CHAPTER-8
APPENDIX 1

PROGRAM FOR INSERTING STUDENT DETAILS

```
# -*- coding: utf-8 -*-
"""
Created on Sat Feb  9 16:49:51 2018

@author: jaimuruganantham
"""
#Import statements
import tkinter as tk
from tkinter import Message ,Text
import cv2,os
import csv
import numpy as np
from PIL import Image, ImageTk
import pandas as pd
import datetime
import time
import tkinter.ttk as ttk
import tkinter.font as font
from tkinter import messagebox
import sqlite3
import urllib3
import urllib
```

```
#Email
import smtplib
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
from email.mime.base import MIMEBase
from email import encoders
url='http://192.168.43.1:8080/shot.jpg'
def AdminLogin():
    windowAdmin=tk.Tk()
    windowAdmin.title(">>>>>Login<<<<<<")
    windowAdmin.configure(background='light blue')
    #Title of the window
    message = tk.Label(windowAdmin, text="AUTOMATED
ATTENDANCE BY IMAGE PROCESSING USING IOT" ,bg="light
cyan"  ,fg="blue"  ,width=60  ,height=2,font=('times', 30, 'italic bold
underline'))
    message.place(x=70, y=20)



    lbl = tk.Label(windowAdmin, text="User ID:",width=20  ,height=2
,fg="red"  ,bg="sky blue" ,font=('times', 15, ' bold ') )
    lbl.place(x=400, y=300)


    txt = tk.Entry(windowAdmin,width=20  ,bg="snow2"
,fg="red",font=('times', 15, ' bold '))
    txt.place(x=700, y=315)
    lbl2 = tk.Label(windowAdmin, text="Password",width=20  ,fg="red"
,bg="sky blue"    ,height=2 ,font=('times', 15, ' bold '))
    lbl2.place(x=400, y=400)


    txt2 = tk.Entry(windowAdmin,show="*",width=20  ,bg="snow2"
,fg="red",font=('times', 15, ' bold ')  )
    txt2.place(x=700, y=415)


    def login():
        if txt.get()=="" and txt2.get()=="":
            messagebox.showerror("Login","Enter the valid datas")
            txt.delete(0,'end')
            txt2.delete(0,'end')
        elif txt.get()=="":
            messagebox.showerror("Login","Enter the valid User Id")
            txt2.delete(0,'end')
        elif txt2.get()=="":
            messagebox.showerror("Login","Enter the valid Password")
            txt.delete(0,'end')
        else:
            UserId=int(txt.get())
            Password=txt2.get()
            conn=sqlite3.connect("StudentDataBase.db")
            cursor2=conn.execute("select UserId,Password from Login")
            def check(UserId):
                cmd="SELECT * FROM Login WHERE UserId="+str(UserId)
                cursur1=conn.execute(cmd)
                isRecordExit=0
                for row in cursur1:
                    isRecordExit=1
                return isRecordExit
            record=check(UserId)
            if record!=1:
                messagebox.showerror("Login","UnAuthorized User")
            else:
                for row in cursor2:
                    if UserId==row[0] and Password==row[1]:
                        messagebox.showinfo("Login","Your are Loged in")
                        conn.commit()
                        conn.close()
                        windowAdmin.destroy()
                        AdminUser()
                        break;
    def signin():
        window1=tk.Tk()
        window1.title(">>>>>Login<<<<<<")
        window1.configure(background='light blue')
        #Title of the window
        message = tk.Label(window1, text="AUTOMATED
ATTENDANCE BY IMAGE PROCESSING USING IOT" ,bg="light
cyan"  ,fg="blue"  ,width=60  ,height=2,font=('times', 30, 'italic bold
underline'))
```

```
        message.place(x=70, y=20)


        message = tk.Label(window1, text="Signin" ,bg="light cyan"
,fg="blue"  ,width=50  ,height=1,font=('times', 30, 'italic bold underline'))

        message.place(x=170, y=150)


        lbl = tk.Label(window1, text="User ID:",width=20  ,height=2
,fg="red"  ,bg="sky blue" ,font=('times', 15, ' bold ') )

        lbl.place(x=400, y=300)


        txt = tk.Entry(window1,width=20  ,bg="snow2"
,fg="red",font=('times', 15, ' bold '))

        txt.place(x=700, y=315)


        lbl2 = tk.Label(window1, text="Password",width=20  ,fg="red"
,bg="sky blue"    ,height=2 ,font=('times', 15, ' bold '))

        lbl2.place(x=400, y=400)


        txt2 = tk.Entry(window1,show="*",width=20  ,bg="snow2"
,fg="red",font=('times', 15, ' bold ')  )

        txt2.place(x=700, y=415)


        def sign():

            UserId=int(txt.get())

            Password=txt2.get()

            conn=sqlite3.connect("StudentDataBase.db")

            cursor=conn.execute("select UserId,Password from Login")
```

```
    def check(UserId):
        cmd="SELECT * FROM Login WHERE UserId="+str(UserId)
        cursur=conn.execute(cmd)
        isRecordExit=0
        for row in cursur:
            isRecordExit=1
        return isRecordExit
    record=check(UserId)
    if record==1:
        messagebox.showerror("Signin","User Already signed in")
    elif record!=1:
        conn.execute("""insert into Login
values(?,?)""",(UserId,Password,))
        messagebox.showinfo("Signin","You are signed in!! Enjoy...")
