Final Deliverables: Source Code

Date	18 November 2022
Team ID	PNT2022TMID53114
Project Name	VirtualEye - Lifeguard for swimming pools to detect active drowning

App.py

```
from cloudant.client import Cloudant
import numpy as np
import os
from flask import Flask, app,request,render_template
from tensorflow.keras import models
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from tensorflow.python.ops.gen_array_ops import concat
from tensorflow.keras.applications.inception_v3 import preprocess_input
import cvlib as cv
from cvlib.object_detection import draw_bbox
import cv2
import time
import numpy as np
from playsound import playsound
#import requests
from flask import Flask, request, render_template, redirect, url_for
#Loading the model


# Authenticate using an IAM API key
client =
Cloudant.iam('06e7c9cd-cbb3-4b56-a40a-e669cf5b0906-bluemix','VPbZAA_fmWRYpJdz4kowa
ZwERWNd4vqCSvOzVI5DXmNn', connect=True)

# Create a database using an initialized client
my_database = client['database1']


app = Flask(_name_)

@app.route("/")
def index():
    return render_template("./login.html")

@app.route("/about")
def about():
    return render_template("./about.html")

@app.route("/demo")
def demo():
    return render_template("./demo.html")

@app.route("/logout")
def logout():
    return render_template("./logout.html")

@app.route("/register")
def register():
    return render_template("./register.html")

@app.route("/result")
def res():
    webcam = cv2.VideoCapture('drowning7.mp4')
    if not webcam.isOpened():
        print("Could not open webcam")
        exit()

    t0 = time.time() #gives time in seconds after 1970
     #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




    t0 = time.time() #gives time in seconds after 1970

    #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:
        print("Could not read frame")
        exit()

    # apply object detection
    bbox, label, conf = cv.detect_common_objects(frame)
    #simplifying for only 1 person

    #s = (len(bbox), 2)
    print(bbox)
    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 = 30
            if(hmov>threshold or vmov>threshold):
                print(x-t0, 's')
                t0 = time.time()
                isDrowning = False

            else:
                    print(x-t0, 's')
                    if((time.time() - t0) > 5):
                        isDrowning = True




                #print('bounding box: ', bbox, 'label: ' label ,'confidence: ' conf[0], 'centre: ', centre)
                #print(bbox,label ,conf, centre)
                print('bbox: ', bbox, 'centre:', centre, 'centre0:', centre0)
                print('Is he drowning: ', isDrowning)

                centre0 = centre
                # draw bounding box over detected objects

        out = draw_bbox(frame, bbox, label, conf,isDrowning)

        #print('Seconds since last epoch: ', time.time()-t0)

        # display output
        cv2.imshow("Real-time object detection", out)
        print(isDrowning)
        if(isDrowning == True):

            playsound('alarm.mp3')

        # press "Q" to stop
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    # release resources
    webcam.release()
    cv2.destroyAllWindows()




@app.route('/afterreg', methods=['GET'])
def afterreg():
    username = request.args.get('uname')
    password = request.args.get('password')
    print(list(request.form.values()))
    data = {
```

```
    'uname': username,
    'password': password
    }
    print(data)
    query = {'uname': {'$eq': data['uname']}}
    docs = my_database.get_query_result(query)
    print(docs)

    print(len(docs.all()))

    if(len(docs.all())==0):
        url = my_database.create_document(data)
        #response = requests.get(url)
        return render_template('login.html', pred="Registration Successful, please login using your
details")
    else:
        return render_template('login.html', pred="You are already a member, please login using
your details")


@app.route('/afterlogin',methods=['GET'])
def afterlogin():
    user = request.args.get('uname')
    passw = request.args.get('password')
    print(user + passw)
    query = {'uname': {'$eq': user}}
    docs = my_database.get_query_result(query)
    print(docs)
    print(len(docs.all()))
    if(len(docs.all())==0):
        return render_template('login.html', pred="The username is not found.")
    else:
        if((user==docs[0][0]['uname'] and passw==docs[0][0]['password'])):
            return render_template('about.html')
        else:
            return render_template('login.html', pred="incorrect password, please try again.")




if _name_ == '_main_':
   app.run()
```

