PROJECT REPORT

VIRTUAL EYE – LIFE GUARD FOR SWIMMING POOLS TO DETECT ACTIVE DROWNING

Team ID : PNT2022TMID03921

Team Members:

1. CHANDRASEKRAN D K(412519104021) [TEAM LEADER]
2. SELVAGANAPATHY G (412519104120)
3. SATHISH N (412519104118)
4. BALAKRISHNAN K (412519104501)

TABLE OF CONTENTS :

INTRODUCTION

Drowning incidents are potentially severe but thankfully rare for most lifeguards. Due to the infrequency of drowning incidents, the visual search for such occurrences is challenging (Lanagan-Leitzel, Skow & Moore, 2015). The difficulties involved in detecting infrequent drowning targets are reflected in other areas of real-world visual search with uncommon target items, such as airport security screenings (Wolfe, Horowitz & Kenner, 2005; Biggs & Mitroff, 2015). For example, Wolfe et al., (2005) found low-prevalence targets (occurring on 1% of trials) were missed more frequently than high-prevalence targets (occurring on 50% of trials), with error rates of 30% and 7%, respectively.

In regards to lifeguarding, visual search has been defined as observing part of an aquatic environment (beaches, pools, open water), and processing and assessing the events happening within that location (Fenner et al., 1999). While this definition suggests that the surveillance of the water is a fundamental and critical role of the lifeguard, there is relatively little focus on training in these areas (Lanagan-Leitzel & Moore, 2010). This is reflected in the UK National Pool Lifeguard Qualification (NPLQ) training manual (Blackwell, 2016), where only 6 out of 214 pages are dedicated to the education of scanning and observation behaviours (Blackwell et al., 2012). With this limited focus on visual training, lifeguards may be underprepared for detecting struggling swimmers in a timely manner.

PROJECT OVERVIEW:-

Lifeguard surveillance is a complex task that is crucial for swimmer safety, though few studies of applied visual search have investigated this domain. This current study compared lifeguard and non-lifeguard search skills using dynamic, naturalistic stimuli (video clips of confederate swimmers) that varied in set size and type of drowning. Lifeguards were more accurate and responded faster to drowning targets. Differences between drowning targets were also found: passive drownings were responded to less often, but more quickly than active drownings, highlighting that passive drownings may be less salient but are highly informative once detected. Set size effects revealed a dip in reaction speeds at an intermediate set-size level, suggesting a possible change in visual search strategies as the array increases in size. Nonetheless, the ability of the test to discriminate between lifeguards and non-lifeguards offers future possibilities for training and assessing lifeguard surveillance skills.

PURPOSE:-

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.

LITERARTURE SURVEY

➢ EXISTING PROBLEM:-

Paper 1 :

NAME : An automatic video based drowning detection system for swimming pools using active contours.

AUTHOR : Nasrin Salehi, Maryam Keyvanara and Seyed Amirhassan Monadjemmi

MERITS : In this paper, a real time drowning detection method based on HSV color space analysis is presented which uses prior knowledge of the video sequences to set the best values for the color channels

CONCLUSION : The presented software can detect drowning person in indoor swimming pools and sends an alarm to the lifeguard rescues if the previously detected person is missing for a specific amount of time

---

Paper 2 :

NAME : The effect of lifeguard experience upon the detection of drowning victims in a realistic dynamic visual search task.

AUTHOR: Victoria Laxton and David Crundall

MERIT : limited time is given for the child or adult after the threshold time , even though it is same triggers the alarm.

CONCLUSION : reliable solution where the life guards have difficulty in monitoring the swimmers likea highly crowded sea.

PAPER 3 :

NAME : Poseidon - Video based drowning detection system in the swimming pool.

AUTHOR : valvi Priyanka , Prof. mr.Amar Palwankar

MERITS : if the swimmer faces any problem it will detect it by its motion and gives an alert.

CONCLUSION : proves it can be a reliable multimedia video-based surveillance system.

PAPER 4 :

NAME : DEWS: A Live Visual Surveillance System for Early Drowning Detection at Pool

AUTHOR : How - Lung ,Kar Ann toh , Junxian wang

MERITS : Examples of interesting behaviors, i.e., distress, drowning, treading and numerous swimming styles, are simulated and collected.

CONCLUSION : Experimental results show that we have established a prototype system which is robust and beyond the stage of proof-of-concept.

PAPER 5 :

NAME : Drowning Detection Based on Background Subtraction

AUTHOR : Lei Fei , Xang Xueli , Chen Donsheng

MERITS : method is effective to detect the drowners and eliminate the shadows.

CONCLUSION : gives a clear view of the drowners not by checking over their shadows.

PROBLEM STATEMENT DEFINITION:-

This project describes the drowning detection system for the prevention of drowning incidents in swimming pools. The problem boundary clearly distinguishes between the positive samples which are inside the boundary to those that are less relevant and outside the
boundary. It works like an “extra lifeguard” for under the water of swimming pools. For instance, if it happens to someone to drown inside the swimming pool, it makes them take an excess amount of water content which affectsthe internal organs and sometimes it may be the cause of death. This detection system tracks the movements of everything inside the water bodies and will help to guard the lives by finding them easily.

Classification of drowning stages:

Stage 1*:* User needs a way to detect a person from drowning, because now a days lots of kids lose their livesby drowning.

Stage 2: User is a fireman who needs a way to detect a person from drowning because he needs to savepeople.

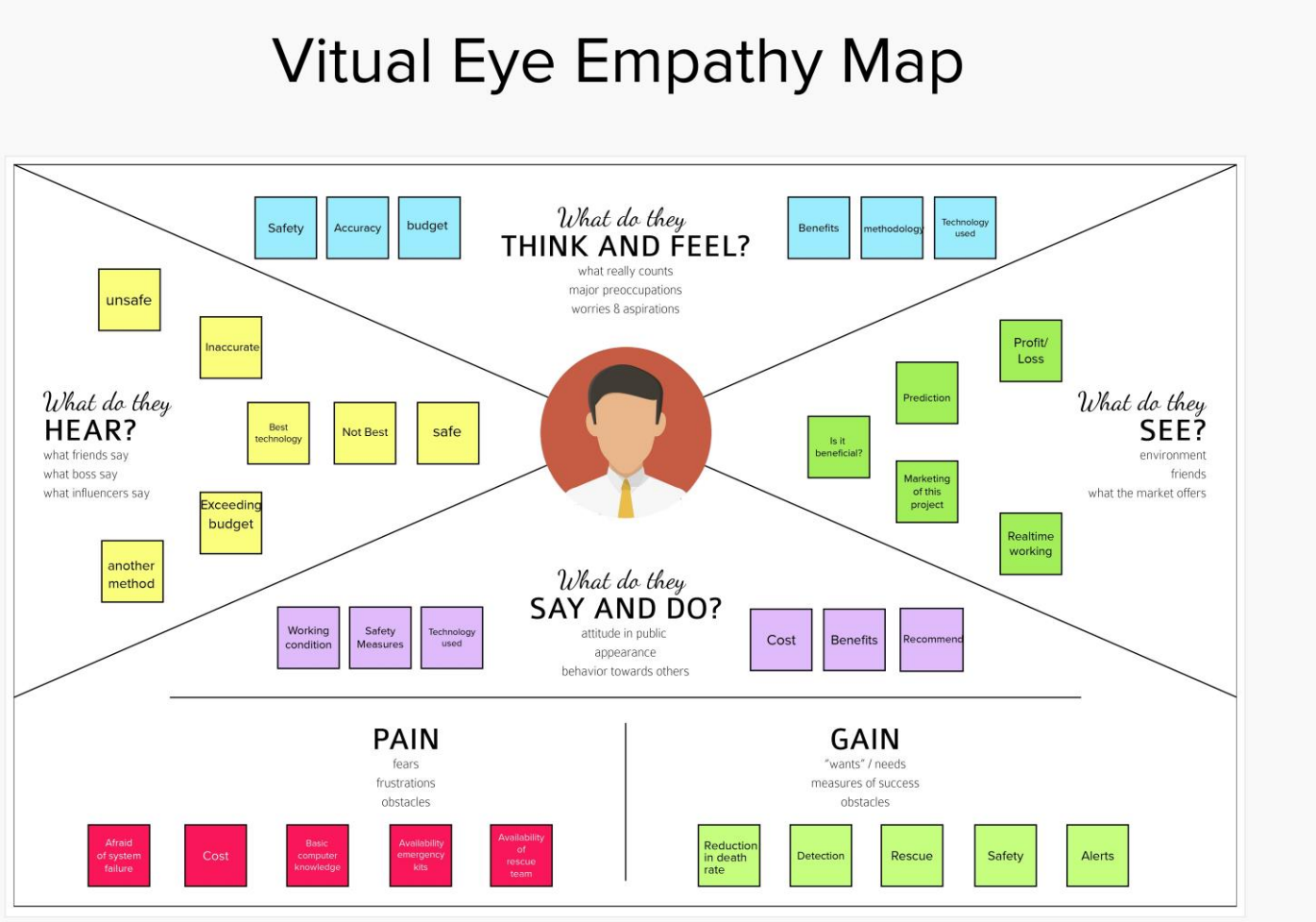
Stage 3: User need a way to detect the real timelive environment and safeguard system.

