import cv2

import face\_recognition

import pickle

import os

import firebase\_admin

from firebase\_admin import credentials

from firebase\_admin import db

from firebase\_admin import  storage

cred = credentials.Certificate("attendancerealtime-6f775-firebase-adminsdk-sns0g-a6ad6f7676.json")

firebase\_admin.initialize\_app(cred, {

    'databaseURL': "https://attendancerealtime-6f775-default-rtdb.firebaseio.com/",

    'storageBucket': "attendancerealtime-6f775.appspot.com"

})

# Importing student images

folderPath = 'Images'

pathList = os.listdir(folderPath)

print(pathList)

imgList = []

studentIds = []

for path in pathList:

    imgList.append(cv2.imread(os.path.join(folderPath, path)))

    studentIds.append(os.path.splitext(path)[0])

    fileName = f'{folderPath}/{path}'

    bucket = storage.bucket()

    blob = bucket.blob(fileName)

    blob.upload\_from\_filename(fileName)

    # print(path)

    # print(os.path.splitext(path)[0])

print(studentIds)

def findEncodings(imagesList):

    encodeList = []

    for img in imagesList:

        img = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

        encode = face\_recognition.face\_encodings(img)[0]

        encodeList.append(encode)

    return encodeList

print("Encoding Started ...")

encodeListKnown = findEncodings(imgList)

encodeListKnownWithIds = [encodeListKnown, studentIds]

print("Encoding Complete")

file = open("EncodeFile.p", 'wb')

pickle.dump(encodeListKnownWithIds, file)

file.close()

print("File Saved")

Giải thích encoder

khởi tạo kết nối với Firebase để lưu trữ dữ liệu, xử lý và tải hình ảnh sinh viên, tính toán mã hóa khuôn mặt cho những hình ảnh này và lưu các mã hóa này để sử dụng trong tương lai cho việc theo dõi điểm danh

Nhập Các Thư Viện Cần Thiết:

cv2: Thư viện OpenCV dùng cho các tác vụ thị giác máy tính, như xử lý hình ảnh.

face\_recognition: Thư viện cho các tác vụ nhận diện khuôn mặt.

pickle: Dùng để chuẩn hóa và khôi phục cấu trúc đối tượng Python.

os: Cung cấp các chức năng tương tác với hệ điều hành.

firebase\_admin và các lệnh nhập liên quan: Cho phép tương tác với Firebase, một nền tảng của Google phát triển để tạo ứng dụng di động và web. Ở đây, nó được sử dụng cho dịch vụ cơ sở dữ liệu và lưu trữ.

Khởi Tạo Firebase:

Mã khởi tạo Firebase sử dụng thông tin xác thực từ một tệp JSON. Điều này thiết lập kết nối với cơ sở dữ liệu thời gian thực và lưu trữ của Firebase.

Nhập Hình Ảnh Sinh Viên:

Mã liệt kê và đọc hình ảnh từ thư mục đã chỉ định (Images), giả sử đây là hình ảnh của sinh viên cho việc theo dõi điểm danh.

Nó cũng tải hình ảnh này lên lưu trữ Firebase, điều này có nghĩa là hệ thống duy trì một bản sao dựa trên đám mây của tất cả hình ảnh sinh viên.

Mã Hóa Khuôn Mặt:

Hàm findEncodings xử lý mỗi hình ảnh, chuyển nó sang định dạng RGB (vì OpenCV nạp hình ảnh theo định dạng BGR) và sau đó tính toán mã hóa khuôn mặt sử dụng thư viện face\_recognition.

Mã hóa khuôn mặt là cơ sở dữ liệu số biểu diễn đặc điểm khuôn mặt có thể được sử dụng cho nhận diện khuôn mặt.