Object_detection.py

```
#import necessary packages
import cv2
import os
import numpy as np
from .utils import download_file

initialize = True
net = None
dest_dir = os.path.expanduser('~') + os.path.sep + '.cvlib' + os.path.sep + 'object_detection' +
os.path.sep + 'yolo' + os.path.sep + 'yolov3'
classes = None
#colors are BGR instead of RGB in python
COLORS = [0,0,255], [255,0,0]

def populate_class_labels():

    #we are using a pre existent classifier which is more reliable and more efficient than one
    #we could make using only a laptop
    #The classifier should be downloaded automatically when you run this script
    class_file_name = 'yolov3_classes.txt'
    class_file_abs_path = dest_dir + os.path.sep + class_file_name
    url = 'https://github.com/Nico31415/Drowning-Detector/raw/master/yolov3.txt'
    if not os.path.exists(class_file_abs_path):
        download_file(url=url, file_name=class_file_name, dest_dir=dest_dir)
    f = open(class_file_abs_path, 'r')
    classes = [line.strip() for line in f.readlines()]

    return classes


def get_output_layers(net):

    #the number of output layers in a neural network is the number of possible
    #things the network can detect, such as a person, a dog, a tie, a phone...
    layer_names = net.getLayerNames()

    output_layers = [layer_names[i - 1] for i in net.getUnconnectedOutLayers()]

    return output_layers


def draw_bbox(img, bbox, labels, confidence, Drowning, write_conf=False):
    global COLORS
    global classes

    if classes is None:
        classes = populate_class_labels()

    for i, label in enumerate(labels):

        #if the person is drowning, the box will be drawn red instead of blue
        if label == 'person' and Drowning:
            color = COLORS[0]
            label = 'DROWNING'
        else:
            color = COLORS[1]

        if write_conf:
            label += ' ' + str(format(confidence[i] * 100, '.2f')) + '%'


        #you only need to points (the opposite corners) to draw a rectangle. These points
        #are stored in the variable bbox
        cv2.rectangle(img, (bbox[i][0],bbox[i][1]), (bbox[i][2],bbox[i][3]), color, 2)

        cv2.putText(img, label, (bbox[i][0],bbox[i][1]-10), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
color, 2)

    return img

def detect_common_objects(image, confidence=0.5, nms_thresh=0.3):

    Height, Width = image.shape[:2]
    scale = 0.00392

    global classes
    global dest_dir

    #all the weights and the neural network algorithm are already preconfigured
    #as we are using YOLO

    #this part of the script just downloads the YOLO files
    config_file_name = 'yolov3.cfg'
    config_file_abs_path = dest_dir + os.path.sep + config_file_name
```

```
weights_file_name = 'yolov3.weights'
weights_file_abs_path = dest_dir + os.path.sep + weights_file_name

url = 'https://github.com/Nico31415/Drowning-Detector/raw/master/yolov3.cfg'

if not os.path.exists(config_file_abs_path):
    download_file(url=url, file_name=config_file_name, dest_dir=dest_dir)

url = 'https://pjreddie.com/media/files/yolov3.weights'

if not os.path.exists(weights_file_abs_path):
    download_file(url=url, file_name=weights_file_name, dest_dir=dest_dir)



global initialize
global net

if initialize:
    classes = populate_class_labels()
    net = cv2.dnn.readNet(weights_file_abs_path, config_file_abs_path)
    initialize = False

blob = cv2.dnn.blobFromImage(image, scale, (416,416), (0,0,0), True, crop=False)

net.setInput(blob)

outs = net.forward(get_output_layers(net))

class_ids = []
confidences = []
boxes = []

for out in outs:
    for detection in out:
        scores = detection[5:]
        class_id = np.argmax(scores)
        max_conf = scores[class_id]
        if max_conf > confidence:
            center_x = int(detection[0] * Width)
            center_y = int(detection[1] * Height)
            w = int(detection[2] * Width)
            h = int(detection[3] * Height)
            x = center_x - w / 2
```

```
                y = center_y - h / 2
                class_ids.append(class_id)
                confidences.append(float(max_conf))
                boxes.append([x, y, w, h])


    indices = cv2.dnn.NMSBoxes(boxes, confidences, confidence, nms_thresh)

    bbox = []
    label = []
    conf = []

    for i in indices:
        i = i
        box = boxes[i]
        x = box[0]
        y = box[1]
        w = box[2]
        h = box[3]
        bbox.append([round(x), round(y), round(x+w), round(y+h)])
        label.append(str(classes[class_ids[i]]))
        conf.append(confidences[i])

    return bbox, label, conf
```

about.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>VirtualEye About</title>
    <link rel= "stylesheet" type= "text/css" href= "{{ url_for('static',filename='styles/style.css') }}">
    <style>
        body {
    padding: 0;
    margin: 0;
    font-weight: bold;
    font-family: sans-serif;
    font-weight: bold;
    display: flex;
```