Stage 4: Distance: Z-axis coordinates from the depth information acquired by the camera, corrected byadding from the deviation in the y-axis direction (straight line distance from the camera to the nose).

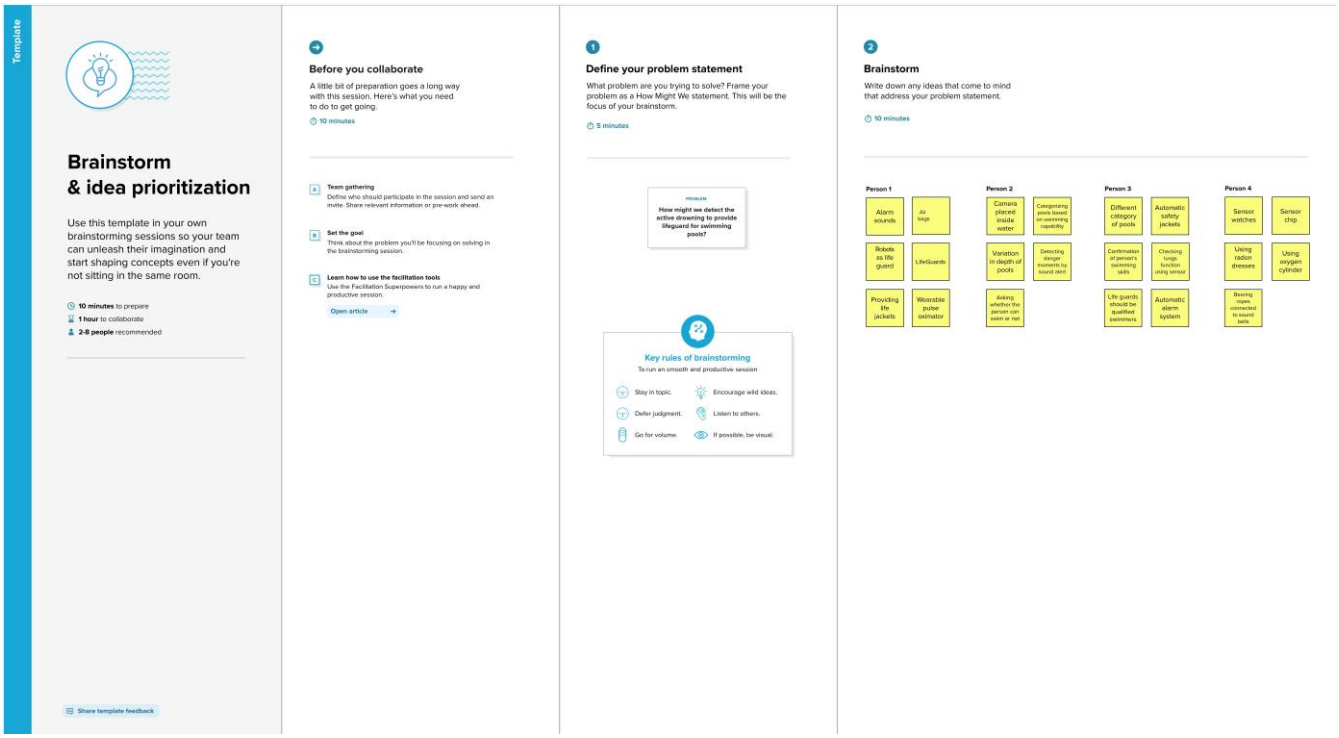
Stage 5: Depth: Z-axis coordinates obtained from thedepth information acquired by the camera.

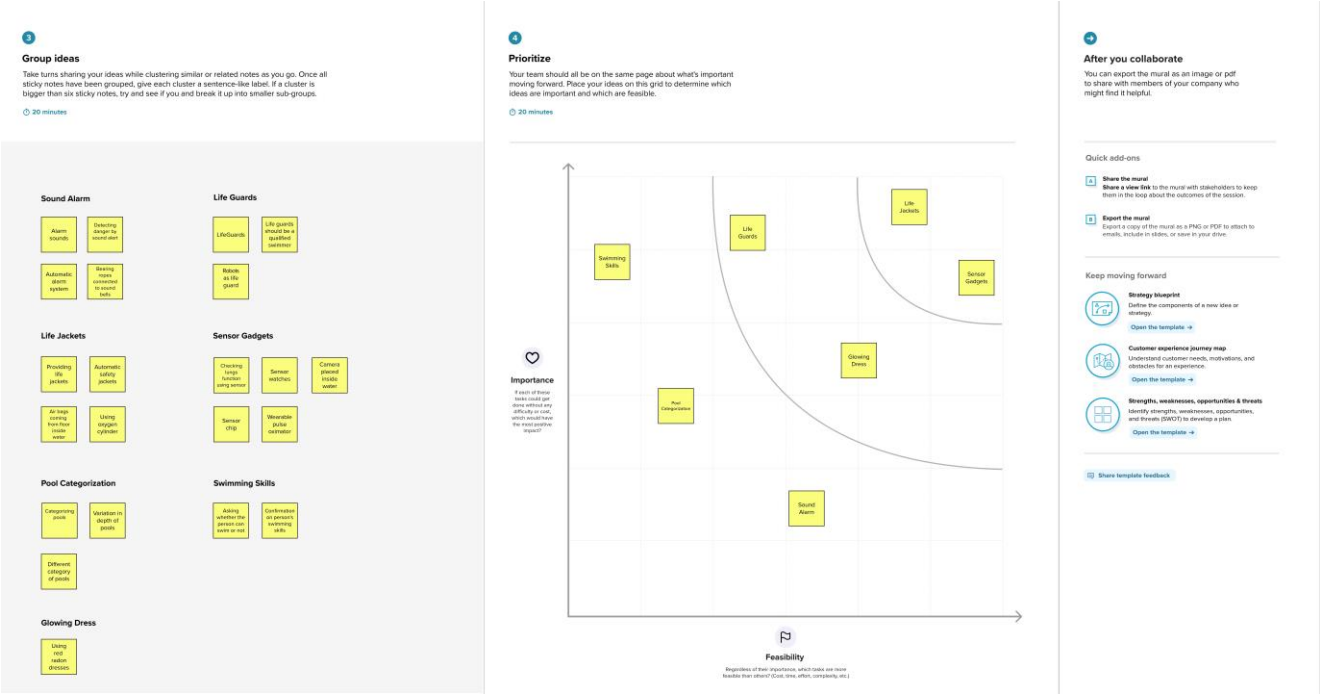
IDEATION AND PROPOSED SOLUTION

EMPATHY MAP CANVAS:-



IDEATION AND BRAINSTORMING:-





PROPOSED SOLUTION:-

S.No.	Parameters	Description
1.	Problem Statement (Problem to be solved)	Detection of active drowning in swimming pools and alert the life guards and to save the person who is drowning.
2.	Idea / Solution description	When the swimmer is drowning the camera detects it and alert the lifeguards by producing alarm sounds. And after the detection the automatic lifejackets will be popped in the drowning area for faster safety.
3.	Novelty / Uniqueness	Deep Learning for detection. Alarm for alert. Automatic life jackets for safety measures.
4.	Social Impact / Customer Satisfaction	The videos and images on the working of the model and recorded videos can be shown to the customers for their satisfaction.
5.	Business Model (Revenue Model)	This model can be recommended to other swimming pools if the working of the model leads to the more safety of the customers. Customers can also recommend this to others with their usage experience.
6.	Scalability of the Solution	With the help of sound alarm the lifeguards will be alerted quickly. In case lifeguards are not attentive the automatic lifejacket plays a major role in saving the customer from drowning.

