APPLICATION BUILDING

Python Code

Date	17 November 2022
Team ID	PNT2022TMID49453
Project Name	VirtualEye - Life Guard For Swimming Pools To Detect Active Drowning
Maximum Marks	8 Marks

App.py:

```
# import necessary
        packages
import cvlib as cv
from cvlib.object_detection
        import draw_bbox
# import necessary
        packages
from flask import Flask,
        render_template,
        request,redirect,url_f
        or
import requests
import os
from sys import exit
import cvlib as cv
import cv2
import time
import numpy as np
import math
import argparse
import playsound
from cloudant.client import
        Cloudant
client = Cloudant.iam(
    '07c5a12f-84fd-49c6-
        bbfa-de80bd989d12-
        bluemix','Rnz_zCc7h
        N5Lb5uRHaxn-
        WrlN9yqbtz4QKlFVZ
        4ETZpk',connect=Tr
        ue)
_id= 'name'
name= 'a@b.c'
psw = '123'
my_database =
        client.create_databa
        se('my_database')

app = Flask(__name__)
```

```
@app.route('/')
def index():
    return
            render_template('ind
            ex.html')

@app.route('/login')
def login():  # put
            application's code
            here
    return
            render_template('logi
            n.html')


@app.route('/register')
def register():
    return
            render_template('regi
            ster.html')

@app.route('/home')
def home():
    return
            render_template('ind
            ex1.html')
@app.route('/afterlogin',met
            hods=['POST'])
def afterlogin():
    user= request.form['_id']
    passw=
            request.form['psw']
    print(user,passw)
    query = {'_id':{'$eq':user}}
    docs =
            my_database.get_qu
            ery_result(query)
    print(docs)
    print(len(docs.all()))
    if(len(docs.all())==0):
        return
            render_template('logi
            n.html', pred='' the
            username is not
            found.'')
    else:

            if((user==docs[0][0]['
            _id'] and
            passw==docs[0][0]['p
            sw'])):
             return
            redirect(url_for('/step
            2'))
```

```
        else:
            print('Invalid User')

@app.route('/afterreg',
           methods=['POST'])
def afterreg():
    x = [x for x in
           request.form.values()
           ]
    print(x)
    data={
    '_id':x[1],
    'name':x[0],
    'psw':x[2],
    }
    print(data)

    query = {'_id': {'$eq':
           data['_id']}}

    docs =
           my_database.get_qu
           ery_result(query)
    print(docs)

    print(len(docs.all()))
    if(len(docs.all())==0):

           url=my_database.cre
           ate_document(data)
        return
           render_template('regi
           ster.html',
           pred="Registration
           Successful,please
           login using your
           details")
    else:
        return
           render_template('regi
           ster.html',
           pred="Registration
           Successful,please
           login using your
           details")


@app.route('/step2')
def step2():

    print("Begin")

    webcam =
           cv2.VideoCapture("g
           arden.mp4")
```

```
    padding = 20

    if not
        webcam.isOpened():
       print("Could not open
        webcam")
       exit()


    t0 = time.time() #gives
        time in seconds after
        1970
    #print('t0=',t0)
#variable dcount stands for
        how many seconds
        the person has been
        standing still for
    centre0 = np.zeros(2)
    isDrowning = False

#this loop happens
        approximately every
        1 second, so if a
        person doesn't
        move,
#or moves very little for
        10seconds, we can
        say they are
        drowning

# loop through frames
    while
        webcam.isOpened():

    # read frame from
        webcam
       status, frame =
        webcam.read()

       if not status:
        break
    #small_frame =
        cv2.resize(frame,(0,0
        ),fx = 0.5,fy = 0.5)
    # apply object detection
       bbox, label, conf =
        cv.detect_common_
        objects(frame,
        confidence=0.25,
        model='yolov3-tiny')

       print(bbox, label, conf)


       if(len(bbox)>0):
```

```
   bbox0 = bbox[0]
#centre = np.zeros(s)
   centre = [0,0]


#for i in range(0,
   len(bbox)):
    #centre[i]
   =[(bbox[i][0]+bbox[i][
   2])/2,(bbox[i][1]+bbox
   [i][3])/2 ]

   centre
   =[(bbox0[0]+bbox0[2]
   )/2,(bbox0[1]+bbox0[
   3])/2 ]

#make vertical and
   horizontal movement
   variables
   hmov =
   abs(centre[0]-
   centre0[0])
   vmov =
   abs(centre[1]-
   centre0[1])

#there is still need to
   tweek the threshold
#this threshold is for
   checking how much
   the centre has
   moved

   x=time.time()

   threshold = 10

   #print("hmov=",hmov
   )
   if(hmov>threshold or
   vmov>threshold):
       print(x-t0, 'sif')
       t0 = time.time()
       isDrowning =
   False

   else:
       print(x-t0, 'selse')
       if((time.time() - t0)
   > 10):
           isDrowning =
   True

   print('bbox: ', bbox,
```

```
            'centre:', centre,
            'centre0:', centre0)
            print('Is he/she
            drowning: ',
            isDrowning)
                    #print('End of
            the program')


            centre0 = centre
            # draw bounding box
            over detected
            objects
            # draw bounding box
            over detected
            objects
        out =
            draw_bbox(frame,
            bbox, label, conf,
            write_conf=True)
            # display output
        cv2.imshow("Real-time
            object detection",
            out)
        if(isDrowning == True):
            webcam.release()

            cv2.destroyAllWindo
            ws()
            return
            render_template('ind
            ex1.html',
            prediction_text = "1")
    # press "Q" to stop
        if cv2.waitKey(1) &
            0xFF == ord('q'):
            break

# release resources
    webcam.release()
    cv2.destroyAllWindows()
if __name__ == '__main__':
            app.run(debug=True)
```