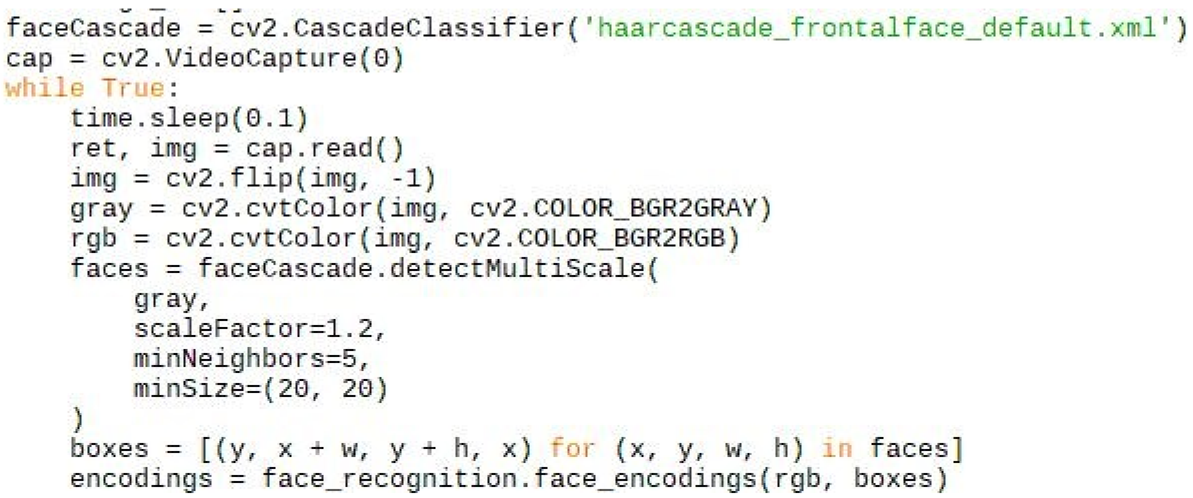
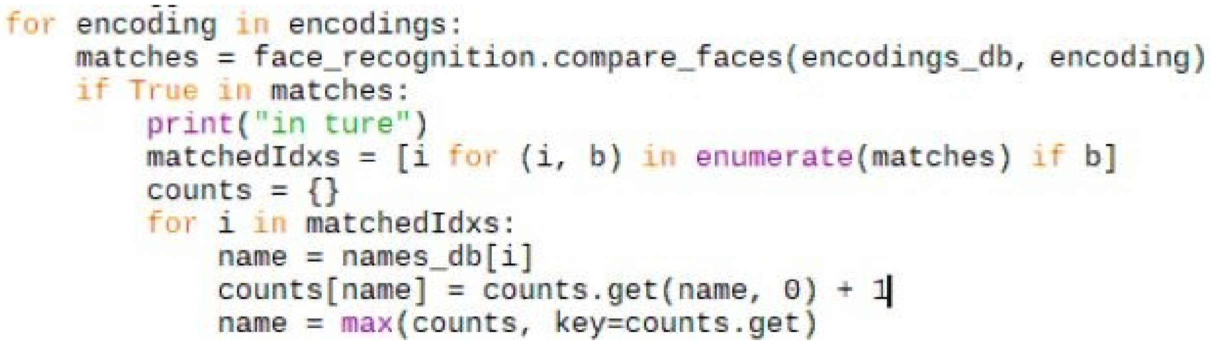
Practicum Face Recognition on Raspberry Pi

In this project I have four functions, first function is called camera_stream this function allow me to capture video frame from Pi camera using OpenCV then I convert the frame to gray level then detect faces in video frame using cascade classifier, then I pass this faces to face_recognition library to get face encoding for every face in frame and store it to encoding list.



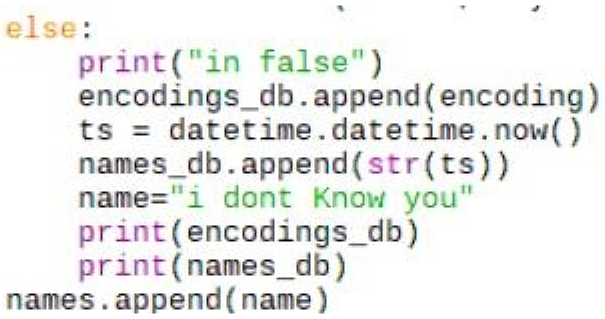
```
faceCascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
cap = cv2.VideoCapture(0)
while True:
    time.sleep(0.1)
    ret, img = cap.read()
    img = cv2.flip(img, -1)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    faces = faceCascade.detectMultiScale(
        gray,
        scaleFactor=1.2,
        minNeighbors=5,
        minSize=(20, 20)
    )
    boxes = [(y, x + w, y + h, x) for (x, y, w, h) in faces]
    encodings = face_recognition.face_encodings(rgb, boxes)
```

Then after get encoding I start comparing it with stored database using face_recognition library then I check if there is a match, if its true its going to determine the recognized face with the largest number of votes (note: in the event of an unlikely tie Python will select first entry in the dictionary) this case happen when a face match more than one encoding then I call the stored data in database for matching encoding and added to name list which will be viewed on screen.



