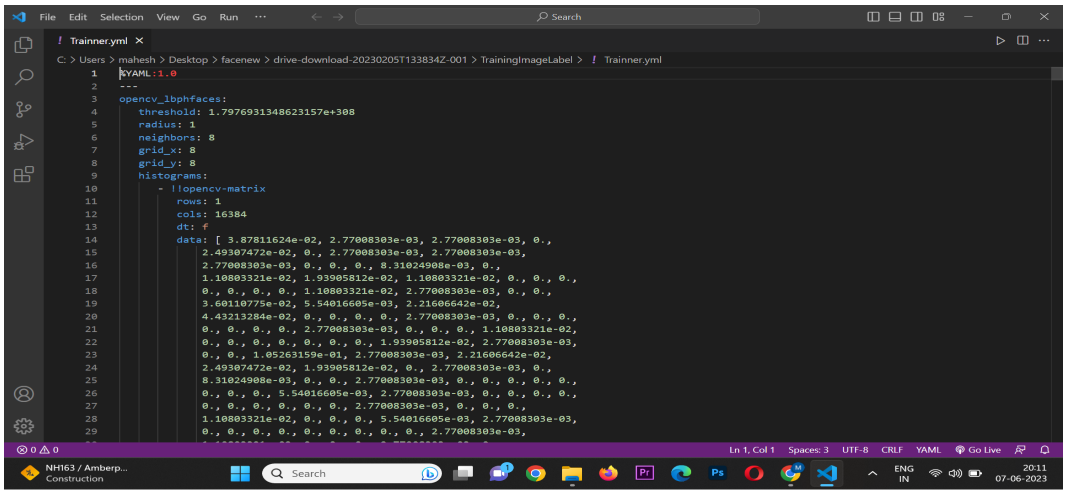
The attendance is stored in the Attendance CSV file.



| | A | B | C | D |
|---|---|---|---|---|
| 1 | Id | Name | Date | Time |
| 2 | 49 | adnan | 13-03-2023 | 23:24:55 |
| 3 | | | | |

The features are extracted after the sample data has been saved.

The trained images are converted into arrays of features and stored in .yml file.


