CAMERA:

def comm():

import cv2

faceCascade = cv2.CascadeClassifier('C:/Users/Naithanis/AppData/Roaming/SPB\_Data/facial\_recognition/Cascades/haarcascade\_frontalface\_default.xml')

cap = cv2.VideoCapture(0)

cap.set(3,720) # set Width

cap.set(4,720) # set Height

while True:

ret, img = cap.read()

img = cv2.flip(img, 1)

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

faces = faceCascade.detectMultiScale(

gray,

scaleFactor=1.2,

minNeighbors=10,

minSize=(20, 20)

)

for (x,y,w,h) in faces:

cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,0),2)

roi\_gray = gray[y:y+h, x:x+w]

roi\_color = img[y:y+h, x:x+w]

cv2.imshow('video',img)

k = cv2.waitKey(30) & 0xff

if k == 27: # press 'ESC' to quit

break

##PROGRAMME END

FRONT END:

from tkinter import \*

from PIL import ImageTk,Image

import camera

import datagathering

import train\_face

from name\_storage import name

import database\_display

#import recogniser

t = Tk(className="Facial\_Recognition")

canvas=Canvas(t,width=800,height=349)

image=ImageTk.PhotoImage(Image.open("C:\\Users\\Naithanis\\AppData\\Roaming\\SPB\_Data\\facial\_recognition\\background.png"))

canvas.create\_image(0,0,anchor=NW , image=image)

def user():

datagathering.input()

name()

train\_face.train()

def capture():

camera.comm()

def database():

database\_display.disp()

Label(t, text = 'FACE RECOGNITION [ADMINSTRATOR MODULE]',font='bold',bg='white',fg='black').place(x=230,y=20)

Button(t,text="CAMERA",bg='black',fg='white',command=capture).place(x=150,y=260,width=200,height=50)

Button(t,text="FACE REGISTRATION",bg='black',fg='white',command=user).place(x=450,y=260,width=200,height=50)

Button(t,text="REGISTERED DATABASE",bg='black',fg='white',command=database).place(x=300,y=200,width=200,height=50)

canvas.pack()

t.mainloop()

DATA GATHERING:

from id\_input import user

def input():

import cv2

cam = cv2.VideoCapture(0)

cam.set(3, 620) # set video width

cam.set(4, 620) # set video height

face\_detector = cv2.CascadeClassifier('C:/Users/Naithanis/AppData/Roaming/SPB\_Data/facial\_recognition/Cascades/haarcascade\_frontalface\_default.xml')

#For each person, enter one numeric face id

face\_id=user()

print(face\_id)

print("/n [INFO] Initializing face capture. Please wait ....")

# print(str(face\_id))

# Initialize individual sampling face count

count = 0

while(True):

ret, img = cam.read()

img = cv2.flip(img, 1)

gray = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

faces = face\_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)

count += 1

# Save the captured image into the datasets folder

cv2.imwrite("C:/Users/Naithanis/AppData/Roaming/SPB\_Data/facial\_recognition/storage/capture." +str(face\_id)+ '.' +

str(count) + ".jpg", gray[y:y+h,x:x+w])

cv2.imshow('image', img)

k = cv2.waitKey(100) & 0xff # Press 'ESC' for exiting video

if k == 27:

break

elif count >= 20: # Take 8 face sample and stop video

break

# Do a bit of cleanup

print("/n [INFO] Exiting Program")

cv2.destroyAllWindows()

ID INPUT:

def user():

from tkinter import Canvas,NW

from PIL import ImageTk,Image

import tkinter as tk

import cv2

root=tk.Toplevel()

root.title("User\_Id")

# setting the windows size

canvas=Canvas(root,width=800,height=349)

image=ImageTk.PhotoImage(Image.open("C:\\Users\\Naithanis\\AppData\\Roaming\\SPB\_Data\\facial\_recognition\\background.png"))

canvas.create\_image(0,0,anchor=NW , image=image)

canvas.pack()