PROBLEM SOLUTION FIT

Define CS, fit into CL

1.CUSTOMER SEGMENT(S) CS

Customers are the people who are going to swim in the swimming pool

6.CUSTOMER LIMITATIONS DC, BUDGET, DEVICES CL

The expectation of the customer for the budget will be low with high performance of the model.

5.AVAILABLE SOLUTIONS PROS & CONS AS

When a camera detects an active drowning of a person it alerts the lifeguards by sound alarm. But sometimes they can be late. So we can use automatic lifejacket.

Explore AS, differentiate

Focus on PR, tap into BE, understand RC

2.PROBLEMS / PAINS + ITS FREQUENCY PR

The fastest way to provide safety is by implementing automatic lifejackets and its cost can be high.

9.PROBLEM ROOT / CAUSE RC

The problem can be occur if the swimmer do not know swimming or if the swimmer is allergical to the chlorine used in pools or if he drinks the water too much while swimming or else his body conditions are weak.

7.BEHAVIOR + ITS INTENSITY BE

In order to save the person who is drowning, the behavior of the person must be detected by the camera. The intensity of the detection by the camera can be improved by providing many similar actions of the persons while drowning through deep learning.

Focus on PR, tap into BE, understand RC

Identify strong TR & EM

3.TRIGGERS TO ACT TR

Lifeguards must be triggered by sound alarms to ensure safety to swimmers.

4.EMOTIONS BEFORE / AFTER EM

The swimmer will be frightened to swim and in case if he tries to swim and starting to drown he will be trembled and moves hands faster to come above water surface.

10.YOUR SOLUTION SL

Thus in order to protect customers the camera must be placed covering all the pool areas. Swimming pool can be categorized for both the persons who knows or who do not know the swimming. Also life jackets can be provided for safety purpose.

8.CHANNELS OF BEHAVIOR CH

ONLINE

The customers should be contact with the help of email by registration process.

OFFLINE

The customers should be in contact by the lifeguards.

Extract online and offline CH of BE

REQUIREMENT ANALYSIS

Functional requirement:-

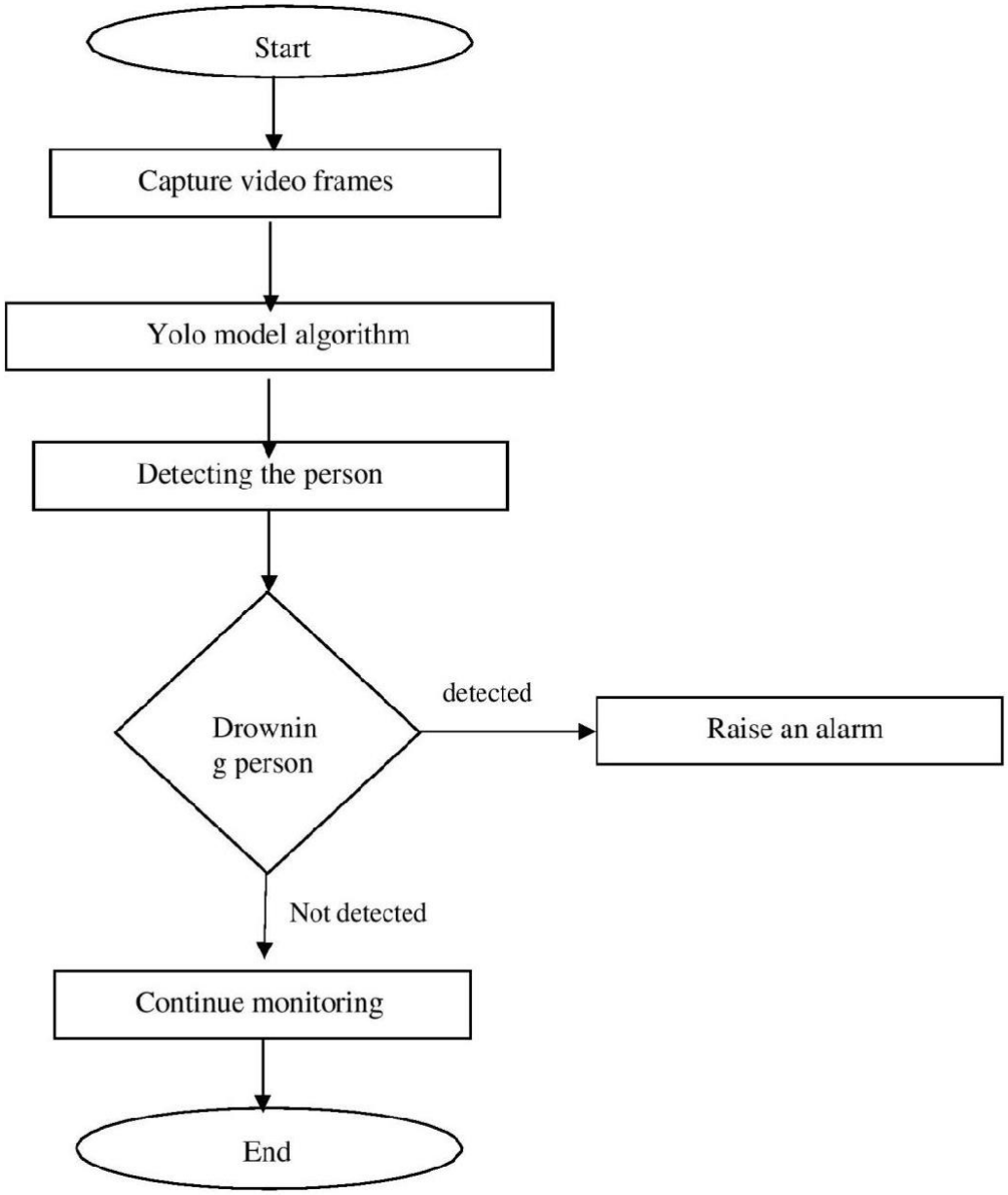
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Alarm system	Monitor and detect the drowning person Alert the lifeguard by trigger the alarm
FR-4	Output	Visual representation Image detection Report generation

Non-Functional Requirement

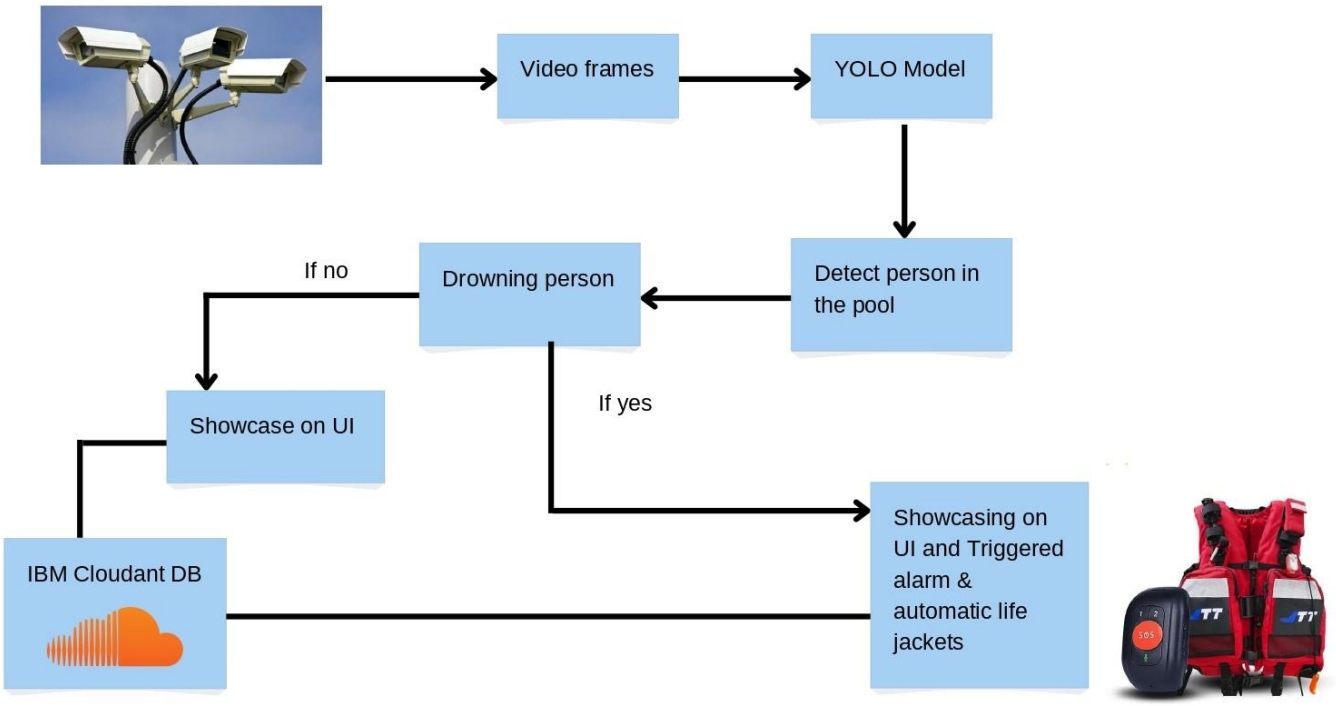
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Eco – Friendly.
NFR-2	Security	Observing each and every body movement of the swimmers.
NFR-3	Reliability	Suitable for all the swimming pools.
NFR-4	Performance	Life guard can visually access the developing situation within seconds of the event first occurring and initiate the rescue procedure when necessary.
NFR-5	Availability	24/7 monitoring cameras.
NFR-6	Scalability	Its comfortable for all swimmers. The lifespan is high. Work more efficiently.