Trainner.yml
C: > Users > mahesh > Desktop > facenew > drive-download-20230205T133834Z-001 > TrainingImageLabel > Trainner.yml

```yaml
%YAML:1.0
---
opencv_lbphfaces:
   threshold: 1.7976931348623157e+308
   radius: 1
   neighbors: 8
   grid_x: 8
   grid_y: 8
   histograms:
      - !!opencv-matrix
         rows: 1
         cols: 16384
         dt: f
         data: [ 3.87811624e-02, 2.77008303e-03, 2.77008303e-03, 0.,
             2.49307472e-02, 0., 2.77008303e-03, 2.77008303e-03,
             2.77008303e-03, 0., 0., 0., 8.31024908e-03, 0.,
             1.10803321e-02, 1.93905812e-02, 1.10803321e-02, 0., 0., 0.,
             0., 0., 0., 0., 1.10803321e-02, 2.77008303e-03, 0., 0.,
             3.60110775e-02, 5.54016605e-03, 2.21606642e-02,
             4.43213284e-02, 0., 0., 0., 0., 2.77008303e-03, 0., 0., 0.,
             0., 0., 0., 0., 2.77008303e-03, 0., 0., 0., 1.10803321e-02,
             0., 0., 0., 0., 0., 0., 0., 1.93905812e-02, 2.77008303e-03,
             0., 0., 1.05263159e-01, 2.77008303e-03, 2.21606642e-02,
             2.49307472e-02, 1.93905812e-02, 0., 2.77008303e-03, 0.,
             8.31024908e-03, 0., 0., 2.77008303e-03, 0., 0., 0., 0., 0.,
             0., 0., 0., 5.54016605e-03, 2.77008303e-03, 0., 0., 0., 0.,
             0., 0., 0., 0., 0., 0., 2.77008303e-03, 0., 0., 0.,
             1.10803321e-02, 0., 0., 0., 5.54016605e-03, 2.77008303e-03,
             0., 0., 0., 0., 0., 0., 0., 0., 0., 2.77008303e-03,
############################## IMPORTING#################################
import tkinter as tk #for window GUI(graphical user interface)
from tkinter import ttk
from tkinter import messagebox as mess
import tkinter.simpledialog as tsd
import cv2,os#openCV
import csv#comma separated value
import numpy as np#numerical python
from PIL import Image#
import pandas as pd
import datetime
import time



##############################FUNCTIONS ##################################
#to check the local paths in our operating systems
def assure_path_exists(path):
   dir = os.path.dirname(path)
   if not os.path.exists(dir):
        os.makedirs(dir)
###################################################################################
#to display the time in the output
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' ")
###################################################################################
#to check the haar cascade files whether it is exist or not
def check_haarcascadefile():
    exists = os.path.isfile("haarcascade_frontalface_default.xml")
    if exists:
        pass
    else:
        mess._show(title='Some file missing', message='Please install the required files')
        window.destroy()
###################################################################################
#to set the password
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()

###########################################################################

#to change the password

def change_pass():

    global master

    master = tk.Tk()

    master.geometry("400x160")

    master.resizable(False,False)

    master.title("Change Password")

    master.configure(background="white")

    lbl4 = tk.Label(master,text='    Enter Old Password',bg='white',font=('times', 12, '
bold '))

    lbl4.place(x=10,y=10)

    global old

    old=tk.Entry(master,width=25 ,fg="black",relief='solid',font=('times', 12, ' bold '
                    ),show='*')
    old.place(x=180,y=10)

    lbl5 = tk.Label(master, text='   Enter New Password', bg='white', font=('times', 12,
' bold '))

    lbl5.place(x=10, y=45)

    global new

    new = tk.Entry(master, width=25, fg="black",relief='solid', font=('times', 12, '
bold '),show='*')

    new.place(x=180, y=45)

    lbl6 = tk.Label(master, text='Confirm New Password', bg='white', font=('times',
12, ' bold '))

    lbl6.place(x=10, y=80)

    global nnew

    nnew = tk.Entry(master, width=25, fg="black", relief='solid',font=('times', 12, '
bold '),show='*')

    nnew.place(x=180, y=80)

    cancel=tk.Button(master,text="Cancel", command=master.destroy ,fg="black"
,bg="red" ,height=1,width=25 , activebackground = "white" ,font=('times', 10, ' bold
'))

    cancel.place(x=200, y=120)

    save1 = tk.Button(master, text="Save", command=save_pass, fg="black",
bg="#3ece48", height = 1,width=25, activebackground="white", font=('times', 10, '
bold '))

    save1.place(x=10, y=120)

    master.mainloop()

#########################################################################

#to check wether the entered password is correct or not to save profile

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')
```

```
############################################################################
#for clear buttons in the window screen
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)
############################################################################
#to take images
def TakeImages():
    check_haarcascadefile()
    columns = ['SERIAL NO.', '', 'ID', '', 'NAME']
    assure_path_exists("StudentDetails/")
    assure_path_exists("TrainingImage/")
    serial = 1
    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

            print(f'coord{img}')

        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)

###################################################################################

#to train images

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]))

####################################################################################

#to store the images with spacific features

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
```

```
###########################################################################

#to take the attendance and to recognize the faces which are already trained

def TrackImages():

    check_haarcascadefile()

    assure_path_exists("Attendance/Attendance.csv")

    assure_path_exists("StudentDetails/""StudentDetails.csv")

    for k in tv.get_children():

        tv.delete(k)

    msg = ''

    i = 0

    j = 0

    recognizer =  cv2.face.LBPHFaceRecognizer_create()

    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 #fonts in the frame


    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)

        attendance = None

        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])

            print(serial, conf)

            Id = 'Unknown'


            bb = str(Id)
```

```
          if (conf < 50):
             try:
                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[0]
                ID = df.loc[df['SERIAL NO.'] == serial]['ID'].values[0]
                ID = str(ID)
                # ID = ID[1:-1]
                bb = str(aa)
                # bb = bb[2:-2]
                attendance = [str(ID), bb, str(date), str(timeStamp)]
             except Exception as ex:
                print(ex)
                pass
          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')
```

```
## Attendance Data Sheet as Data Source
attendance_file = "Attendance\Attendance.csv" + date + ".csv"


## Wait 5 Seconds before display
#from time import sleep
attendance_file_exists = os.path.exists(attendance_file)
## Only to write the headers
#attendance_file_exists = os.path.isfile(attendance_file)


with open(attendance_file, 'a') as current_attendance_file:
    headers = ['Id', 'Name', 'Date', 'Time']
    csvwriter = csv.DictWriter(current_attendance_file,
                               delimiter=",",
                               lineterminator="\n",
                               fieldnames=headers)
    if not attendance_file_exists:
        ## This means there is no attendance file present
        ## Create a new file with headers
        csvwriter.writeheader()


    if attendance:
        attendance_taken = False
        ## To handle existing information
```

```
            attendance_df = pd.read_csv(attendance_file, header=0)

            if not attendance_df.empty and len(attendance_df.index) != 0 and
attendance_file_exists:

                for index, row in attendance_df.iterrows():

                    #print("HELLO THERE")

                    if(int(attendance[0]) == row["Id"]):

                        print(f'Attendance of {row["Name"]} already taken!')

                        attendance_taken = True

            ## Write the usual attendance record

            attendance_data = dict(zip(headers, attendance))


            print("***** Hello this is CSV Write *****")


            print(f'Original Attendance Data ::: {attendance}')

            print(f'Attendance record is ::: {attendance_data}')

            if not attendance_taken:

                csvwriter.writerow(attendance_data)


    ## Below snippet will show data on the UI

    with open(attendance_file, 'r') as csvFile1:

        reader1 = list(csv.reader(csvFile1))

        tvidx = 0

        for rows in reader1[1:]:
            print(f"Lines ::: {rows}")

            tv.insert('', 'end', text= str(rows[0]), values=(str(rows[1]), str(rows[2]),
str(rows[3])))

            tvidx += 1

            # i = i + 1

            # if (i > 1):

            #     if (i % 2 != 0):

            #         iidd = str(lines[0]) + '   '

            #         tv.insert('', 0, text=iidd, values=(str(lines[0]), str(lines[1]),
str(lines[2])))



    cam.release()

    cv2.destroyAllWindows()

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

    #to show the month in the window screen

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 #############################

#to organize all the inputed options in the proper way

window = tk.Tk()

window.geometry("1280x720")

window.resizable(True,False)

window.title("Attendance System")

window.configure(background='#262523')


frame1 = tk.Frame(window, bg="#00aeff")

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


frame2 = tk.Frame(window, bg="#00aeff")
```

```
frame2.place(relx=0.51, rely=0.17, relwidth=0.38, relheight=0.80)


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

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


frame3 = tk.Frame(window, bg="#c4c6ce")

frame3.place(relx=0.52, rely=0.09, relwidth=0.09, relheight=0.07)


frame4 = tk.Frame(window, bg="#c4c6ce")

frame4.place(relx=0.36, rely=0.09, relwidth=0.16, relheight=0.07)


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

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


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

clock.pack(fill='both',expand=1)

tick()


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

head2.grid(row=0,column=0)
```