#for storing id

# defining a function that will get the id

id\_var=tk.StringVar()

def submit():

root.destroy()

# creating a label for

# id using widget Label

id\_label = tk.Label(root, text = 'ENTER USER ID:',font='bold',bg="black",fg="white")

# creating a entry for input

# id using widget Entry

id\_entry = tk.Entry(root,textvariable=id\_var)

# creating a button using the widget

# Button that will call the submit function

sub\_btn=tk.Button(root,text = 'SUBMIT',command = submit)

# placing the label and entry in

# the required position using grid method

id\_label.place(x=360,y=200)

id\_entry.place(x=360,y=230)

sub\_btn.place(x=390,y=260)

# executing a finite loop till

# the window to is destroyed by clicking submit

root.wait\_window(root)

#returning the value in id\_var

return id\_var.get()

USER\_NAME INPUT AND NAME\_STORAGE:

def name\_input():

from tkinter import Canvas,NW

from PIL import ImageTk,Image

import tkinter as tk

root=tk.Toplevel()

root.title("person\_name")

canvas=Canvas(root,width=800,height=349)

image=ImageTk.PhotoImage(Image.open("C:\\Users\\Naithanis\\AppData\\Roaming\\SPB\_Data\\facial\_recognition\\background.png"))

canvas.create\_image(0,0,anchor=NW , image=image)

canvas.pack()

#name\_submit command

name\_var=tk.StringVar()

def name\_submit():

root.destroy()

#To prnt label

name\_label=tk.Label(root,text="USER NAME",font="bold")

#To take input prom the user

name\_entry=tk.Entry(root,textvariable=name\_var)

#Btn to submit the input taken from the user

btn=tk.Button(root,text="Submit",font="bold",command=name\_submit)

name\_label.place(x=360,y=200)

name\_entry.place(x=360,y=230)

btn.place(x=390,y=260)

root.wait\_window(root)

return name\_var.get()

def name():

#code to save the person's name to the txt file

from user\_name\_input import name\_input

name\_file=open("user.txt","a")

name\_file.write(name\_input() + "\n")

name\_file.close()

TRAIN FACE:

def train():

import cv2

import numpy as np

from PIL import Image

import os

# Path for face image database

path = 'C:/Users/Naithanis/AppData/Roaming/SPB\_Data/facial\_recognition/storage'

recognizer = cv2.face.LBPHFaceRecognizer\_create()

detector = cv2.CascadeClassifier("C:/Users/Naithanis/AppData/Roaming/SPB\_Data/facial\_recognition/Cascades/haarcascade\_frontalface\_default.xml");

# function to get the images and label data

def getImagesAndLabels(path):

imagePaths = [os.path.join(path,f) for f in os.listdir(path)]

faceSamples=[]

ids = []

for imagePath in imagePaths:

PIL\_img = Image.open(imagePath).convert('L') # grayscale

img\_numpy = np.array(PIL\_img,'uint8')

id = int(os.path.split(imagePath)[-1].split(".")[1])

faces = detector.detectMultiScale(img\_numpy)

for (x,y,w,h) in faces:

faceSamples.append(img\_numpy[y:y+h,x:x+w])

ids.append(id)

return faceSamples,ids

print ("/n [INFO] Training faces. It will take a few seconds. Wait ...")

faces,ids = getImagesAndLabels(path)

recognizer.train(faces, np.array(ids))

# Save the model into trainer/trainer.yml

recognizer.write('C:/Users/Naithanis/AppData/Roaming/SPB\_Data/facial\_recognition/trainer/trainer.yml')

# Print the numer of faces trained and end program

print("/n [INFO] {0} faces trained. Exiting Program".format(len(np.unique(ids))))

