**CODE EXPLAINATION**

**Importing Images**

As we have imported before we can use the same face\_recognition.load\_image\_file() function to import our images. But when we have multiple images, importing them individually can become messy. Therefore we will write a script to import all images in a given folder at once. For this we will need the os library so we will import that first. We will store all the images in one list and their names in another.

import face\_recognition  
import cv2  
import numpy as np  
import os  
from datetime import datetime

path = 'Images'  
images = [] # LIST CONTAINING ALL THE IMAGES  
className = [] # LIST CONTAINING ALL THE CORRESPONDING CLASS Names  
myList = os.listdir(path)  
print(myList)  
for cl in myList:  
 curImg = cv2.imread(f'{path}/{cl}')  
 images.append(curImg)  
 className.append(os.path.splitext(cl)[0])

A person smiling for the picture

Description automatically generated with low confidenceA person wearing glasses

Description automatically generated with medium confidenceA person in a suit

Description automatically generated with medium confidence

**Compute Encodings**

Now that we have a list of images we can iterate through those and create a corresponding encoded list for known faces. To do this we will create a function. As earlier we will first convert it into RGB and then find its encoding using the face\_encodings() function. Then we will append each encoding to our list.

def findEncodings(images):  
 encodeList = []  
 for img in images:  
 img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)  
 encode = face\_recognition.face\_encodings(img)[0]  
 encodeList.append(encode)  
 return encodeList

Now we can simply call this function with the images list as the input arguments.

encodeListKnown = findEncodings(images)  
print('Encodings Complete')

**The While loop**

The while loop is created to run the webcam. But before the while loop we have to create a video capture object so that we can grab frames from the webcam.

cap = cv2.VideoCapture(0)

**Webcam Image**

First we will read the image from the webcam and then resize it to quarter the size. This is done to increase the speed of the system. Even though the image being used is 1/4 th of the original, we will still use the original size while displaying. Next we will convert it to RGB.

while True:  
 success, img = cap.read()  
 imgS = cv2.resize(img, (0, 0), fx=0.25, fy=0.25)  
 imgS = cv2.cvtColor(imgS, cv2.COLOR\_BGR2RGB)

**Webcam Encodings**

Once we have the webcam frame we will find all the faces in our image. The face\_locations function is used for this purpose. Later we will find the face\_encodings as well.

facesCurFrame = face\_recognition.face\_locations(imgS)  
encodesCurFrame = face\_recognition.face\_encodings(imgS, facesCurFrame)

**Find Matches**

Now we can match the current face encodings to our known faces encoding list to find the matches. We will also compute the distance. This is done to find the best match in case more than one face is detected at a time.

for encodeFace, faceLoc in zip(encodesCurFrame, facesCurFrame):  
 matches = face\_recognition.compare\_faces(encodeListKnown, encodeFace)  
 faceDis = face\_recognition.face\_distance(encodeListKnown, encodeFace)

Once we have the list of face distances we can find the minimum one, as this would be the best match.

matchIndex = np.argmin(faceDis)

Now based on the index value we can determine the name of the person and display it on the original Image.

if matches[matchIndex]:  
 name = classNames[matchIndex].upper()  
 # print(name)  
 y1, x2, y2, x1 = faceLoc  
 y1, x2, y2, x1 = y1 \* 4, x2 \* 4, y2 \* 4, x1 \* 4  
 cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 2)  
 cv2.rectangle(img, (x1, y2 - 35), (x2, y2), (0, 255, 0), cv2.FILLED)  
 cv2.putText(img, name, (x1 + 6, y2 - 6), cv2.FONT\_HERSHEY\_COMPLEX, 1, (255, 255, 255), 2)  
 markAttendance(name)

**Marking Attendance**

Lastly we are going to add the automated attendance code. We will start by writing a function that requires only one input which is the name of the user. First we open our Attendance file which is in csv format. Then we read all the lines and iterate through each line using a for loop. Next we can split using comma ‘,’. This will allow us to get the first element which is the name of the user. If the user in the camera already has an entry in the file then nothing will happen. On the other hand if the user is new then the name of the user along with the current time stamp will be stored. We can use the datetime class in the date time package to get the current time.

def markAttendance(name):  
 with open('Attendence.csv','r+') as f:  
 myDataList = f.readlines()  
 nameList =[]  
 for line in myDataList:  
 entry = line.split(',')  
 nameList.append(entry[0])  
 if name not in line:  
 now = datetime.now()  
 dt\_string = now.strftime("%H:%M:%S")  
 f.writelines(f'\n{name},{dt\_string}')

Graphical user interface, text, application, chat or text message

Description automatically generatedGraphical user interface, application, table, Excel

Description automatically generated

**Labeling Unknown faces as well**

To find the unknown faces we will replace this

if matches[matchIndex]:  
 name = classNames[matchIndex].upper()  
 # print(name)  
 y1, x2, y2, x1 = faceLoc  
 y1, x2, y2, x1 = y1 \* 4, x2 \* 4, y2 \* 4, x1 \* 4  
 cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 2)  
 cv2.rectangle(img, (x1, y2 - 35), (x2, y2), (0, 255, 0), cv2.FILLED)  
 cv2.putText(img, name, (x1 + 6, y2 - 6), cv2.FONT\_HERSHEY\_COMPLEX, 1, (255, 255, 255), 2)  
 markAttendance(name)

with this:

if faceDis[matchIndex] < 0.50:  
 name = classNames[matchIndex].upper()  
 markAttendance(name)  
else:  
 name = 'Unknown'  
# print(name)  
y1, x2, y2, x1 = faceLoc  
y1, x2, y2, x1 = y1 \* 4, x2 \* 4, y2 \* 4, x1 \* 4  
cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 2)  
cv2.rectangle(img, (x1, y2 - 35), (x2, y2), (0, 255, 0), cv2.FILLED)  
cv2.putText(img, name, (x1 + 6, y2 - 6), cv2.FONT\_HERSHEY\_COMPLEX, 1, (255, 255, 255), 2)

All this does is to check if the distance to our min face is less than 0.5 or not. If its not then this means the person is unknown so we change the name to unknown and don’t mark the attendance.