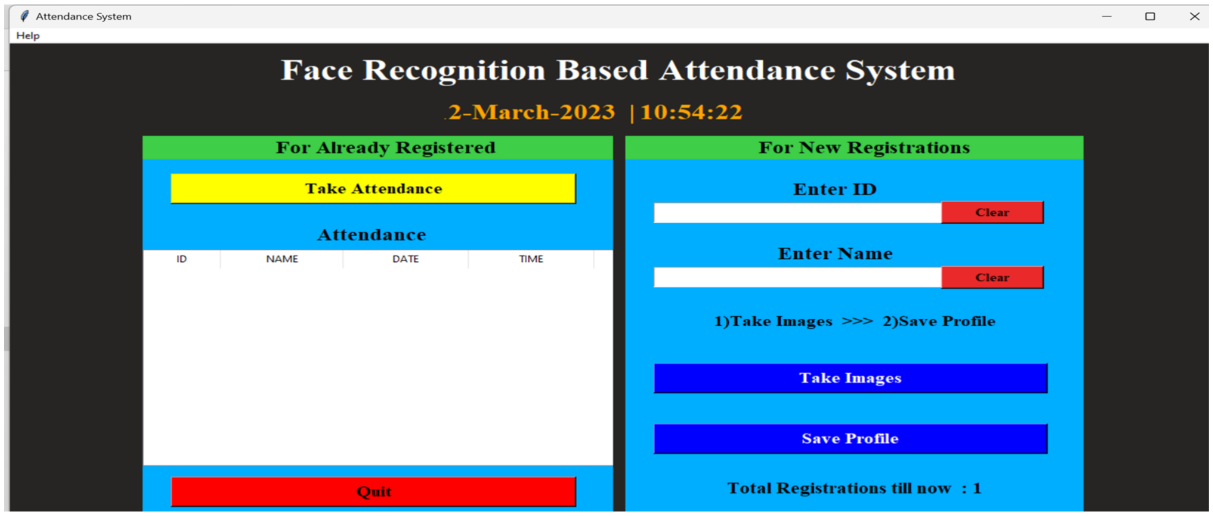
```
head1 = tk.Label(frame1, text="              For Already Registered                        ",
fg="black",bg="#3ece48" ,font=('times', 17, ' bold ') )

head1.place(x=0,y=0)


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

lbl.place(x=80, y=55)


txt = tk.Entry(frame2,width=32 ,fg="black",font=('times', 15, ' bold '))

txt.place(x=30, y=88)


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

lbl2.place(x=80, y=140)


txt2 = tk.Entry(frame2,width=32 ,fg="black",font=('times', 15, ' bold ')  )

txt2.place(x=30, y=173)


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

message1.place(x=7, y=230)


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

message.place(x=7, y=450)
```