PROJECT DESIGN

Data Flow Diagram:-



Solution and Technical Architecture:-



PROJECT PLANNING AND SCHEDULING

Sprint Planning and Estimation:-

S.NO	MILESTONE	DESCRIPTION	DURATION
1	Prerequisites	Prerequisites are all the needs atthe requirement level needed for the execution of the different phasesof a project.	1 WEEK
2	Create & Configure IBM cloud services	IBM Cloud provides solutions that enable higher levels of compliance, security, and management, with proven architecture patterns and methodsfor rapid delivery for running mission critical workloads.	2 WEEK

3	Develop the python script	A Python script is a set of commands included in a file that isintended to be run similarly to a program. The concept is that the file will be run or performed from the command line or from within a Python interactive shell to performa particular activity. Of course, thefile includes methods and imports different modules.	1 WEEK
4	Develop web application	A web application (or web app) is application software that runs in aweb browser, unlike software programsthat run locally and natively on theoperating system (OS) of the device.	3 WEEK
5	Ideation phase	Ideation is the process where you generate ideas and solutions through sessions such as Sketching, Prototyping, Brainstorming, Brain writing, Worst Possible Idea, and a wealth of other ideation techniques.	1 WEEK

6	Project design phases	Project design is an early phase ofa project where the project's keyfeatures, structure, criteria for success, and major deliverables areplanned out. The aim is to develop one ormore designs that can be used toachieve the desired project goals.	2 WEEK
7	Project planning phase	In the Planning Phase, the Project Manager works with the project team to create the technical design, task list, resource plan, communications plan, budget, and initial schedule for the project, and establishes the roles and responsibilities of the project team and its stakeholders.	2 WEEK
8	Project development phase	Project development is the process of planning and allocating resources to fully develop a project or product from concept to go-live.	4 WEEK

Sprint Delivery Schedule:-

Sprint	Total Story Points	Duration	Sprint start date	Sprint end date (planned)	Story points completed (as on planned end date)	Srint release date (Actual)
Sprint-1	10	6 Days	24 Oct 2022	29 Oct 2022	6	29 OCT 2022
Sprint-2	18	6 Days	31 Oct 2022	05 Nov 2022	12	31 OCT 2022
Sprint-3	16	6 Days	07 Nov 2022	12 Nov 2022	11	12 NOV 2022
Sprint-4	12	6 days	14 Nov 2022	19 Nov 2022	12	14 NOV 2022

Velocity:

Sprint 1 : Average velocity = 8/6 = 1.3 V

Sprint 2 : Average velocity = 14/6 = 2.3 V

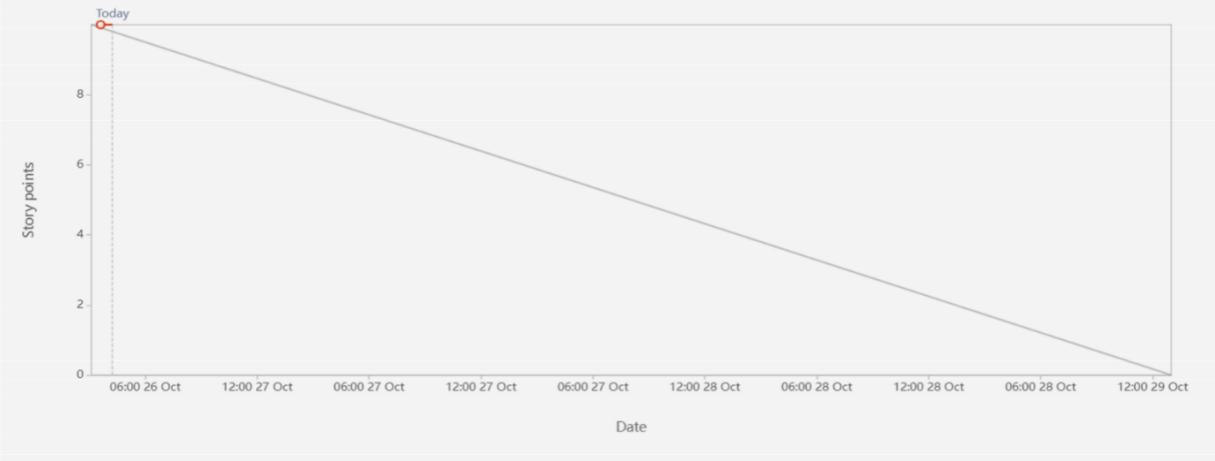
Sprint 3 : Average velocity = 16/6 = 2.6 V

Sprint 4 : Average velocity = 12/6 = 2.0 V

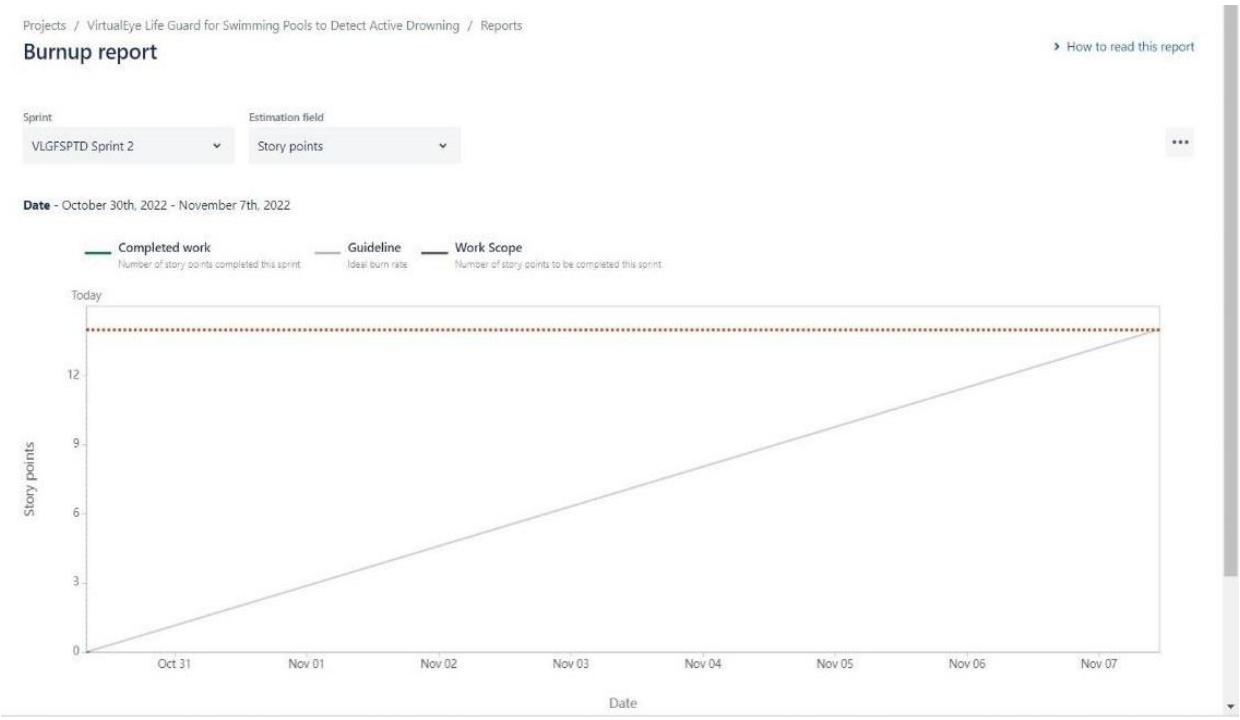
Total Team Average Velocity = 2.08 V

Reports from JIRA:-

BURN DOWN CHART:



BURN UP CHART :



CODING AND SOLUTIONING

_NOTE : codes are available in the APPENDIX

FEATURE 1 :

FOR HTML CODE :

For this project create three HTML files namely

- **index.html**
- **base.html**
- **register.html**
- **login.html**
- **prediction.html**
- **logout.html**

and save them in the templates folder.

