MAIN SOURCE CODE

Title : Face Recognition for Access to Public Services

Name : Ukhem Fahmi Thoriqul Haq

Import Library

In [ ]:
```python
from datetime import datetime
import requests
import face_recognition
import cv2
import numpy as np
import pandas as pd
import winsound

from googleapiclient import discovery
from google.oauth2 import service_account
```

Membuat Constant Variable

Untuk mengakses Google sheets bisa langsung menggunakan Google API, atau bisa dengan API pihak ketiga seperti Sheety API

In [ ]:
```python
# ------------------------------------------ CONSTANT GOOGLE-API ----------------------------------------- #
SCOPES = ['https://www.googleapis.com/auth/spreadsheets']
SERVICE_ACCOUNT_FILE = 'ekc_elektro_itk.json'
CREDENTIALS = service_account.Credentials.from_service_account_file(
    filename=SERVICE_ACCOUNT_FILE, scopes=SCOPES)

SERVICE = discovery.build('sheets', 'v4', credentials=CREDENTIALS)
SPREADSHEET_ID = "1c6uz6mBi5DX0KDEaNtDF-GNmeckyGyP_QGE55RP2fYA"
RANGE_POST = "Record!A1:C1"
INPUT_USER = "USER_ENTERED"

# ------------------------------------------ CONSTANT SHEETY-API ----------------------------------------- #
SHEETY_ENDPOINT = "https://api.sheety.co/d71c9673#Silahkan-buat-sendiri/System/myfile"
#link ini tidak bisa kalian pakai

BEARER_AUTHENTICATION = {
    "Authorization": "Bearer fix_dapat_a"
}
```

Membuat Function

In [ ]:
```python
# -------------------------------------------- DEFINISI FUNGSI ------------------------------------------- #
FONT = cv2.FONT_HERSHEY_SIMPLEX

def waktu_deteksi():
    """Memberikan waktu saat pendeteksian ini terjadi."""
    global tanggal_sekarang, jam_sekarang
    date = datetime.today()
    tanggal_sekarang = date.strftime("%d/%m/%Y")
    jam_sekarang = date.strftime("%H:%M:%S")

def save_to_database_by_sheety():
    """Menyimpan data orang yang telah terdeteksi."""
    attendance_parameter = {
        "ekc": {
            "nama": nama_sekarang,
            "jam": jam_sekarang,
            "tanggal": tanggal_sekarang,
        }
    }

    requests.post(url=SHEETY_ENDPOINT, json=attendance_parameter, headers=BEARER_AUTHENTICATION)
    print(nama_sekarang, jam_sekarang, tanggal_sekarang)

def confirm_bell():
    """Memberikan sebuah bunyi sebagai bentuk konfirmasi data telah tercatat oleh sistem."""
    bel = winsound
    bel.Beep(frequency, duration)


def save_to_database_by_google_API():
    """Menyimpan data orang yang telah terdeteksi ke GOOGLE SHEETS dengan Google API"""
    data = [[nama_sekarang, jam_sekarang, tanggal_sekarang]]
    request = SERVICE.spreadsheets().values().append(spreadsheetId=SPREADSHEET_ID,
                                                     range=RANGE_POST,
                                                     valueInputOption=INPUT_USER,
                                                     insertDataOption="INSERT_ROWS",
                                                     body={"values": data})
    response = request.execute()
    # print(response)
```

