import cv2

import numpy as np

import streamlit as st

import face\_recognition

import base64

* Here, we're importing the necessary libraries:
  + cv2: OpenCV, a library mainly used for image processing tasks like reading, writing, and manipulating images.
  + numpy: A fundamental package for numerical computing in Python. It's used for array processing and performing mathematical operations on arrays.
  + streamlit: A Python library for creating web applications with minimal effort. It's particularly useful for creating interactive data dashboards.
  + face\_recognition: A face detection and recognition library built using dlib and OpenCV.
  + base64: A module providing functions to encode binary data into ASCII characters using base64 encoding.

python

def image\_to\_base64(image):

if image is None:

return None

\_, buffer = cv2.imencode('.png', image)

if buffer is None:

return None

return base64.b64encode(buffer).decode()

* This function image\_to\_base64 takes an image as input and converts it to a base64 encoded string.
* It first checks if the image is None. If so, it returns None.
* Then, it uses cv2.imencode to encode the image as a PNG format.
* If the buffer obtained from encoding is None, it returns None.
* Finally, it encodes the buffer in base64 and returns the decoded string.

python

def capture\_image():

cap = cv2.VideoCapture(0)

if not cap.isOpened():

st.error("Error: Unable to open camera.")

return None

ret, frame = cap.read()

if not ret:

st.error("Error: Unable to capture frame from camera.")

return None

frame\_rgb = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

cap.release()

return frame\_rgb

* This capture\_image function captures an image from the camera.
* It starts by opening the default camera (index 0) using cv2.VideoCapture(0).
* If the camera fails to open, it displays an error message using Streamlit's st.error and returns None.
* It reads a frame from the camera using cap.read().
* If it fails to read a frame, it displays an error message and returns None.
* The captured frame is then converted from BGR (default format for OpenCV) to RGB format.
* Finally, it releases the camera resource using cap.release() and returns the captured frame in RGB format.

python

def match\_faces(image1, image2):

image1\_encoding = face\_recognition.face\_encodings(image1)

image2\_encoding = face\_recognition.face\_encodings(image2)

if len(image1\_encoding) == 0 or len(image2\_encoding) == 0:

return False, 0

image1\_encoding = image1\_encoding[0]

image2\_encoding = image2\_encoding[0]

results = face\_recognition.compare\_faces([image1\_encoding], image2\_encoding)

match\_percentage = face\_recognition.face\_distance([image1\_encoding], image2\_encoding)

return results[0], (1 - match\_percentage[0]) \* 100

* This function match\_faces compares faces between two images.
* It first encodes the faces in both images using face\_recognition.face\_encodings.
* If either of the images doesn't contain a face (i.e., if the length of either encoding list is 0), it returns False for match and 0 for match percentage.
* It then compares the face encodings of the two images using face\_recognition.compare\_faces.
* Additionally, it calculates the match percentage using face\_recognition.face\_distance.
* Finally, it returns a boolean indicating whether the faces match and the match percentage.

python

def main():

st.title("Face Matching Web App")

session\_state = st.session\_state.setdefault('state', {

'image1': None,

'image2': None,

'capture\_first\_image': False,

'capture\_second\_image': False,

'match\_button\_pressed': False,

})

* This is the main function, the entry point of the Streamlit web app.
* It sets the title of the app to "Face Matching Web App" using st.title.
* It initializes the session state using Streamlit's st.session\_state. Session state allows maintaining the state of the application across different sessions.Top of FormBottom of Form  
    
  Before running the project, open cmd and run as administrator and paste the code below:
* >>> pip install opencv-python-headless  
  then second code :
* >>> pip install opencv-python-headless numpy streamlit face\_recognition  
    
  >>> Save the python file, save the file with app.py   
  >>> open cmd where the file is saved and in cmd type:

streamlit run app.py