```python
############################################ 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 : 'bhanuganesh342@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="red")
    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)
```

```
140 def clear2():
141     txt2.delete(0, 'end')
142     res = "1)Take Images  >>>  2)Save Profile"
143     message1.configure(text=res)
144 
145 ##################################################################################
146 
147 def TakeImages():
148     check_haarcascadefile()
149     columns = ['SERIAL NO.', '', 'ID', '', 'NAME']
150     assure_path_exists("StudentDetails/")
151     assure_path_exists("TrainingImage/")
152     serial = 0
153     exists = os.path.isfile("StudentDetails\StudentDetails.csv")
154     if exists:
155         with open("StudentDetails\StudentDetails.csv", 'r') as csvFile1:
156             reader1 = csv.reader(csvFile1)
157             for l in reader1:
158                 serial = serial + 1
159         serial = (serial // 2)
160         csvFile1.close()
161     else:
162         with open("StudentDetails\StudentDetails.csv", 'a+') as csvFile1:
163             writer = csv.writer(csvFile1)
164             writer.writerow(columns)
165             serial = 1
166         csvFile1.close()
167     Id = (txt.get())
168     name = (txt2.get())
169     if ((name.isalpha()) or (' ' in name)):
170         cam = cv2.VideoCapture(0)
171         harcascadePath = "haarcascade_frontalface_default.xml"
172         detector = cv2.CascadeClassifier(harcascadePath)
173         sampleNum = 0
174         while (True):
175             ret, img = cam.read()
176             gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
177             faces = detector.detectMultiScale(gray, 1.3, 5)
178             for (x, y, w, h) in faces:
179                 cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
180                 # incrementing sample number
181                 sampleNum = sampleNum + 1
182                 # saving the captured face in the dataset folder TrainingImage
183                 cv2.imwrite("TrainingImage\ " + name + "." + str(serial) + "." + Id + '.' + str(sampleNum) + ".jpg",
184                             gray[y:y + h, x:x + w])
185                 # display the frame
186                 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]))

##########################################################################################3

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)

###################### 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)
```

```
462 trainImg = tk.Button(frame2, text="Save Profile", command=psw ,fg="white"  ,bg="#6d00fc"  ,width=34  ,height=1, activebackground = "white" ,font=('
    comic', 15, ' bold '))
463 trainImg.place(x=30, y=380)
464 trackImg = tk.Button(frame1, text="Take Attendance", command=TrackImages  ,fg="black"  ,bg="#3ffc00"  ,width=35  ,height=1, activebackground = "
    white" ,font=('comic', 15, ' bold '))
465 trackImg.place(x=30,y=50)
466 quitWindow = tk.Button(frame1, text="Quit", command=window.destroy  ,fg="black"  ,bg="#eb4600"  ,width=35 ,height=1, activebackground = "white" ,
    font=('comic', 15, ' bold '))
467 quitWindow.place(x=30, y=450)
468
469 ###################### END ######################################
470
471 window.configure(menu=menubar)
472 window.mainloop()
473
474 ##################################################################################################
475
```