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

lbl3.place(x=100, y=115)



res=0

exists = os.path.isfile("StudentDetails\StudentDetails.csv")

if exists:

    with open("StudentDetails\StudentDetails.csv", 'r') as csvFile1:

        reader1 = csv.reader(csvFile1)

        for l in reader1:

            res = res + 1

    res = (res // 2) - 1

    csvFile1.close()

else:

    res = 1

message.configure(text='Total Registrations till now  : '+str(res))

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

#for menu option

menubar = tk.Menu(window,relief='ridge')

filemenu = tk.Menu(menubar,tearoff=0)

filemenu.add_command(label='Change Password', command = change_pass)

filemenu.add_command(label='Contact Us', command = contact)
filemenu.add_command(label='Exit',command = window.destroy)

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

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

#to show the attence in the window screen

tv= ttk.Treeview(frame1,height =13,columns = ('id', 'name','date','time'))

tv.column('#0',width=82)

tv.column('#1',width=130)

tv.column('#2',width=133)

tv.column('#3',width=133)

#tv.grid(row=2,column=0,padx=(0,0),pady=(150,0),columnspan=4)

tv.grid(row=0,column=4,padx=(0,0),pady=(150,0),columnspan=4)

tv.heading('#0',text ='ID')

tv.heading('#1',text ='NAME')

tv.heading('#2',text ='DATE')

tv.heading('#3',text ='TIME')

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

#scrool bar when the attence is heavy

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 ##################################

#to insert widgets in window screen

clearButton = tk.Button(frame2, text="Clear", command=clear  ,fg="black"
,bg="#ea2a2a"  ,width=11 ,activebackground = "white" ,font=('times', 11, ' bold '))
```