```
flex-direction: column;
}
.navbar {
    width: 100%;
    display: flex;
    flex-direction: row;
    top: 0;
    padding: 5px 0;
    background-color: black;
    color: white;
    font-family: sans-serif;
}
.navbar h1 {
    margin-left: 20px;
    text-shadow: 2px 2px 2px black;
    margin-right: 70%;
}
.navlinks {
    align-items: center;
    right: 20px;
    display: flex;
    flex-direction: row;
    margin-bottom: 5px;
}
nav a {
    margin: 0 auto;
    text-decoration: none;
    color: white;
    font-family: sans-serif;
    margin: 5px 15px;
    text-shadow: 2px 2px 2px black;
}
.footer {
    position: fixed;
    text-align: center;
    left: 0;
    bottom: 0;
    width: 100%;
    background-color: black;
    color: white;
    text-align: center;
}
    #heading {
        margin: 50px auto;
        }
        .container{
            width: 90%;
            margin: 20px auto;
        }
        .stuff-container {
            width: fit-content;
        }
        .stuff{
            width: 45%;
            float: left;
            padding: 10px;
            text-align: justify;
        }
        .stuffR{
            text-align: justify;
        }
        .write-up {
            width: 90%;
            margin: 0 auto;
        }
        h3,h2{
            text-align: center;
        }
    </style>

</head>
<body>
    <div class="navbar">
        <h1>Virtual Eye</h1>
        <div class="navlinks">
            <nav><a href="/about" style="color: yellow;">about</a></nav>
            <nav><a href="./demo">demo</a></nav>
            <nav><a href="./logout">log-out</a></nav>
        </div>
    </div>
    <h1 id="heading">Virtual Eye - Life Guard for Swimming Pools to Detect Active
Drowning</h1>
    <div class="container">
        <h2>ABOUT PROJECT</h2>
        <div class="stuff-container">
            <div class="stuff">
                <h3>Problem:</h3>
                <p class="write-up">Swimming is one of the best exercises that helps people to
reduce stress in this urban lifestyle. Swimming pools are found larger in number in hotels, and
weekend tourist spots and barely people have them in their house backyard. Beginners,
especially, often feel it difficult to breathe underwater which causes breathing trouble which in
turn causes a drowning accident. Worldwide, drowning produces a higher rate of mortality
without causing injury to children. Children under six of their age are found to be suffering the
highest drowning mortality rates worldwide. Such kinds of deaths account for the third cause of
unplanned death globally, with about 1.2  million cases yearly. To overcome this conflict, a
meticulous system is to be implemented along the swimming pools to save human life</p>
            </div>
            <div class="stuff stuffR">
                <h3>Solution:</h3>
                <p class="write-up">By studying body movement patterns and connecting cameras to
artificial intelligence (AI) systems we can devise an underwater pool safety system that reduces
the risk of drowning.  Usually, such systems can be developed by installing more than 16
cameras underwater and ceiling and analyzing the video feeds to detect any anomalies. but  AS
a POC we make use of one camera that streams the video underwater and analyses the
position of swimmers to assess the probability of drowning, if it is higher then an alert will be
generated to attract lifeguards' attention.</p>
            </div>
        </div>
    </div>
    <div class="footer">
        <p style="color: white"><b>Copyrights &#169; 2022. All Rights Reserved.</p>
    </div>
</body>
</html>
```

demo.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>VirtualEye</title>
    <link rel="stylesheet" href="style.css">
    <style>
      body {
    padding: 0;
    margin: 0;
    font-weight: bold;
    font-family: sans-serif;
    font-weight: bold;
    display: flex;
    flex-direction: column;
}
.navbar {
    width: 100%;
    display: flex;
    flex-direction: row;
    top: 0;
    padding: 5px 0;
    background-color: black;
    color: white;
    font-family: sans-serif;
}
.navbar h1 {
    margin-left: 20px;
    text-shadow: 2px 2px 2px black;
    margin-right: 70%;
}
.navlinks {
    align-items: center;
    right: 20px;
    display: flex;
    flex-direction: row;
    margin-bottom: 5px;
}
nav a {
    margin: 0 auto;
    text-decoration: none;
    color: white;
    font-family: sans-serif;
    margin: 5px 15px;
    text-shadow: 2px 2px 2px black;
}
.footer {
    position: fixed;
    text-align: center;
    left: 0;
    bottom: 0;
    width: 100%;
    background-color: black;
    color: white;
    text-align: center;
```

```
}
    .grid-container {
      display: grid;
      grid-template-columns: 1fr 1fr;
      grid-gap: 20px;
    }
    .grid-child {
      display: flex;
      flex-direction: column;
      padding: 20px;
      border: 2px solid lightgrey;
      border-radius: 20px;
      margin: 0 20px;
      text-align: justify;
    }
    img {
      height: 100%;
      width: 100%;
    }
    .center {
      justify-content: center;
      align-items: center;
      height: 200px;
      border: 3px solid green;
  }
  #heading {
    margin: 50px auto;
  }
  #demoBtn {
    position: relative;
    width: 90%;
    padding: 10px 0;
    bottom: 10px;
    margin: 50px auto;
    background-color: black;
    border-radius: 10px;
    color: white;
    font-weight: bold;
    font-size: large;
  }
  </style>
</head>
<body>
  <div class="navbar">
```

```
      <h1>Virtual Eye</h1>
      <div class="navlinks">
        <nav><a href="./about">about</a></nav>
        <nav><a href="./demo" style="color: yellow;">demo</a></nav>
        <nav><a href="./logout">log-out</a></nav>
      </div>
  </div>
      <h1 id="heading">
        Virtual Eye - Life Guard for Swimming Pools to Detect Active Drowning
      </h1>
    <hr />
    <div class="grid-container">
      <div class="grid-child">
        <p>Swimming is one of the best exercises that helps people reduce stress in this urban lifestyle. Swimming pools are found in large numbers in hotels, weekend tourist spots and in some rare cases, people's backyards. Beginners often find it difficult to control their breath while underwater and this may cause breathing trouble which in turn may cause a drowning accident. Worldwide, drowning produces a high rate of mortality without causing injury among children. Children under the age of 6 are found to be at the greatest risk of drowning. Such kinds of deaths account for a third of accidental deaths globally, with about 1.2 million cases yearly. Thus, we need a suitable system in place to detect active drowning to prevent loss of life.
        </p>
        <form action="./result" method="GET">
          <input type="submit" id="demoBtn" value="click me for a demo"/>
        </form>
      </div>


      <div class="grid-child">
        <img src="{{url_for('static', filename='drowning.jpeg')}}" alt="Kanye" />
      </div>
    </div>
    <div style="background-color: black; width: 100%">
    <div class="footer">
        <p style="color: white"><b>Copyrights &#169; 2022. All Rights Reserved.</p>
    </div>
  </body>
</html>
```