FRONT\_END RECOGNIZER:

from tkinter import \*

from PIL import ImageTk,Image

import recogniser

t = Tk(className="Facial\_Recognition")

canvas=Canvas(t,width=800,height=349)

image=ImageTk.PhotoImage(Image.open("C:\\Users\\Naithanis\\AppData\\Roaming\\SPB\_Data\\facial\_recognition\\background.png"))

canvas.create\_image(0,0,anchor=NW , image=image)

Label(t, text = 'FACE RECOGNITION [USER\_MODULE]',font='bold').place(x=260,y=20)

def check():

recogniser.recog()

Button(t,text="FACE RECOGNIZER",bg='black',fg='white',command=check).place(x=310,y=260,width=200,height=50)

canvas.pack()

t.mainloop()

RECOGNIZER:

def recog():

import cv2

import numpy as np

import os

import winsound

recognizer = cv2.face.LBPHFaceRecognizer\_create()

recognizer.read('C:/Users/Naithanis/AppData/Roaming/SPB\_Data/facial\_recognition/trainer/trainer.yml')

cascadePath = "C:/Users/Naithanis/AppData/Roaming/SPB\_Data/facial\_recognition/Cascades/haarcascade\_frontalface\_default.xml"

faceCascade = cv2.CascadeClassifier(cascadePath);

font = cv2.FONT\_HERSHEY\_SIMPLEX

#iniciate id counter

face\_id=0

names=[]

with open("user.txt") as f:

for line in f:

names.append(line.strip().split())

print(names)

print(face\_id)

# Initialize and start realtime video capture

cam = cv2.VideoCapture(0)

cam.set(3, 620) # set video widht

cam.set(4, 620) # set video height

# Define min window size to be recognized as a face

minW = 0.1\*cam.get(3)

minH = 0.1\*cam.get(4)

while True:

ret, img =cam.read()

img = cv2.flip(img, 1) # Flip vertically

gray = cv2.cvtColor(img,cv2.COLOR\_BGR2GRAY)

faces = faceCascade.detectMultiScale(

gray,

scaleFactor = 1.2,

minNeighbors = 5,

minSize = (int(minW), int(minH)),

)

for(x,y,w,h) in faces:

cv2.rectangle(img, (x,y), (x+w,y+h), (0,255,0), 2)

face\_id, confidence = recognizer.predict(gray[y:y+h,x:x+w])

# If confidence is less them 100 ==> "0" : perfect match

if (confidence < 100):

face\_id = names[face\_id]

confidence = " {0}%".format(round(100 - confidence))

winsound.MB\_OK

else:

face\_id = "unknown"

confidence = " {0}%".format(round(100 - confidence))

frequency=2500

duration=200 #1000ms = 1sec

winsound.Beep(frequency,duration)

cv2.putText(

img,

str(face\_id),

(x+5,y-5),

font,

1,

(255,255,255),

2

)

cv2.putText(

img,

str(confidence),

(x+5,y+h-5),

font,

1,

(255,255,0),

1

)

cv2.imshow('camera',img)

k = cv2.waitKey(10) & 0xff # Press 'ESC' for exiting video

if k == 27:

break

# Do a bit of cleanup

print("/n [INFO] Exiting Program and cleanup stuff")

cam.release()

cv2.destroyAllWindows()

DATABASE\_DISPLAY:

def disp():

from tkinter import Label,Text,END

import tkinter as tk

root=tk.Toplevel()

root.title("Database\_display")

names=[]

with open("user.txt") as f:

for line in f:

names.append(line.strip())

print(names)

length=len(names)

t = Text(root,fg="white",bg="black",font="bold")

Label(root,text="DATABASE",fg="white",bg="black",font="bold").place(x=330,y=10)

t.insert(END,'\n\n\n')

t.insert(END,'USER ID\t\t\t\t\tNAME\n')

i=0

while(i!=length):

for x in names:

if (i!=0) and (x!='none'):

t.insert(END,i)

t.insert(END,'\t\t\t\t\t')

t.insert(END, x)

t.insert(END,'\n')

i=i+1

t.pack()

root.mainloop()