    conn.commit()
    conn.close()
    window1.destroy()

LoginButton=tk.Button(window1,text="Signin",command=sign,width=1
0,height=1,fg="red",bg="sky blue",font=('times',15,'bold'))
  LoginButton.place(x=900,y=515)


  window1.mainloop()
```

```
LoginButton=tk.Button(windowAdmin,text="Login",command=login,width=10,height=1,fg="red",bg="sky blue",font=('times',15,'bold'))
    LoginButton.place(x=700,y=515)

LoginButton=tk.Button(windowAdmin,text="Signin",command=signin,width=10,height=1,fg="red",bg="sky blue",font=('times',15,'bold'))
    LoginButton.place(x=900,y=515)


    windowAdmin.mainloop()
def AdminUser():
    windowAdminUser=tk.Tk()
    windowAdminUser.title("Admin User")
    windowAdminUser.configure(background='light blue')
    message = tk.Label(windowAdminUser, text="AUTOMATED ATTENDANCE BY IMAGE PROCESSING USING IOT" ,bg="light cyan"  ,fg="blue"  ,width=60  ,height=2,font=('times', 30, 'italic bold underline'))
    message.place(x=70, y=20)
    message = tk.Label(windowAdminUser, text="Admin" ,bg="light cyan"  ,fg="blue"  ,width=50  ,height=1,font=('times', 30, 'italic bold underline'))
    message.place(x=170, y=150)


    def Update1():
        windowAdminUser.destroy()
        Update()
        StartAttendance(txt.get())
    lbl = tk.Label(windowAdminUser, text="Staff Email ID:",width=20
,height=2  ,fg="red"  ,bg="sky blue" ,font=('times', 15, ' bold ') )

    lbl.place(x=400, y=240)



    txt = tk.Entry(windowAdminUser,width=20  ,bg="snow2"
,fg="red",font=('times', 15, ' bold '))

    txt.place(x=700, y=250)

    StudentWindow = tk.Button(windowAdminUser, text="Update
Student Information", command=Update1 ,fg="red"  ,bg="sky blue"
,width=20  ,height=1, activebackground = "Red" ,font=('times', 15, ' bold
'))

    StudentWindow.place(x=650, y=300)

    FaceTrainWindow = tk.Button(windowAdminUser, text="Face Train",
command=TrainImages  ,fg="red"  ,bg="sky blue"  ,width=20  ,height=1,
activebackground = "Red" ,font=('times', 15, ' bold '))

    FaceTrainWindow.place(x=650, y=350)

    FaceDetectionWindow = tk.Button(windowAdminUser, text="Face
Detection Sample", command=TrackImages ,fg="red"  ,bg="sky blue"
,width=20  ,height=1, activebackground = "Red" ,font=('times', 15, ' bold
'))

    FaceDetectionWindow.place(x=650, y=400)

    MianAttendanceWindow = tk.Button(windowAdminUser, text="Start
Attendance", command=StartAttendance1  ,fg="red"  ,bg="sky blue"
,width=20  ,height=1, activebackground = "Red" ,font=('times', 15, ' bold
'))

    MianAttendanceWindow.place(x=650, y=450)

    quiteWindow = tk.Button(windowAdminUser, text="LogOut",
command=AdminLogin1  ,fg="red"  ,bg="sky blue"  ,width=20
,height=1, activebackground = "Red" ,font=('times', 15, ' bold '))

    quiteWindow.place(x=650, y=500)

    windowAdminUser.mainloop()
```

```
def Registration():
    #Creating Window for GUI
    windowRegistration = tk.Tk()
    #Creating Title for Window
    windowRegistration.title("Face_Recogniser")
    #Setting Backgroud Color
    windowRegistration.configure(background='light blue')
    #Configuring windows
    windowRegistration.grid_rowconfigure(0, weight=1)
    windowRegistration.grid_columnconfigure(0, weight=1)

    message = tk.Label(windowRegistration, text="AUTOMATED
ATTENDANCE BY IMAGE PROCESSING USING IOT" ,bg="light
cyan"  ,fg="blue"  ,width=60  ,height=3,font=('times', 30, 'italic bold
underline'))

    message.place(x=70, y=20)

    lbl = tk.Label(windowRegistration, text="Enter ID:",width=20
,height=2  ,fg="red"  ,bg="sky blue" ,font=('times', 15, ' bold ') )
    lbl.place(x=400, y=200)

    txt = tk.Entry(windowRegistration,width=20  ,bg="snow2"
,fg="red",font=('times', 15, ' bold '))
    txt.place(x=700, y=215)

    lbl2 = tk.Label(windowRegistration, text="Enter Name",width=20
,fg="red"  ,bg="sky blue"    ,height=2 ,font=('times', 15, ' bold '))