login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>login</title>
    <link rel="stylesheet" href="style.css">
    <style>
        body {
    padding: 0;
    margin: 0;
    font-weight: bold;
    font-family: sans-serif;
    font-weight: bold;
    display: flex;
    flex-direction: column;
}
.navbar {
    width: 100%;
    display: flex;
    flex-direction: row;
    top: 0;
    padding: 5px 0;
    background-color: black;
    color: white;
    font-family: sans-serif;
}
.navbar h1 {
    margin-left: 20px;
    text-shadow: 2px 2px 2px black;
    margin-right: 70%;
}
.navlinks {
    align-items: center;
    right: 20px;
    display: flex;
    flex-direction: row;
    margin-bottom: 5px;
}
nav a {
    margin: 0 auto;
    text-decoration: none;
    color: white;
    font-family: sans-serif;
    margin: 5px 15px;
    text-shadow: 2px 2px 2px black;
}
.footer {
    position: fixed;
    text-align: center;
    left: 0;
    bottom: 0;
    width: 100%;
    background-color: black;
    color: white;
    text-align: center;
  }
    .fields {
            border: 1px solid grey;
            display: flex;
            flex-direction: column;
            margin: 7.5% auto;
            padding: 10px;
            width: 50%;
            border-radius: 20px;
            margin-bottom: 0;
        }
        #logo {
            height: 100px;
            width: 200px;
            margin: 30px auto;
        }
        .field {
            font-size: large;
            min-height: 30px;
            margin: 20px;
            border-radius: 10px;
        }

        #login {
            background-color: black;
            color: white;
            padding: 10px 0;
            font-weight: bold;
        }
        #goToReg {
            margin: 30px auto;
            width: 50%;
            text-align: center;
        }
    </style>
</head>
<body>
    <div class="navbar">
        <h1>Virtual Eye</h1>
    </div>
    <div class="content">
        <div>
            <form class="fields" action="./afterlogin" method="GET">
                <center><h4>{{pred}}</h4></center>
                <img id="logo" src="{{url_for('static', filename='eye.png')}}" alt="virtualEye">
                <input class="field" type="text" name="uname" placeholder="Enter username" />
                <input class="field" type="password" name="password" placeholder="Enter
password" />
                <input type="submit" class="field" id="login" value="login" />
            </form>
        </div>
    </div>
    <div id="goToReg">don't have an account? <a href="./register">register here</a></div>
    <div class="footer">
        <p style="color: white"><b>Copyrights &#169; 2022. All Rights Reserved.</p>
    </div>
</body>
</html>
```

logout.html