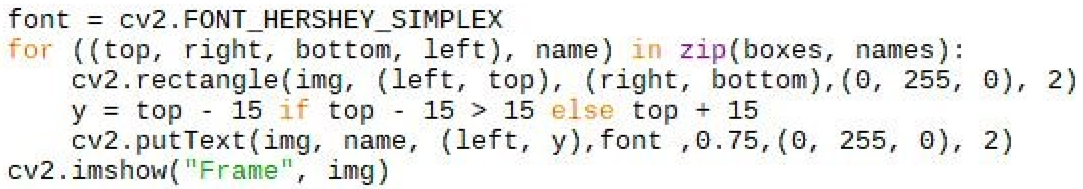
```
for encoding in encodings:
    matches = face_recognition.compare_faces(encodings_db, encoding)
    if True in matches:
        print("in ture")
        matchedIdxs = [i for (i, b) in enumerate(matches) if b]
        counts = {}
        for i in matchedIdxs:
            name = names_db[i]
            counts[name] = counts.get(name, 0) + 1
            name = max(counts, key=counts.get)
```

If false, this mean that its the first time to see this face it's going to add “I don’t know you” to names list to be viewed on screen then add this encoding to database then add current time to database as the name of this encoding.



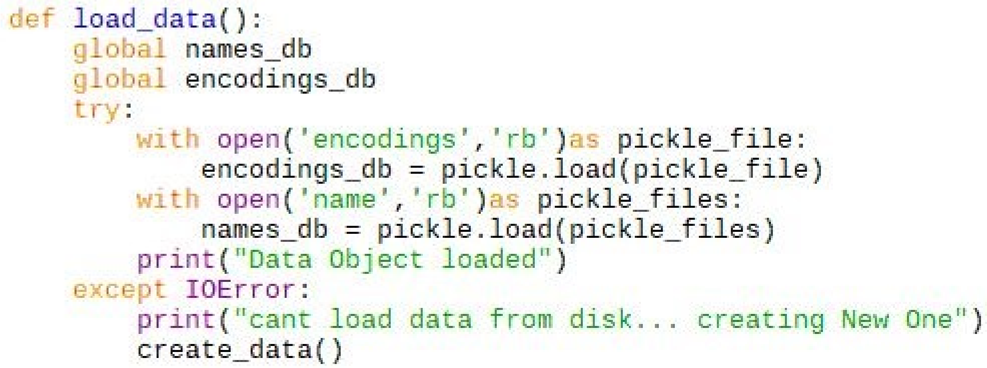
```
else:
    print("in false")
    encodings_db.append(encoding)
    ts = datetime.datetime.now()
    names_db.append(str(ts))
    name="i dont Know you"
    print(encodings_db)
    print(names_db)
names.append(name)
```

Then after getting all names for faces in this frame I show it to screen using OpenCV by putting a rectangle and put its name on it



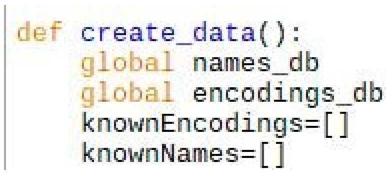
```
font = cv2.FONT_HERSHEY_SIMPLEX
for ((top, right, bottom, left), name) in zip(boxes, names):
    cv2.rectangle(img, (left, top), (right, bottom),(0, 255, 0), 2)
    y = top - 15 if top - 15 > 15 else top + 15
    cv2.putText(img, name, (left, y),font ,0.75,(0, 255, 0), 2)
cv2.imshow("Frame", img)
```

Second function is called load_data which allow you to load database from hard drive using pickle library database is two lists encodings and name, if it couldn’t load for any error it will create a new database.



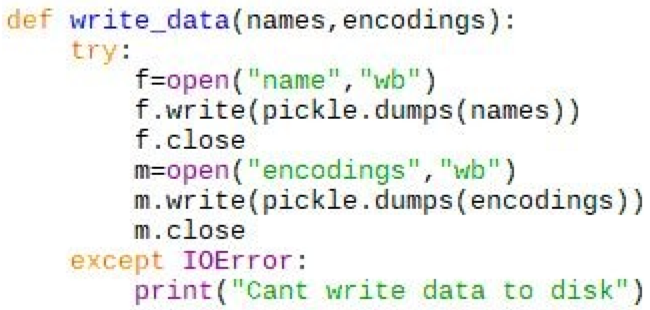
```
def load_data():
    global names_db
    global encodings_db
    try:
        with open('encodings','rb')as pickle_file:
            encodings_db = pickle.load(pickle_file)
        with open('name','rb')as pickle_files:
            names_db = pickle.load(pickle_files)
        print("Data Object loaded")
    except IOError:
        print("cant load data from disk... creating New One")
        create_data()
```

Third function is called create_data which allow you to create new database.



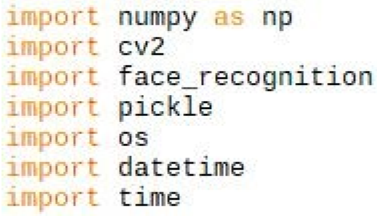
```
def create_data():
    global names_db
    global encodings_db
    knownEncodings=[]
    knownNames=[]
```

Last function is called write_data which allow you to write database lists to hard drive using pickle library.



```
def write_data(names,encodings):
    try:
        f=open("name","wb")
        f.write(pickle.dumps(names))
        f.close
        m=open("encodings","wb")
        m.write(pickle.dumps(encodings))
        m.close
    except IOError:
        print("Cant write data to disk")
```

In this project I have used below libraries:



```
import numpy as np
import cv2
import face_recognition
import pickle
import os
import datetime
import time
```

Below references and resources helped me to install some off this library and some helped me for face detection using OpenCV.

Note: every time I restart the Pi I have to write this command to make camera work with OpenCV. “sudo modprobe bcm2835-v4l2”

Reference and resources:

https://github.com/ageitgey/face_recognition

https://www.hackster.io/mjrobot/real-time-face-recognition-an-end-to-end-project-a10826

https://www.pyimagesearch.com/2017/09/04/raspbian-stretch-install-opencv-3-python-on-your-raspberry-pi/

https://www.pyimagesearch.com/2018/06/25/raspberry-pi-face-recognition/