Chuẩn Hóa và Lưu Mã Hóa:

Mã hóa khuôn mặt và ID sinh viên tương ứng được chuẩn hóa sử dụng pickle và lưu vào một tệp (EncodeFile.p). Tệp này có thể được sử dụng sau này để nhanh chóng tải mã hóa vào bộ nhớ cho các tác vụ nhận diện khuôn mặt, như trong quá trình điểm danh thực tế

import os

import pickle

import numpy as np

import cv2

import face\_recognition

import cvzone

import firebase\_admin

from firebase\_admin import credentials

from firebase\_admin import db

from firebase\_admin import storage

import numpy as np

from datetime import datetime

cred = credentials.Certificate("attendancerealtime-6f775-firebase-adminsdk-sns0g-a6ad6f7676.json")

firebase\_admin.initialize\_app(cred, {

    'databaseURL': "https://attendancerealtime-6f775-default-rtdb.firebaseio.com/",

    'storageBucket': "attendancerealtime-6f775.appspot.com"

})

bucket = storage.bucket()

cap = cv2.VideoCapture(0)

cap.set(3, 640)

cap.set(4, 480)

imgBackground = cv2.imread('Resources/background.png')

# Importing the mode images into a list

folderModePath = 'Resources/Modes'

modePathList = os.listdir(folderModePath)

imgModeList = []

for path in modePathList:

    imgModeList.append(cv2.imread(os.path.join(folderModePath, path)))

# print(len(imgModeList))

# Load the encoding file

print("Loading Encode File ...")

file = open('EncodeFile.p', 'rb')

encodeListKnownWithIds = pickle.load(file)

file.close()

encodeListKnown, studentIds = encodeListKnownWithIds

# print(studentIds)

print("Encode File Loaded")

modeType = 0

counter = 0

id = -1

imgStudent = []

displayTime = 0

while True:

    success, img = cap.read()

    imgS = cv2.resize(img, (0, 0), None, 0.25, 0.25)

    imgS = cv2.cvtColor(imgS, cv2.COLOR\_BGR2RGB)

    faceCurFrame = face\_recognition.face\_locations(imgS)

    encodeCurFrame = face\_recognition.face\_encodings(imgS, faceCurFrame)

    imgBackground[162:162 + 480, 55:55 + 640] = img

    imgBackground[44:44 + 633, 808:808 + 414] = imgModeList[modeType]

    if faceCurFrame and displayTime == 0:

        for encodeFace, faceLoc in zip(encodeCurFrame, faceCurFrame):

            matches = face\_recognition.compare\_faces(encodeListKnown, encodeFace)

            faceDis = face\_recognition.face\_distance(encodeListKnown, encodeFace)

            # print("matches", matches)

            # print("faceDis", faceDis)

            matchIndex = np.argmin(faceDis)

            # print("Match Index", matchIndex)

            if matches[matchIndex]:

                # print("Known Face Detected")

                # print(studentIds[matchIndex])

                y1, x2, y2, x1 = faceLoc

                y1, x2, y2, x1 = y1 \* 4, x2 \* 4, y2 \* 4, x1 \* 4

                bbox = 55 + x1, 162 + y1, x2 - x1, y2 - y1

                imgBackground = cvzone.cornerRect(imgBackground, bbox, rt=0)

                id = studentIds[matchIndex]

                if counter == 0:

                    cvzone.putTextRect(imgBackground, "Loading", (275, 400))

                    cv2.imshow("Face Attendance", imgBackground)

                    cv2.waitKey(1)

                    counter = 1

                    modeType = 1

                displayTime = 2

        if counter != 0:

            if counter == 1:

                # Get the Data

                studentInfo = db.reference(f'Students/{id}').get()

                print(studentInfo)

                # Get the Image from the storage

                blob = bucket.get\_blob(f'Images/{id}.png')

                if blob is not None:

                    array = np.frombuffer(blob.download\_as\_string(), np.uint8)

                    imgStudent = cv2.imdecode(array, cv2.COLOR\_BGRA2BGR)

                else:

                    print(f"File 'Images/{id}.png' not found in Firebase Storage.")