FEATURE 2 :

FOR PYTHON CODE ;

A. Import the libraries
B. Create a database using an initiated client.
C. An object of Flask class is our WSGI application. Flask constructor takes the name of the currentmodule (_name_) as argument.
D. Configure the registration page
E. Configure the login page

F. For logout from web application.

G. Create res() function for drowning detection

H. Creating bounding box

I. Main Function

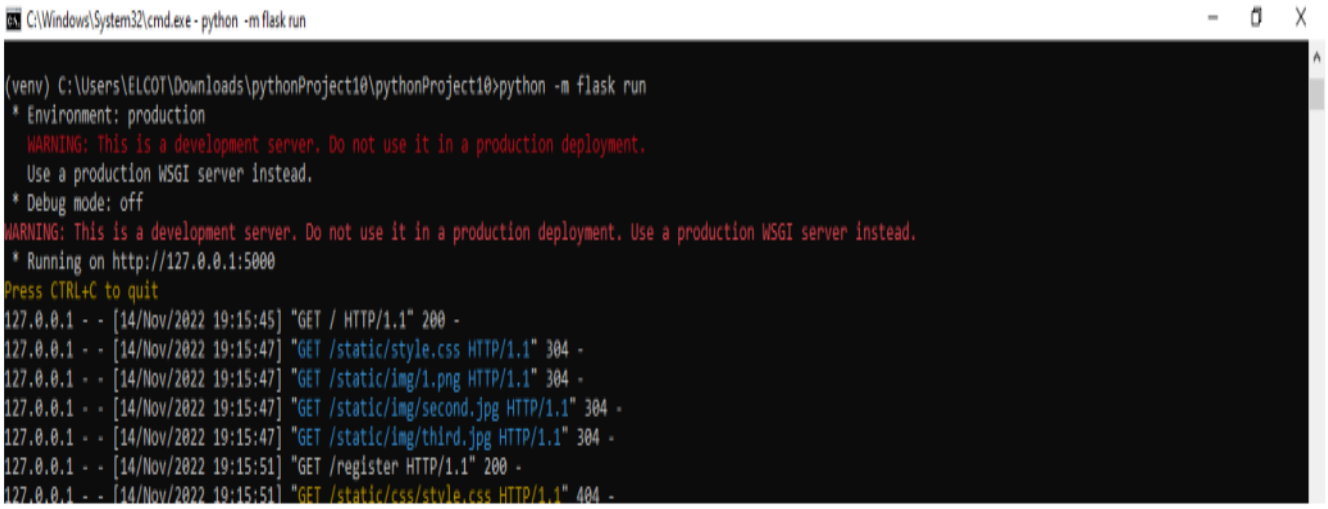
RUN THE APPLICATION :

Run the application

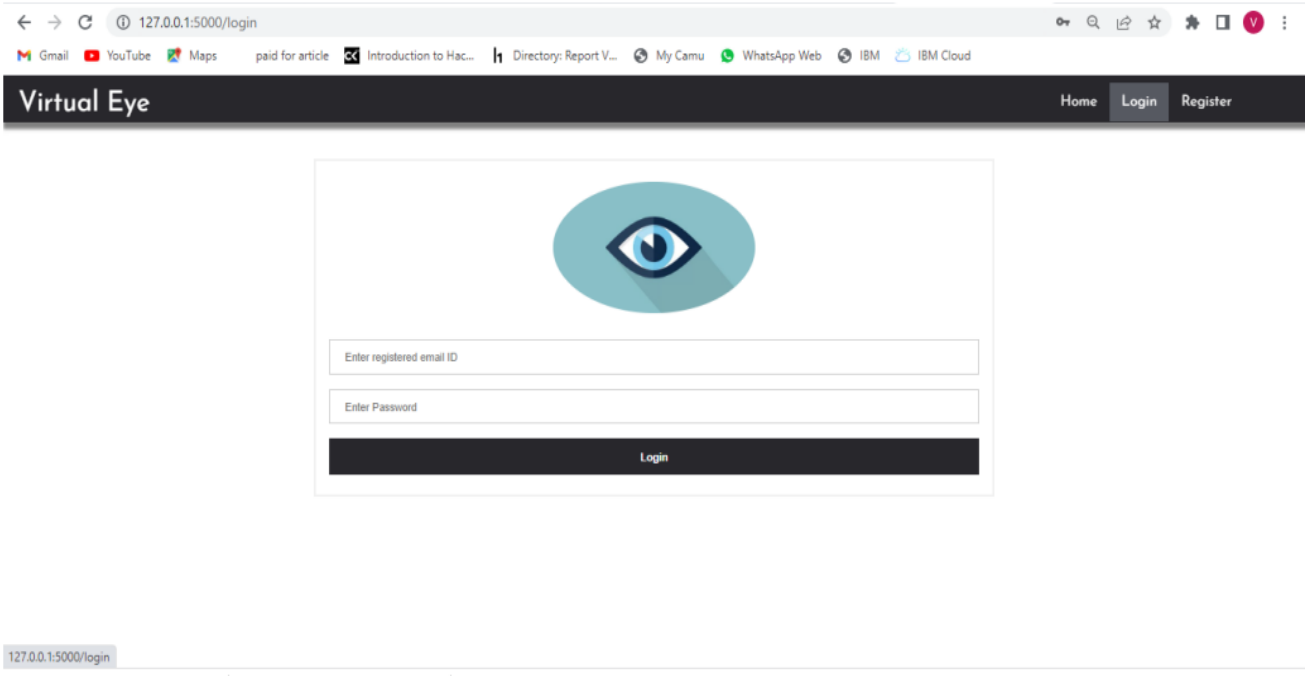
- Open the anaconda prompt from the start menu
- Navigate to the folder where your python script is.
- Now type the "python app.py" command
- Navigate to the localhost where you can view your web page.
- Click on the predict button from the top right corner, enter the inputs, click on the submit button, and see the result/prediction on the web.

TESTING

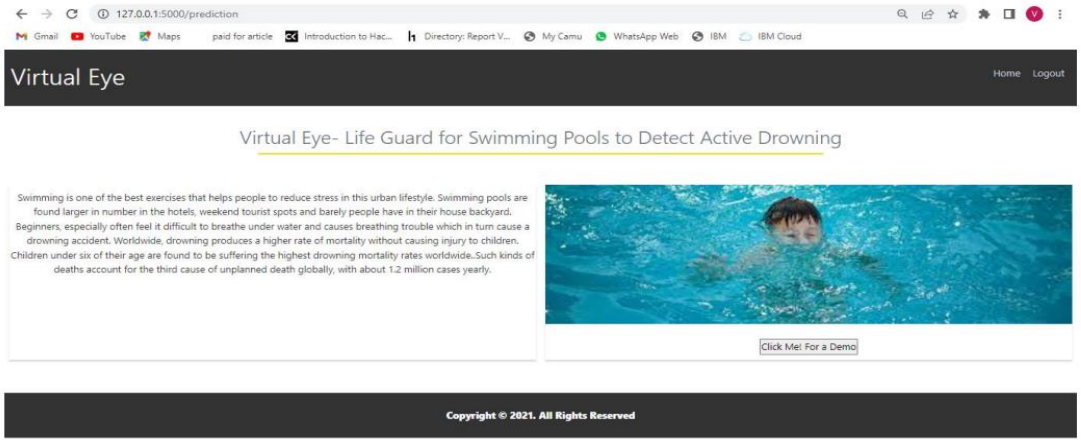
USER ACCEPTANCE TESTING :