```
clearButton.place(x=335, y=86)

clearButton2 = tk.Button(frame2, text="Clear", command=clear2  ,fg="black"
,bg="#ea2a2a"  ,width=11 , activebackground = "white" ,font=('times', 11, ' bold '))

clearButton2.place(x=335, y=172)

takeImg = tk.Button(frame2, text="Take Images", command=TakeImages
,fg="white"  ,bg="blue"  ,width=34  ,height=1, activebackground = "white"
,font=('times', 15, ' bold '))

takeImg.place(x=30, y=300)

trainImg = tk.Button(frame2, text="Save Profile", command=psw ,fg="white"
,bg="blue"  ,width=34  ,height=1, activebackground = "white" ,font=('times', 15, '
bold '))

trainImg.place(x=30, y=380)

trackImg = tk.Button(frame1, text="Take Attendance", command=TrackImages
,fg="black"  ,bg="yellow"  ,width=35  ,height=1, activebackground = "white"
,font=('times', 15, ' bold '))

trackImg.place(x=30,y=50)

quitWindow = tk.Button(frame1, text="Quit", command=window.destroy
,fg="black"  ,bg="red"  ,width=35 ,height=1, activebackground = "white"
,font=('times', 15, ' bold '))

quitWindow.place(x=30, y=450)

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

#to run the total GUI

window.configure(menu=menubar)

window.mainloop()

##########################################################################
```

## 6.SAMPLE SCREEN

Here is a sample of how the interface of the attendance system would look like.

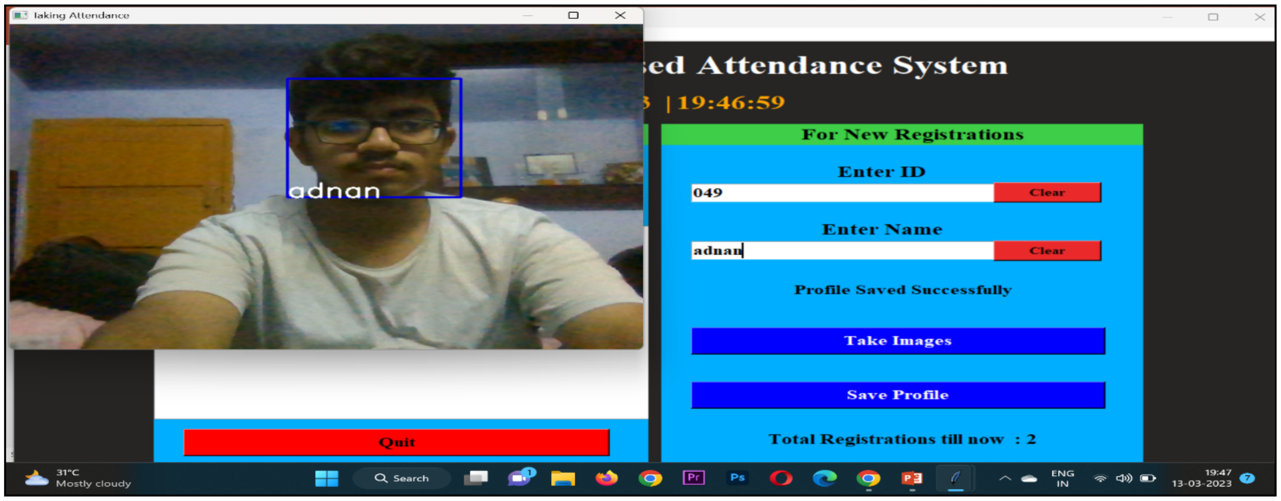


Attendance System

Help

Face Recognition Based Attendance System

2-March-2023 | 10:54:22



Taking Attendance

ed Attendance System

3 | 19:46:59

# 7.CONCLUSION

We have developed a Machine learning model that detects the activities which include Student attendance System in education institutions or institutions where an individual has to record his/her presence.

The primary reason for implementing this project is to improve the attendance  systems and reduce human efforts and improve accuracy.

The whole project is portable.

The project provides an executable file which can be carried out and installed on systems(admin system).

# 8.REFERENCES

https://github.com/

https://pypi.org/project/opencv-python/