import cv2

import os

from flask import Flask,request,render\_template

from datetime import date

from datetime import datetime

import numpy as np

from sklearn.neighbors import KNeighborsClassifier

import pandas as pd

import joblib

#### Defining Flask App

app = Flask(\_\_name\_\_)

#### Saving Date today in 2 different formats

datetoday = date.today().strftime("%m\_%d\_%y")

datetoday2 = date.today().strftime("%d-%B-%Y")

#### Initializing VideoCapture object to access WebCam

face\_detector = cv2.CascadeClassifier('static/haarcascade\_frontalface\_default.xml')

cap = cv2.VideoCapture(0)

#### If these directories don't exist, create them

if not os.path.isdir('Attendance'):

os.makedirs('Attendance')

if not os.path.isdir('static/faces'):

os.makedirs('static/faces')

if f'Attendance-{datetoday}.csv' not in os.listdir('Attendance'):

with open(f'Attendance/Attendance-{datetoday}.csv','w') as f:

f.write('Name,Roll,Time')

#### get a number of total registered users

def totalreg():

return len(os.listdir('static/faces'))

#### extract the face from an image

def extract\_faces(img):

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

face\_points = face\_detector.detectMultiScale(gray, 1.3, 5)

return face\_points

#### Identify face using ML model

def identify\_face(facearray):

model = joblib.load('static/face\_recognition\_model.pkl')

return model.predict(facearray)

#### A function which trains the model on all the faces available in faces folder

def train\_model():

faces = []

labels = []

userlist = os.listdir('static/faces')

for user in userlist:

for imgname in os.listdir(f'static/faces/{user}'):

img = cv2.imread(f'static/faces/{user}/{imgname}')

resized\_face = cv2.resize(img, (50, 50))

faces.append(resized\_face.ravel())

labels.append(user)

faces = np.array(faces)

knn = KNeighborsClassifier(n\_neighbors=5)

knn.fit(faces,labels)

joblib.dump(knn,'static/face\_recognition\_model.pkl')

#### Extract info from today's attendance file in attendance folder

def extract\_attendance():

df = pd.read\_csv(f'Attendance/Attendance-{datetoday}.csv')

names = df['Name']

rolls = df['Roll']

times = df['Time']

l = len(df)

return names,rolls,times,l

#### Add Attendance of a specific user

def add\_attendance(name):

username = name.split('\_')[0]

userid = name.split('\_')[1]

current\_time = datetime.now().strftime("%H:%M:%S")

df = pd.read\_csv(f'Attendance/Attendance-{datetoday}.csv')

if int(userid) not in list(df['Roll']):

with open(f'Attendance/Attendance-{datetoday}.csv','a') as f:

f.write(f'\n{username},{userid},{current\_time}')

################## ROUTING FUNCTIONS #########################

#### Our main page

@app.route('/')

def home():

names,rolls,times,l = extract\_attendance()

Return render\_template('home.html',names=names,rolls=rolls,times=times,l=l,totalreg=totalreg(),datetoday2=datetoday2)

#### This function will run when we click on Take Attendance Button

@app.route('/start',methods=['GET'])

def start():

if 'face\_recognition\_model.pkl' not in os.listdir('static'):

return render\_template('home.html',totalreg=totalreg(),datetoday2=datetoday2,mess='There is no trained model in the static folder. Please add a new face to continue.')

cap = cv2.VideoCapture(0)

ret = True

while ret:

ret,frame = cap.read()

if extract\_faces(frame)!=():

(x,y,w,h) = extract\_faces(frame)[0]

cv2.rectangle(frame,(x, y), (x+w, y+h), (255, 0, 20), 2)

face = cv2.resize(frame[y:y+h,x:x+w], (50, 50))

identified\_person = identify\_face(face.reshape(1,-1))[0]

add\_attendance(identified\_person)

cv2.putText(frame,f'{identified\_person}',(30,30),cv2.FONT\_HERSHEY\_SIMPLEX,1,(255, 0, 20),2,cv2.LINE\_AA)

cv2.imshow('Attendance',frame)

if cv2.waitKey(1)==27:

Break

cap.release()

cv2.destroyAllWindows()

names,rolls,times,l = extract\_attendance()

return render\_template('home.html',names=names,rolls=rolls,times=times,l=l,totalreg=totalreg(),datetoday2=datetoday2)

#### This function will run when we add a new user

@app.route('/add',methods=['GET','POST'])

def add():

newusername = request.form['newusername']

newuserid = request.form['newuserid']

userimagefolder = 'static/faces/'+newusername+'\_'+str(newuserid)

if not os.path.isdir(userimagefolder):

os.makedirs(userimagefolder)

cap = cv2.VideoCapture(0)

i,j = 0,0

while 1:

\_,frame = cap.read()

faces = extract\_faces(frame)

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

cv2.rectangle(frame,(x, y), (x+w, y+h), (255, 0, 20), 2)

cv2.putText(frame,Images Captured: {i}/50',(30,30),cv2.FONT\_HERSHEY\_SIMPLEX,1,(255, 0, 20),2,cv2.LINE\_AA)

if j%10==0:

name = newusername+'\_'+str(i)+'.jpg'

cv2.imwrite(userimagefolder+'/'+name,frame[y:y+h,x:x+w])

i+=1

j+=1

if j==500:

Break

cv2.imshow('Adding new User',frame)

if cv2.waitKey(1)==27:

Break

cap.release()

cv2.destroyAllWindows()

print('Training Model')

train\_model()

names,rolls,times,l = extract\_attendance()

return render\_template('home.html',names=names,rolls=rolls,times=times,l=l,totalreg=totalreg(),datetoday2=datetoday2)

#### Our main function which runs the Flask App

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)