```

```
    lbl2.place(x=400, y=300)


    txt2 = tk.Entry(windowRegistration,width=20  ,bg="snow2"
,fg="red",font=('times', 15, ' bold ')  )

    txt2.place(x=700, y=315)


    lbl2 = tk.Label(windowRegistration, text="Email:",width=20
,fg="red"  ,bg="sky blue"    ,height=2 ,font=('times', 15, ' bold '))

    lbl2.place(x=400, y=400)


    txt3 = tk.Entry(windowRegistration,width=20  ,bg="snow2"
,fg="red",font=('times', 15, ' bold ')  )

    txt3.place(x=700, y=415)


    lbl3 = tk.Label(windowRegistration, text="Notification : ",width=20
,fg="red"  ,bg="sky blue"  ,height=2 ,font=('times', 15, ' bold underline '))

    lbl3.place(x=400, y=500)


    message = tk.Label(windowRegistration, text="" ,bg="snow2"
,fg="red"  ,width=30  ,height=2, activebackground = "yellow"
,font=('times', 15, ' bold '))

    message.place(x=700, y=500)


    def clear():

        txt.delete(0, 'end')

        res = ""

        message.configure(text= res)
```

```
def clear2():
    txt2.delete(0, 'end')
    res = ""
    message.configure(text= res)


def is_number(s):
    try:
        float(s)
        return True
    except ValueError:
        pass


    try:
        import unicodedata
        unicodedata.numeric(s)
        return True
    except (TypeError, ValueError):
        pass


    return False


def TakeImages():
    try:
        Ids=int(txt.get())
    except:
        messagebox.showerror("Error","Enter the value Id")
        windowRegistration.destroy()
        Registration()
    if txt.get()=="" and txt2.get()=="":
        messagebox.showerror("Error","Invalid data")
        windowRegistration.destroy()
        Registraiton()
    else:
        conn=sqlite3.connect("StudentDataBase.db")
        cur=conn.cursor()
        registrationNo=int(txt.get())
        def insertStudentDetails(registrationNo):
            cmd="SELECT * FROM Student1 WHERE RegistrationNo="+str(registrationNo)
            cursur=conn.execute(cmd)
            isRecordExit=0
            for row in cursur:
                isRecordExit=1
            return isRecordExit
        record=insertStudentDetails(registrationNo)
        if record==1:
            messagebox.showerror("Ivalid Data","Registration Number Already exist")
            windowRegistration.destroy()
            Registration()
        Id=(txt.get())
        name=(txt2.get())
        email="jaimruganantham@gmail.com"
        if(is_number(Id) and name.isalpha()):
            cam = cv2.VideoCapture(0)
            harcascadePath = "haarcascade_frontalface_default.xml"
            detector=cv2.CascadeClassifier(harcascadePath)
            sampleNum=0
            while(True):
                ret, img = cam.read()
                img=cv2.flip(img,1)
                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 +"."+Id +'.'+
str(sampleNum) + ".jpg", gray[y:y+h,x:x+w])
                    #display the frame
                    cv2.imshow('frame',img)
                #wait for 100 miliseconds
                if cv2.waitKey(100) & 0xFF == ord('q'):
                    break
                # break if the sample number is morethan 100
                elif sampleNum>60:
                    break
            cam.release()
            cv2.destroyAllWindows()
            res = "Images Saved for ID : " + Id +" Name : "+ name
            row = [Id , name, email]
            with open('StudentDetails\StudentDetails.csv','a+') as csvFile:
                writer = csv.writer(csvFile)
                writer.writerow(row)
            csvFile.close()
            message.configure(text= res)
            cur.execute('''insert into Student1(RegistrationNo)
values(?)''',(registrationNo,))
            conn.commit()
            conn.close()
            messagebox.showinfo("Image","Successfully Registered ..!")