```
<html>
    <head>
    <link rel="stylesheet" href="style.css">
    <style>
        body {
    padding: 0;
    margin: 0;
    font-weight: bold;
    font-family: sans-serif;
    font-weight: bold;
    display: flex;
    flex-direction: column;
}
.navbar {
    width: 100%;
    display: flex;
    flex-direction: row;
    top: 0;
    padding: 5px 0;
    background-color: black;
    color: white;
    font-family: sans-serif;
}
.navbar h1 {
    margin-left: 20px;
    text-shadow: 2px 2px 2px black;
    margin-right: 70%;
}
.navlinks {
    align-items: center;
    right: 20px;
    display: flex;
    flex-direction: row;
    margin-bottom: 5px;
}
nav a {
    margin: 0 auto;
    text-decoration: none;
    color: white;
    font-family: sans-serif;
    margin: 5px 15px;
    text-shadow: 2px 2px 2px black;
}
.footer {
    position: fixed;
    text-align: center;
    left: 0;
    bottom: 0;
    width: 100%;
    background-color: black;
    color: white;
    text-align: center;
  }
        .content {
            margin: 10% auto;
            text-align: center;
        }
        #login {
            background-color: black;
            color: white;
            padding: 10px 0;
            font-weight: bold;
            width: 300px;
            border-radius: 10px;
            font-size: large;
        }
    </style>
    </head>
    <body>
        <div class="navbar">
            <h1>Virtual Eye</h1>
            <div class="navlinks">
                <nav><a href="./login">login</a></nav>
                <nav><a href="./register">register</a></nav>
            </div>
        </div>
        <div class="content">
            <h1>Successfully Logged Out!</h3>
            <h3 id = "info" >Login for more information</h5>
            <button class="field" id="login" onclick="window.location.href = '/';">Log In</button>
        </div>
        <div class="footer">
            <p style="color: white"><b>Copyrights &#169; 2022. All Rights Reserved.</p>
        </div>
    </body>
</html>
```

register.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>register</title>
    <link rel="stylesheet" href="style.css">
    <style>
      body {
    padding: 0;
    margin: 0;
    font-weight: bold;
    font-family: sans-serif;
    font-weight: bold;
    display: flex;
    flex-direction: column;
}
.navbar {
    width: 100%;
    display: flex;
    flex-direction: row;
    top: 0;
    padding: 5px 0;
    background-color: black;
    color: white;
    font-family: sans-serif;
}
.navbar h1 {
    margin-left: 20px;
    text-shadow: 2px 2px 2px black;
    margin-right: 70%;
}
.navlinks {
    align-items: center;
    right: 20px;
    display: flex;
    flex-direction: row;
    margin-bottom: 5px;
}
nav a {
    margin: 0 auto;
    text-decoration: none;
    color: white;
    font-family: sans-serif;
    margin: 5px 15px;
    text-shadow: 2px 2px 2px black;
}
.footer {
    position: fixed;
    text-align: center;
    left: 0;
    bottom: 0;
```

```
    width: 100%;
    background-color: black;
    color: white;
    text-align: center;
   }
    .fields {
        border: 1px solid grey;
        display: flex;
        flex-direction: column;
        margin: 5% auto;
        padding: 10px;
        width: 50%;
        border-radius: 20px;
        margin-bottom: 0;
    }
    #logo {
        height: 100px;
        width: 200px;
        margin: 30px auto;
    }
    .field {
        font-size: large;
        min-height: 30px;
        margin: 20px;
        border-radius: 10px;
    }
    #register {
            background-color: black;
            color: white;
            padding: 10px 0;
            font-weight: bold;
        }
    #goToLogin {
        margin: 30px auto;
        width: 50%;
        text-align: center;
    }
    </style>
</head>
<body>
    <div class="navbar">
        <h1>Virtual Eye</h1>
    </div>
    <div>
```

```
        <form class="fields" action="./afterreg" method="GET">
                <img id="logo" src="{{url_for('static', filename='eye.png')}}" alt="virtualEye">
                <input class="field" name="uname" type="text" placeholder="Enter Name" />
                <input class="field" name="password" type="password" placeholder="Enter
password" />
                <input type="submit" class="field" id="register" value="register"/>
            </form>
        </div>
    </div>
    <div id="goToLogin">have an account? <a href="/">login here</a></div>
    <div class="footer">
        <p style="color: white"><b>Copyrights &#169; 2022. All Rights Reserved.</p>
    </div>
</body>
</html>
```