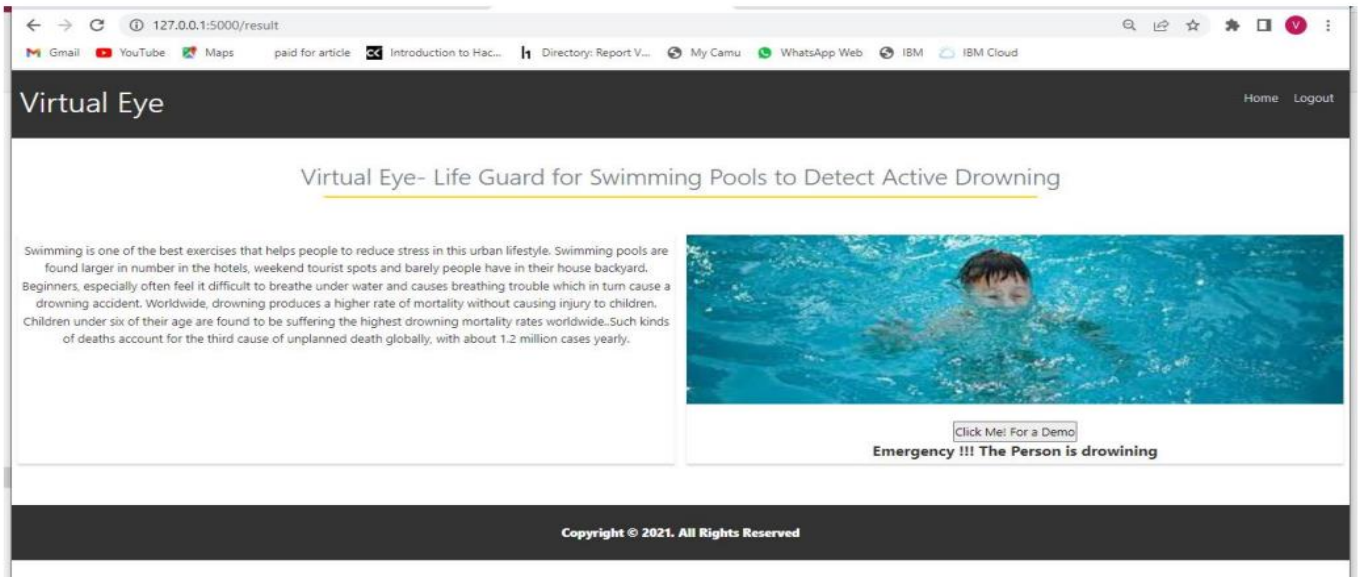
While logging in you need to provide your registered credentials

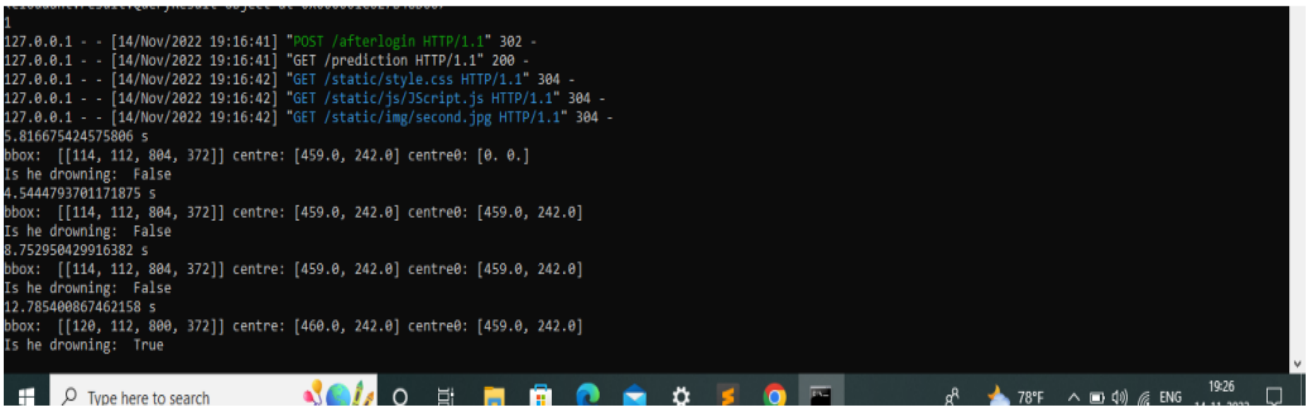


After successfully login you will redirect to the prediction page where we have to click on the demo button to launch the opencv window for video analysis.



Output:-




1
127.0.0.1 - - [14/Nov/2022 19:16:41] "POST /afterlogin HTTP/1.1" 302 -
127.0.0.1 - - [14/Nov/2022 19:16:41] "GET /prediction HTTP/1.1" 200 -
127.0.0.1 - - [14/Nov/2022 19:16:42] "GET /static/style.css HTTP/1.1" 304 -
127.0.0.1 - - [14/Nov/2022 19:16:42] "GET /static/js/JScript.js HTTP/1.1" 304 -
127.0.0.1 - - [14/Nov/2022 19:16:42] "GET /static/img/second.jpg HTTP/1.1" 304 -
5.816675424575806 s
bbox: [[114, 112, 804, 372]] centre: [459.0, 242.0] centre0: [0. 0.]
Is he drowning: False
4.5444793701171875 s
bbox: [[114, 112, 804, 372]] centre: [459.0, 242.0] centre0: [459.0, 242.0]
Is he drowning: False
8.752950429916382 s
bbox: [[114, 112, 804, 372]] centre: [459.0, 242.0] centre0: [459.0, 242.0]
Is he drowning: False
12.785400867462158 s
bbox: [[120, 112, 800, 372]] centre: [460.0, 242.0] centre0: [459.0, 242.0]
Is he drowning: True
Type here to search
78°F
ENG
19:26

RESULTS

DROWNING.mp4

ADVANTAGES AND DISADVANTAGES

ADVANTAGES :

A. It works like an “extra lifeguard” for under the water of swimming pools

B. This project describes the drowning detection system for the prevention of drowning incidents in swimming pools

C. This is project a Accurate Pulse Rate of every individual swimmer is also detected and send as signal to the Life Guard through alert message so it help Life Guard to do earlier prediction of a swimmer pulse rate is reduced or increased By doing this they can get alert in advance and can save more then one person from Drowning

DISADVANTAGES :

A. A limitation of this equipment is that if too many swimmers, theocclusion problem arises.

B. The reflection and refraction of light in air-water interference will affect the image quality, and drowning man

C. This method detected is not easy to distinguish swimmers and divers

D. This system needs constant observation which is the main disadvantage.

CONCLUSION

We provided a method to check human
tracking and semantic event detection within the context of video surveillance system capable of automatically detecting drowning incidents in a swimming pool. In the current work, an effective background detection that incorporates prior knowledge using YOLO algorithm and contour detection enables swimmers to be reliably detected and tracked despite the significant presence of water ripples. The system has been tested on several instances of simulated water conditions such as water reflection, lightening condition and false alarms. Our algorithm was able to detect all the drowning conditions along with the exact position of the drowning person in the swimming pool and had an average detection delay of
1.53 seconds, which is relatively low compared to the needed rescue time for a lifeguard operation. Our results show that the proposed method can be used as a reliablemultimedia video-based surveillance system.

FUTURE SCOPE :

Life safety in water has been a concern for many centuries. Latest technology advancements has enabled us to come up with effective drowning detection systems. However many of those solutions are costly and limited to few. Survey reports show us that highest numbers of deaths are reported in low and middle income countries. The survey report also mentions the children have the largest death ratio compared to adults. Also the deaths reported in these incidents are more from open water bodies than closed water bodies like swimming pools.

The solution described above will be able to address these issues. The swimming goggles with drowning detection unit can be economically viable solution. The range of the alarms transmission can be improved by using underwater acoustics. Any age groups will be comfortable wearing the goggles, without hampering the recreational joy while swimming. The goggles can be useful even in sea. The alarm receivers can be placed at different locations in the water bodies which is having high chance of drowning. Another major advantage ofthis approach unlike other approach is the ease of use in all atmospheric conditions, like rain or wind to day or night. Thissolution is also a reliable solution where the life guards have difficulty to monitor the swimmers like a highly crowded set.