        else:
            if(is_number(Id)):
                res = "Enter Alphabetical Name"
                message.configure(text= res)
                messagebox.showerror("Error","Enter the Alphabetical
Name")
            elif(name.isalpha()):
                res = "Enter Numeric Id"
                message.configure(text= res)
                messagebox.showerror("Invalid data","Enter the Numeri Id")
```

```
          else:
            res="Enter the valid data"
            message.configure(text=res)
            messagebox.showerror("Ivalid Data","Enter the valid data")

    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)]
        #print(imagePaths)

        #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 Back():
        windowRegistration.destroy()
        Update()
    clearButton = tk.Button(windowRegistration, text="Clear",
command=clear  ,fg="red"  ,bg="cadet blue"  ,width=20  ,height=2
,activebackground = "Red" ,font=('times', 15, ' bold '))
    clearButton.place(x=950, y=200)
    clearButton2 = tk.Button(windowRegistration, text="Clear",
command=clear2  ,fg="red"  ,bg="cadet blue"  ,width=20  ,height=2,
activebackground = "Red" ,font=('times', 15, ' bold '))
    clearButton2.place(x=950, y=300)
    takeImg = tk.Button(windowRegistration, text="Register",
command=TakeImages ,fg="red"  ,bg="cadet blue"  ,width=20
,height=1, activebackground = "Red" ,font=('times', 15, ' bold '))
    takeImg.place(x=700, y=600)
    takeImg = tk.Button(windowRegistration, text="Back",
command=Back ,fg="red"  ,bg="cadet blue"  ,width=20  ,height=1,
activebackground = "Red" ,font=('times', 15, ' bold '))
    takeImg.place(x=950, y=600)
    copyWrite = tk.Text(windowRegistration,
background=windowRegistration.cget("background"),
borderwidth=0,font=('times', 30, 'italic bold underline'))
    copyWrite.tag_configure("superscript", offset=10)
    copyWrite.configure(state="disabled",fg="red"  )
    copyWrite.pack(side="left")
    copyWrite.place(x=800, y=750)
    windowRegistration.mainloop()
```

```
def TrainImages():
    recognizer = cv2.face_LBPHFaceRecognizer.create()#recognizer = 
cv2.face.LBPHFaceRecognizer_create()#$cv2.createLBPHFaceRecogniz
er()
    harcascadePath = "haarcascade_frontalface_default.xml"
    detector =cv2.CascadeClassifier(harcascadePath)
    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)]
        #print(imagePaths)

        #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
    faces,Id = getImagesAndLabels("TrainingImage")
    recognizer.train(faces, np.array(Id))
    recognizer.save("Trainner.yml")
    res = "Image Trained"#+",".join(str(f) for f in Id)
    messagebox.showinfo("Trainner","Hi Your data is trainned")


def TrackImages():
    recognizer = 
cv2.face.LBPHFaceRecognizer_create()#cv2.createLBPHFaceRecognize
r()
    recognizer.read("Trainner.yml")
    harcascadePath = "haarcascade_frontalface_default.xml"
    faceCascade = cv2.CascadeClassifier(harcascadePath);
    df=pd.read_csv("StudentDetails\StudentDetails.csv")
    cam = cv2.VideoCapture(0)
    font = cv2.FONT_HERSHEY_SIMPLEX
    col_names =  ['Id','Name','Date','Time']
    attendance = pd.DataFrame(columns = col_names)
    while True:
        ret, im =cam.read()
        im=cv2.flip(im,1)
        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)
            Id, conf = recognizer.predict(gray[y:y+h,x:x+w])
            if(conf < 50):
                ts = time.time()
                date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-
%d')
                timeStamp =
datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
                aa=df.loc[df['Id'] == Id]['Name'].values
                tt=str(Id)+"-"+aa
                attendance.loc[len(attendance)] = [Id,aa,date,timeStamp]
            else:
                Id='Unknown'
                tt=str(Id)
            if(conf > 75):
                noOfFile=len(os.listdir("ImagesUnknown"))+1
                cv2.imwrite("ImagesUnknown\Image"+str(noOfFile) + ".jpg",
im[y:y+h,x:x+w])
            cv2.putText(im,str(tt),(x,y+h), font, 1,(255,255,255),2)
        cv2.imshow('im',im)
        if (cv2.waitKey(1)==ord('q')):
            break
    cam.release()
    cv2.destroyAllWindows()
    messagebox.showinfo("Images","Successfully Verified.....")