                    # Xử lý thêm ở đây nếu file không tồn tại

                array = np.frombuffer(blob.download\_as\_string(), np.uint8)

                imgStudent = cv2.imdecode(array, cv2.COLOR\_BGRA2BGR)

                # Update data of attendance

                datetimeObject = datetime.strptime(studentInfo['last\_attendance\_time'],

                                                   "%Y-%m-%d %H:%M:%S")

                secondsElapsed = (datetime.now() - datetimeObject).total\_seconds()

                print(secondsElapsed)

                if secondsElapsed > 30:

                    ref = db.reference(f'Students/{id}')

                    studentInfo['total\_attendance'] += 1

                    ref.child('total\_attendance').set(studentInfo['total\_attendance'])

                    ref.child('last\_attendance\_time').set(datetime.now().strftime("%Y-%m-%d %H:%M:%S"))

                else:

                    modeType = 3

                    counter = 0

                    imgBackground[44:44 + 633, 808:808 + 414] = imgModeList[modeType]

            if modeType != 3:

                if 10 < counter < 20:

                    modeType = 2

                imgBackground[44:44 + 633, 808:808 + 414] = imgModeList[modeType]

                if counter <= 10:

                    cv2.putText(imgBackground, str(studentInfo['total\_attendance']), (861, 125),

                                cv2.FONT\_HERSHEY\_COMPLEX, 1, (255, 255, 255), 1)

                    cv2.putText(imgBackground, str(studentInfo['major']), (1006, 550),

                                cv2.FONT\_HERSHEY\_COMPLEX, 0.5, (255, 255, 255), 1)

                    cv2.putText(imgBackground, str(id), (1006, 493),

                                cv2.FONT\_HERSHEY\_COMPLEX, 0.5, (255, 255, 255), 1)

                    cv2.putText(imgBackground, str(studentInfo['standing']), (910, 625),

                                cv2.FONT\_HERSHEY\_COMPLEX, 0.6, (100, 100, 100), 1)

                    cv2.putText(imgBackground, str(studentInfo['year']), (1025, 625),

                                cv2.FONT\_HERSHEY\_COMPLEX, 0.6, (100, 100, 100), 1)

                    cv2.putText(imgBackground, str(studentInfo['starting\_year']), (1125, 625),

                                cv2.FONT\_HERSHEY\_COMPLEX, 0.6, (100, 100, 100), 1)

                    (w, h), \_ = cv2.getTextSize(studentInfo['name'], cv2.FONT\_HERSHEY\_COMPLEX, 1, 1)

                    offset = (414 - w) // 2

                    cv2.putText(imgBackground, str(studentInfo['name']), (808 + offset, 445),

                                cv2.FONT\_HERSHEY\_COMPLEX, 1, (50, 50, 50), 1)

                    if imgStudent is not None and imgStudent.size != 0:

                        imgStudentResized = cv2.resize(imgStudent, (216, 216))  # Resize thành (216, 216)

                        imgBackground[175:175 + 216, 909:909 + 216] = imgStudentResized

                counter += 1

                if counter >= 20:

                    counter = 0

                    modeType = 0

                    studentInfo = []

                    imgStudent = []

                    imgBackground[44:44 + 633, 808:808 + 414] = imgModeList[modeType]

    else:

        modeType = 0

        counter = 0

    # cv2.imshow("Webcam", img)

    if displayTime > 0:

        cv2.imshow("Face Attendance", imgBackground)

        cv2.waitKey(1000)  # Thời gian chờ 1 giây

        displayTime -= 1  # Giảm thời gian hiển thị

        continue  # Quay lại đầu vòng lặp

    cv2.imshow("Face Attendance", imgBackground)

    cv2.waitKey(4)

/// giải thích file face detection

Khởi Tạo và Kết Nối Firebase: Dùng để kết nối với cơ sở dữ liệu và lưu trữ trên Firebase, giúp quản lý dữ liệu sinh viên và hình ảnh.

Khởi Tạo Camera và Đọc Hình Ảnh: Sử dụng camera máy tính để chụp ảnh và xử lý hình ảnh trực tiếp.