APPENDIX

Source code :

Python :

```
from flask import Flask
from flask import flash, request, redirect, render_template, url_for
from cloudant.client import Cloudant
from cvlib.object_detection import draw_bbox

import time
import os
import cvlib as cv
import cv2
import time
import numpy as np

app = Flask(__name__)
client = Cloudant.iam(
    'ede396d3-a103-4438-9e2d-a90fa381ba44-bluemix',
    '6w0F_DB2tuHNFpl07kPloLvoFYu30NhmbiMCc4jBx_aT',
    connect=True)
db = client['user_details']

@app.after_request
def add_header(r):
    """
    Add headers to both force latest IE rendering engine or Chrome Frame,
    and also to cache the rendered page for 10 minutes.
    """
    r.headers["Cache-Control"] = "no-cache, no-store, must-revalidate"
    r.headers["Pragma"] = "no-cache"
    r.headers["Expires"] = "0"
    r.headers['Cache-Control'] = 'public, max-age=0'
    return r

@app.route("/")
def home():
    return render_template('index.html', title="VirtualEye - Home")

@app.route("/login", methods=["GET", "POST"])
def login():
    if request.method == "POST":
        x = [x for x in request.form.values()]
        data = {
        '_id' : x[0],
```

```
        'psw': x[1]
        }
        query = {'_id': {'$eq' : data['_id']}}
        docs = db.get_query_result(query)
        if len(docs.all()) == 0:
            db.create_document(data)
            return render_template('login.html', title='VirtualEye - Login', status='NR')
        else:
            if x[0] == docs[0][0]['_id'] and x[1] == docs[0][0]['psw']:
                return redirect(url_for('prediction'))
            else:
                return render_template('login.html', title="VirtualEye - Login", status="Failed")
    return render_template('login.html', title="VirtualEye - Login")


@app.route("/register", methods=['GET', 'POST'])
def register():
    if request.method == "POST":
        x = [x for x in request.form.values()]
        data = {
        '_id' : x[1],
        'name': x[0],
        'psw': x[2]
        }
        query = {'_id': {'$eq' : data['_id']}}
        docs = db.get_query_result(query)
        if len(docs.all()) == 0:
            db.create_document(data)
            return render_template('register.html', title='VirtualEye - Register', status='Success')
        else:
            return render_template('register.html', title='VirtualEye - Register', status='Failed')
    return render_template('register.html', title='VirtualEye - Register')

@app.route("/demo", methods=['GET'])
def demo():
    return render_template('base.html', title="VirtualEye - Demo")

@app.route("/forgotpassword")
def forgotpass():
    return render_template('base.html', title="VirtualEye")

@app.route("/logout")
def logout():
    return render_template('logout.html', title="VirtualEye - Logged out")

@app.route('/result')
def prediction():
    webcam = cv2.VideoCapture('drowning.mp4')

    if not webcam.isOpened():
        flash("Could not open webcam")
```

```
        exit()


    t0 = time.time()
    centre0 = np.zeros(2)
    isDrowning = False

    while webcam.isOpened():
        status, frame = webcam.read()

        bbox, label, conf = cv.detect_common_objects(frame)

        if len(bbox) > 0:
            centre = [0, 0]
            centre = [(bbox[0][0] + bbox[0][2]) / 2, (bbox[0][1] + bbox[0][3]) / 2]
            hmov = abs(centre[0] - centre0[0])
            vmov = abs(centre[1] - centre0[1])

            x = time.time()
            threshold = 10

            if hmov > threshold or vmov > threshold:
                print(x - t0, 's')
                t0 = time.time()
                isDrowning = False
            else:
                print(x - t0, 's')
                if time.time() - t0 > 10:
                    isDrowning = True

            print('bbox:', bbox, ' center:', centre, ' centre0:', centre0)
            print('Are they drowning: ', isDrowning)

            centre0 = centre

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

        cv2.imshow("Real-time object detection", out)

        if isDrowning:
            os.system("mpg123 -q alarm.mp3")
            webcam.release()
            cv2.destroyAllWindows()
            return render_template('prediction.html', prediction="Emergency!!! The Person is
drowning", title="VirtualEye - Prediction")

        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    webcam.release()
    cv2.destroyAllWindows()
```

```
    return render_template('prediction.html', title='VirtualEye - Prediction', prediction='Waiting for
footage')


if __name__ == '__main__':
    app.run(debug=True)
```

init.py:

```
from .face_detection import detect_face
from .object_detection import detect_common_objects
from .utils import get_frames, animate
```

face_detection:

```
# import necessary packages
import cv2
import numpy as np
import os
from pkg_resources import resource_filename, Requirement

is_initialized = False
prototxt = None
caffemodel = None
net = None

def detect_face(image, threshold=0.5, enable_gpu=False):

    if image is None:
        return None

    global is_initialized
    global prototxt
    global caffemodel
    global net

    if not is_initialized:

        # access resource files inside package
        prototxt = resource_filename(Requirement.parse('cvlib'),
                                                    'cvlib' + os.path.sep +
'data' + os.path.sep + 'deploy.prototxt')
        caffemodel = resource_filename(Requirement.parse('cvlib'),
                                                'cvlib' + os.path.sep + 'data'
+ os.path.sep + 'res10_300x300_ssd_iter_140000.caffemodel')

        # read pre-trained wieights
```

```
  net = cv2.dnn.readNetFromCaffe(prototxt, caffemode
l)

        is_initialized = True

    # enable GPU if requested
    if enable_gpu:
        net.setPreferableBackend(cv2.dnn.DNN_BACKEND_CUDA)
        net.setPreferableTarget(cv2.dnn.DNN_TARGET_CUDA)

    (h, w) = image.shape[:2]

    # preprocessing input image
    blob = cv2.dnn.blobFromImage(image, 1.0, (300,300), (104.0,177.0,123.0))
    net.setInput(blob)

    # apply face detection
    detections = net.forward()

    faces = []
    confidences = []

    # loop through detected faces
    for i in range(0, detections.shape[2]):
        conf = detections[0,0,i,2]

        # ignore detections with low confidence
        if conf < threshold:
            continue

        # get corner points of face rectangle
        box = detections[0,0,i,3:7] * np.array([w,h,w,h])
        (startX, startY, endX, endY) = box.astype('int')

        faces.append([startX, startY, endX, endY])
        confidences.append(conf)

    # return all detected faces and
    # corresponding confidences
    return faces, confidences
```

object detection.py :

```
import cv2
import os
import numpy as np
from .utils import download_file

initialize = True
net = None
dest_dir = os.path.expanduser('./cvlib') + os.path.sep + 'yolo' + os.path.sep +
'yolov3'
classes = None
COLORS = np.random.uniform(0, 255, size=(80, 3))
```

```
def populate_class_labels():

    class_file_name = 'yolov3_classes.txt'
    class_file_abs_path = dest_dir + os.path.sep + class_file_name
    url = 'https://github.com/arunponnusamy/object-detection-
opencv/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):

    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, colors=None, write_conf=False):
    """A method to apply a box to the image
    Args:
        img: An image in the form of a numPy array
        bbox: An array of bounding boxes
        labels: An array of labels
        colors: An array of colours the length of the number of targets(80)
        write_conf: An option to write the confidences to the image
    """


    global COLORS
    global classes

    if classes is None:
        classes = populate_class_labels()

    for i, label in enumerate(labels):

        if colors is None:
            color = COLORS[classes.index(label)]
        else:
            color = colors[classes.index(label)]

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

        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, model='yolov4',
enable_gpu=False):
    """A method to detect common objects
```

```
    Args:
        image: A colour image in a numpy array
        confidence: A value to filter out objects recognised to a lower confidence
score
        nms_thresh: An NMS value
        model: The detection model to be used, supported models are: yolov3, yolov3-
tiny, yolov4, yolov4-tiny
        enable_gpu: A boolean to set whether the GPU will be used

    """

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

    global classes
    global dest_dir


    if model == 'yolov3-tiny':
        config_file_name = 'yolov3-tiny.cfg'
        cfg_url = "https://github.com/pjreddie/darknet/raw/master/cfg/yolov3-tiny.cfg"
        weights_file_name = 'yolov3-tiny.weights'
        weights_url = 'https://pjreddie.com/media/files/yolov3-tiny.weights'
        blob = cv2.dnn.blobFromImage(image, scale, (416,416), (0,0,0), True,
crop=False)

    elif model == 'yolov4':
        config_file_name = 'yolov4.cfg'
        cfg_url =
'https://raw.githubusercontent.com/AlexeyAB/darknet/master/cfg/yolov4.cfg'
        weights_file_name = 'yolov4.weights'
        weights_url =
'https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolov4.w
eights'
        blob = cv2.dnn.blobFromImage(image, scale, (416,416), (0,0,0), True,
crop=False)

    elif model == 'yolov4-tiny':
        config_file_name = 'yolov4-tiny.cfg'
        cfg_url =
'https://raw.githubusercontent.com/AlexeyAB/darknet/master/cfg/yolov4-tiny.cfg'
        weights_file_name = 'yolov4-tiny.weights'
        weights_url =
'https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v4_pre/yolov4-
tiny.weights'
        blob = cv2.dnn.blobFromImage(image, scale, (416,416), (0,0,0), True,
crop=False)

    else:
        config_file_name = 'yolov3.cfg'
        cfg_url = 'https://github.com/arunponnusamy/object-detection-
opencv/raw/master/yolov3.cfg'
        weights_file_name = 'yolov3.weights'
        weights_url = 'https://pjreddie.com/media/files/yolov3.weights'
        blob = cv2.dnn.blobFromImage(image, scale, (416,416), (0,0,0), True,
crop=False)

    config_file_abs_path = dest_dir + os.path.sep + config_file_name
    weights_file_abs_path = dest_dir + os.path.sep + weights_file_name
```