Main Progam

In [ ]:
```python
# ----------------------------------------------- VARIABLE ----------------------------------------------- #
df_face_encode = pd.read_csv("data/elektro_face_encodings.csv", delimiter=',', header=None)
known_face_encodings = df_face_encode.to_numpy()

data_name = open("data/elektro-nama.csv", "r")
df_name = data_name.read()
df_name = df_name.split("\n")

# Initialize some variables
face_locations = []
face_encodings = []
face_names = []
process_this_frame = True

nama_sebelum = ""
unknown = 0

frequency = 800 # Hz
duration = 200  # millisecond

list_nama_berurut = [""]


# --------------------------------------------- MAIN FUNCTION -------------------------------------------- #
# Get a reference to webcam #0 (the default one)
video_capture = cv2.VideoCapture(1)
while True:
    # Grab a single frame of video
    waktu = datetime.now().replace(microsecond=0)
    waktu3 = waktu.strftime(f"%A  %d/%m/%Y  %H:%M:%S")

    ret, frame = video_capture.read()


    # Only process every other frame of video to save time
    if process_this_frame:
        # Resize frame of video to 1/4 size for faster face recognition processing
        small_frame = cv2.resize(frame, (0, 0), fx=0.25, fy=0.25)

        # Convert the image from BGR color (which OpenCV uses) to RGB color (which face_recognition uses)
        rgb_small_frame = small_frame[:, :, ::-1]

        # Find all the faces and face encodings in the current frame of video
        face_locations = face_recognition.face_locations(rgb_small_frame)

        face_encodings = face_recognition.face_encodings(rgb_small_frame, face_locations)

        face_names = []

        # ------------------------------------ MENCARI WAJAH YANG COCOK ------------------------------------ #
        for face_encoding in face_encodings:
            # See if the face is a match for the known face(s)
            matches = face_recognition.compare_faces(known_face_encodings, face_encoding)
            nama_sekarang = "Unknown"

            # Use the known face with the smallest distance to the new face
            face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
            best_match_index = np.argmin(face_distances)
            if matches[best_match_index]:
                nama_sekarang = df_name[best_match_index].replace('"', '')

                if nama_sekarang == list_nama_berurut[0]:
                    list_nama_berurut.append(nama_sekarang)
                else:
                    list_nama_berurut = [nama_sekarang]

                nama = list_nama_berurut[0]

                jumlah_consecutive = list_nama_berurut.count(nama)
                try:
                    if jumlah_consecutive >= 10:
                        list_nama_berurut = [""]

                        if nama == nama_sebelum:
                            pass
                        else:
                            nama_sebelum = nama
                            # --------------------- SAVE DATA TERDETEKSI TO GOOGLE SHEET ------------------ #
                            waktu_deteksi()
                            # save_to_database_by_sheety()
                            save_to_database_by_google_API()
                            # --------------------------------- BEL CONFIRMS ------------------------------ #
                            confirm_bell()
                    else:
                        pass
                except:
                    pass

            face_names.append(nama_sekarang)

    process_this_frame = not process_this_frame

    # ------------------------------------------ DISPLAY DINAMIS ------------------------------------------ #
    # Display the results
    for (top, right, bottom, left), nama_sekarang in zip(face_locations, face_names):
        # Scale back up face locations since the frame we detected in was scaled to 1/4 size
        top *= 4
        right *= 4
        bottom *= 4
        left *= 4

        # Draw a box around the face
        cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)

        # Draw a label with a name below the face
        cv2.rectangle(frame, (left, bottom - 30), (right, bottom), (0, 0, 255), cv2.FILLED)
        cv2.putText(frame, nama_sekarang.split()[0], (left + 6, bottom - 8), FONT, 0.65, (255, 255, 255), 1)


    # ------------------------------------------ DISPLAY STATIS ------------------------------------------- #

    # Setup status box
    cv2.rectangle(frame, (0, 0), (640, 40), (0, 0, 0), -1) #(245, 117, 16)

    cv2.putText(frame, (waktu3 + "    |    EKC - ELEKTRO - ITK"), (18, 27),
                FONT, 0.58, (255, 255, 255), 1, cv2.LINE_AA)

    # Display the resulting image
    cv2.imshow('SMART GOVERNANCE - Face Recognition For Access to Public Service', frame)

    # Hit 'q' on the keyboard to quit!
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# Release handle to the webcam
video_capture.release()
cv2.destroyAllWindows()
```