Tải và Xử Lý Dữ Liệu Nhận Diện Khuôn Mặt: Đọc file chứa dữ liệu đã mã hóa của khuôn mặt sinh viên và tải thông tin này vào bộ nhớ.

Nhận Diện Khuôn Mặt và Xác Định Danh Tính: Phân tích hình ảnh từ camera để nhận diện khuôn mặt và so sánh với cơ sở dữ liệu khuôn mặt đã mã hóa để xác định danh tính của sinh viên.

Hiển Thị Thông Tin và Cập Nhật Dữ Liệu Điểm Danh: Sau khi nhận diện khuôn mặt thành công, hệ thống sẽ hiển thị thông tin của sinh viên trên màn hình và cập nhật dữ liệu điểm danh lên Firebase, bao gồm thời gian điểm danh cuối cùng và tổng số lần điểm danh.

Lưu Trữ và Hiển Thị Hình Ảnh Sinh Viên: Lấy hình ảnh của sinh viên từ Firebase và hiển thị trên màn hình.

Quản Lý Trạng Thái và Hiển Thị Giao Diện: Quản lý trạng thái hiện tại của hệ thống và cập nhật giao diện người dùng tương ứng.

57-87 xử lý sự kiện hiển thị trên giaodiện dùng các ảnh trong modes

89-167 hiển thị thông tin, firebase, lấy ảnh đưa lên fire

import firebase\_admin

from firebase\_admin import credentials

from firebase\_admin import db

cred = credentials.Certificate("attendancerealtime-6f775-firebase-adminsdk-sns0g-a6ad6f7676.json")

firebase\_admin.initialize\_app(cred, {

    'databaseURL': "https://attendancerealtime-6f775-default-rtdb.firebaseio.com/"

})

ref = db.reference('Students')

data = {

    "321654":

        {

            "name": "Murtaza Hassan",

            "major": "Robotics",

            "starting\_year": 2017,

            "total\_attendance": 7,

            "standing": "G",

            "year": 4,

            "last\_attendance\_time": "2022-12-11 00:54:34"

        },

    "852741":

        {

            "name": "Emly Blunt",

            "major": "Economics",

            "starting\_year": 2021,

            "total\_attendance": 12,

            "standing": "B",

            "year": 1,

            "last\_attendance\_time": "2022-12-11 00:54:34"

        },

    "12345":

        {

            "name": "Ronaldo",

            "major": "Football Player",

            "starting\_year": 2021,

            "total\_attendance": 12,

            "standing": "B",

            "year": 1,

            "last\_attendance\_time": "2022-12-11 00:54:34"

        },

    "963852":

        {

            "name": "Elon Musk",

            "major": "Physics",

            "starting\_year": 2020,

            "total\_attendance": 7,

            "standing": "G",

            "year": 2,

            "last\_attendance\_time": "2022-12-11 00:54:34"

        }

}

for key, value in data.items():

    ref.child(key).set(value)

Code này chả có gì cả  
nó là tạo dữ liệu firebase

Ví dụ trong 12345 thì sẽ có thông tin của ronaldo  
tương tự vậy

Lưu ý khi đưa lên firebase

Tạo project mới

Chọn tạo bản thử nghiệm

Sau đó vào real time database lấy link real time ( không cần phải thay đổi bất kỳ thứ gì trên realtime )

Sau đó vào setting project => service account chọn python sau đó chọn generator json

Lấy json về đưa cùng 1 folder nhận diện khuôn mặt

Thay vào code

Vào kho lấy link ( xóa trước :// lấy mỗi dòng chữ ) như trong code

Sau đó thay vào code

Chạy file generator để cập nhật hình ảnh

Sau đó chạy file add database để đưa dữ liệu lênrealtime

Chạy face nhận diện

Nếu muốn hình ảnh sau khi nhận diện được delay

Thì sửa trong waitkey(4) chẳng hạn

Nó sẽ delay 4 giây  
ok