```

```
def Update():

    windowUpdate=tk.Tk()

    windowUpdate.title("Admin User")

    windowUpdate.configure(background='light blue')

    message = tk.Label(windowUpdate, text="AUTOMATED
ATTENDANCE BY IMAGE PROCESSING USING IOT" ,bg="light
cyan"  ,fg="blue"  ,width=60  ,height=2,font=('times', 30, 'italic bold
underline'))

    message.place(x=70, y=20)

    message = tk.Label(windowUpdate, text="Student Updater" ,bg="light
cyan"  ,fg="blue"  ,width=50  ,height=1,font=('times', 30, 'italic bold
underline'))

    message.place(x=170, y=150)

    def Registration1():

        windowUpdate.destroy()

        Registration()

    def AdminUser1():

        windowUpdate.destroy()

        AdminUser()

    def Del():

        windowUpdate.destroy()

        DeleteUser()

    FaceDetectionWindow = tk.Button(windowUpdate, text="Register
Details", command=Registration1 ,fg="red"  ,bg="sky blue"  ,width=20
,height=1, activebackground = "Red" ,font=('times', 15, ' bold '))

    FaceDetectionWindow.place(x=650, y=400)

    MianAttendanceWindow = tk.Button(windowUpdate, text="Delete
Details", command=Del ,fg="red"  ,bg="sky blue"  ,width=20  ,height=1,
activebackground = "Red" ,font=('times', 15, ' bold '))
```

```
    MianAttendanceWindow.place(x=650, y=450)

    quiteWindow = tk.Button(windowUpdate, text="Back",
command=AdminUser1  ,fg="red"  ,bg="sky blue"  ,width=20  ,height=1,
activebackground = "Red" ,font=('times', 15, ' bold '))

    quiteWindow.place(x=650, y=500)

    windowUpdate.mainloop()


def StartAttendance(Email):

    email=Email

    recognizer =
cv2.face.LBPHFaceRecognizer_create()#cv2.createLBPHFaceRecognize
r()

    recognizer.read("Trainner.yml")

    faceCascade = cv2.CascadeClassifier(harcascadePath);

    df=pd.read_csv("StudentDetails\StudentDetails.csv")

    cam = cv2.VideoCapture(0)

    font = cv2.FONT_HERSHEY_SIMPLEX

    col_names =  ['Id','Name','Date','Time']

    c_emails=['Email']

    attendance = pd.DataFrame(columns = col_names)

    def Email(email,fileName):

        email_user = 'attendancedetails19@gmail.com'

        email_password = 'Attendance@19'

        email_send = email


        subject = 'Attendance Details'
```

```
msg = MIMEMultipart()
msg['From'] = email_user
msg['To'] = email_send
msg['Subject'] = subject

body = 'Government college of engineering Computer science and engineering department Attendance details'
msg.attach(MIMEText(body,'plain'))

filename=fileName
attachment  =open(filename,'rb')

part = MIMEBase('application','octet-stream')
part.set_payload((attachment).read())
encoders.encode_base64(part)
part.add_header('Content-Disposition',"attachment; filename= "+filename)

msg.attach(part)
text = msg.as_string()
server = smtplib.SMTP('smtp.gmail.com',587)
server.starttls()
server.login(email_user,email_password)
server.sendmail(email_user,email_send,text)
server.quit()
```

```
    while True:
        #imgResp=urllib.request.urlopen(url)
        #imgNp=np.array(bytearray(imgResp.read()),dtype=np.uint8)
        #im=cv2.imdecode(imgNp,-1)
        ret, im =cam.read()
        img=cv2.flip(im,1)
        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)
            Id, conf = recognizer.predict(gray[y:y+h,x:x+w])
            if(conf < 50):
                ts = time.time()
                date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-
%d')
                timeStamp =
datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
                aa=df.loc[df['Id'] == Id]['Name'].values
                #email=df.loc[df['Id'] == Id]['Email'].values
                tt=str(Id)+"-"+aa
                attendance.loc[len(attendance)] = [Id,aa,date,timeStamp]
            else:
                Id='Unknown'
                tt=str(Id)
            if(conf > 75):
                noOfFile=len(os.listdir("ImagesUnknown"))+1
                cv2.imwrite("ImagesUnknown\Image"+str(noOfFile) + ".jpg",
im[y:y+h,x:x+w])
            cv2.putText(im,str(tt),(x,y+h), font, 1,(255,255,255),2)
        attendance=attendance.drop_duplicates(subset=['Id'],keep='first')
        cv2.imshow('im',im)
        if (cv2.waitKey(1)==ord('q')):
            break
    ts = time.time()
    date = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d')
    timeStamp =
datetime.datetime.fromtimestamp(ts).strftime('%H:%M:%S')
    Hour,Minute,Second=timeStamp.split(":")
    fileName="Attendance\Attendance_"+date+"_"+Hour+"-"+Minute+"-
"+Second+".csv"
    emailfileName="Attendance\Email"+date+"_"+Hour+"-"+Minute+"-
"+Second+".csv"
    attendance.to_csv(fileName,index=False)
    cam.release()
    cv2.destroyAllWindows()
    Email(email,fileName)
    with open(fileName) as csv_file:
        csv_reader = csv.reader(csv_file, delimiter=',')
        line_count = 0
        for row in csv_reader:
            if line_count == 0:
                line_count += 1
            else:
                s=f'{row[0]}'
                with open('StudentDetails\StudentDetails.csv') as csv_file2:
                    csv_reader2 = csv.reader(csv_file2, delimiter=',')
                    line_count = 0
                    for row2 in csv_reader2:
                        if line_count == 0:
                            line_count += 1
                        else:
                            if(s==row2[0]):
                                email=f'{row2[2]}'
                                file="Hall Ticket.docx"
                                Email(email,file)
                            line_count += 1
                line_count += 1
    messagebox.showinfo("Images","Successfully Attendance
taken...Enjoy.")


def DeleteUser():
    windowDelete=tk.Tk()
    windowDelete.title("Delete User")
    windowDelete.configure(background='light blue')
    message = tk.Label(windowDelete, text="AUTOMATED
ATTENDANCE BY IMAGE PROCESSING USING IOT" ,bg="light
cyan"  ,fg="blue"  ,width=60  ,height=2,font=('times', 30, 'italic bold
underline'))
    message.place(x=70, y=20)
    message = tk.Label(windowDelete, text="Admin" ,bg="light cyan"
,fg="blue"  ,width=50  ,height=1,font=('times', 30, 'italic bold underline'))

    message.place(x=170, y=150)

    lbl = tk.Label(windowDelete, text="Enter ID:",width=20  ,height=2
,fg="red"  ,bg="sky blue" ,font=('times', 15, ' bold ') )

    lbl.place(x=400, y=400)

    txt = tk.Entry(windowDelete,width=20  ,bg="snow2"
,fg="red",font=('times', 15, ' bold '))

    txt.place(x=700, y=415)

    def DeleteUser2():

        conn=sqlite3.connect("StudentDataBase.db")

        registrationNo=int(txt.get())

        def deleteStudentDetails(registrationNo):

            cmd="SELECT * FROM Student1 WHERE
RegistrationNo="+str(registrationNo)

            cursur=conn.execute(cmd)

            isRecordExit=0

            for row in cursur:

                isRecordExit=1

            return isRecordExit

        record=deleteStudentDetails(registrationNo)

        if record==1:

            cmd="DELETE FROM Student1 WHERE
RegistrationNo="+str(registrationNo)

            conn.execute(cmd)

            messagebox.showinfo("DeletedConfirmation","Deleted
sucessfully")

            conn.commit()
        else:

            messagebox.showinfo("Deleted","Oops!There is no such
registration n1umber")

        conn.close()

    def Up():

        windowDelete.destroy()

        Update()

    LoginButton=tk.Button(windowDelete,text="Deltet
User",command=DeleteUser2,width=10,height=1,fg="red",bg="sky
blue",font=('times',15,'bold'))

    LoginButton.place(x=700,y=515)


LoginButton=tk.Button(windowDelete,text="Back",command=Up,width
=10,height=1,fg="red",bg="sky blue",font=('times',15,'bold'))

    LoginButton.place(x=900,y=515)

    windowDelete.mainloop()

AdminLogin()
```