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

if not os.path.exists(weights_file_abs_path):
    download_file(url=weights_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

# enables opencv dnn module to use CUDA on Nvidia card instead of cpu
if enable_gpu:

    net.setPreferableBackend(cv2.dnn.DNN_BACKEND_CUDA)
    net.setPreferableTarget(cv2.dnn.DNN_TARGET_CUDA)

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:
    box = boxes[i]
    x = box[0]
    y = box[1]
    w = box[2]
    h = box[3]
    bbox.append([int(x), int(y), int(x+w), int(y+h)])
    label.append(str(classes[class_ids[i]]))
```

```
        conf.append(confidences[i])

    return bbox, label, conf


class YOLO:

    def __init__(self, weights, config, labels, version='yolov3'):

        print('[INFO] Initializing YOLO ..')

        self.config = config
        self.weights = weights
        self.version = version

        with open(labels, 'r') as f:
            self.labels = [line.strip() for line in f.readlines()]

        self.colors = np.random.uniform(0, 255, size=(len(self.labels), 3))

        self.net = cv2.dnn.readNet(self.weights, self.config)

        layer_names = self.net.getLayerNames()

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


    def detect_objects(self, image, confidence=0.5, nms_thresh=0.3,
                       enable_gpu=False):

        if enable_gpu:
            net.setPreferableBackend(cv2.dnn.DNN_BACKEND_CUDA)
            net.setPreferableTarget(cv2.dnn.DNN_TARGET_CUDA)

        Height, Width = image.shape[:2]

        scale = 0.00392

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

        self.net.setInput(blob)

        outs = self.net.forward(self.output_layers)

        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:
            box = boxes[i]
            x = box[0]
            y = box[1]
            w = box[2]
            h = box[3]
            bbox.append([int(x), int(y), int(x+w), int(y+h)])
            label.append(str(self.labels[class_ids[i]]))
            conf.append(confidences[i])

        return bbox, label, conf


    def draw_bbox(self, img, bbox, labels, confidence, colors=None, write_conf=False):

        if colors is None:
            colors = self.colors

        for i, label in enumerate(labels):

            color = colors[self.labels.index(label)]

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

            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)
```

utils.py :

```
import requests
import progressbar as pb
import os
import cv2
import imageio
from imutils import paths
import numpy as np

def download_file(url, file_name, dest_dir):

    if not os.path.exists(dest_dir):
        os.makedirs(dest_dir)

    full_path_to_file = dest_dir + os.path.sep + file_name

    if os.path.exists(dest_dir + os.path.sep + file_name):
        return full_path_to_file


    print("Downloading " + file_name + " from " + url)

    try:
        r = requests.get(url, allow_redirects=True, stream=True)
    except:
        print("Could not establish connection. Download failed")
        return None

    file_size = int(r.headers['Content-Length'])
    chunk_size = 1024
    num_bars = round(file_size / chunk_size)

    bar = pb.ProgressBar(maxval=num_bars).start()

    if r.status_code != requests.codes.ok:
        print("Error occurred while downloading file")
        return None

    count = 0

    with open(full_path_to_file, 'wb') as file:
        for chunk in  r.iter_content(chunk_size=chunk_size):
            file.write(chunk)
            bar.update(count)
            count +=1
```

```
    return full_path_to_file


def get_frames(video_file, save_dir=None, save_prefix='', ext='jpg'):

    video = cv2.VideoCapture(video_file)

    if not video.isOpened():

        print("[ERROR] Could not open video file ", video_file)
        video.release()
        return

    frames = []

    frame_count = 0

    while video.isOpened():

        status, frame = video.read()


        if not status:
            break

        frames.append(frame)

        if save_dir:

            frame_count += 1

            out_file = save_dir + os.path.sep + save_prefix + \
                    'frame_' + str(frame_count) + '.' + ext

            print('[INFO] Writing file to .. ', out_file)

            cv2.imwrite(out_file, frame)

    video.release()

    return frames


def animate(src, gif_name, reshape=None, fps=25):

    if not isinstance(src, list):

        if os.path.isdir(src):
```

```
        src = list(paths.list_images(src))

        for idx, image in enumerate(src):
            src[idx] = cv2.imread(image)


if reshape:

    for idx, image in enumerate(src):
        src[idx] = cv2.resize(image, reshape)

for idx, image in enumerate(src):
        src[idx] = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

src = np.array(src)

imageio.mimsave(gif_name, src, fps=fps)
```

HTML code :

INDEX PAGE :

```
<!doctype html>
<html lang="en">

<head>
 <!-- Required meta tags -->
 <meta charset="utf-8">
 <meta name="viewport" content="width=device-width, initial-scale=1">

 <!-- Tailwind CSS -->
    <link rel="stylesheet" type="text/css" href="{{url_for('static',
filename='css/tailwind.min.css')}}" />
 <title>{{ title }}</title>
</head>

<body>
 <header>
        <nav class="flex items-center justify-between flex-wrap bg-gray-700 p-6">
            <div class="flex items-center flex-shrink-0 text-white mr-6 basis-
4/5">
                <span class="font-semibold text-xl tracking-tight">Virtual
Eye</span>
            </div>
            <div class="w-full block flex-grow lg:flex lg:items-center lg:w-
auto">
                <div class="text-sm lg:flex-grow font-semibold">
                         {% block navbar %}
                         <a href="{{url_for('home')}}" class="block mt-4
lg:inline-block lg:mt-0 text-gray-400 hover:text-white mr-4">
                         Home
                    </a>
                    <a href="{{url_for('register')}}"
                        class="block mt-4 lg:inline-block lg:mt-0 text-gray-400
hover:text-white mr-4">
                        Register
                    </a>
                    <a href="{{url_for('login')}}" class="block mt-4 lg:inline-
block lg:mt-0 text-gray-400 hover:text-white mr-4">
                        Login
                    </a>
                    <a href="{{url_for('demo')}}" class="block mt-4 lg:inline-
block lg:mt-0 text-gray-400 hover:text-white mr-4">
                        Demo
                    </a>
                         {% endblock %}
                </div>
            </div>
        </nav>
    </header>
 {% block main %}
 {% endblock %}
</body>

</html>
```

LOGIN PAGE :

```
{% extends 'base.html' %}

{% block main %}
{% if status == 'NR' %}
<div class="m-4 p-4 text-sm text-yellow-700 bg-yellow-100 rounded-lg dark:bg-
yellow-200 dark:text-yellow-800" role="alert">
    <span class="font-medium">Oops!</span> Looks like you have not registered
yet.
</div>
{% endif %}
{% if status == 'Failed' %}
<div class="m-4 p-4 text-sm text-red-700 bg-red-100 rounded-lg dark:bg-red-200
dark:text-red-800" role="error">
    <span class="font-medium">Oops!</span> Looks like you have entered wrong
password.
</div>
{% endif %}
<div class="md:flex md:justify-center mb-6 mt-24">
    <form class="bg-white shadow-xl rounded px-8 pt-10 pb-8 mb-4" method="post">
        <img class="m-auto w-24 h-24 rounded-full mb-8" src="{{url_for('static',
filename='img/logo.png')}}" alt="Virutal Eye Logo">
        <div class="mb-4">
            <label class="block text-gray-700 text-sm font-bold mb-2"
for="username">
                Email ID
            </label>
            <input
                class="shadow appearance-none border rounded w-full py-2 px-3
text-gray-700 leading-tight focus:outline-none focus:shadow-outline"
                id="username" type="email" name="Email ID" required>
        </div>
        <div class="mb-6">
            <label class="block text-gray-700 text-sm font-bold mb-2"
for="password">
                Password
            </label>
            <input
                class="shadow appearance-none border rounded w-full py-2 px-3
text-gray-700 mb-3 leading-tight focus:outline-none focus:shadow-outline"
                id="password" type="password" name="password" required>
        </div>
        <div class="mt-6">
            <input
                class="bg-blue-500 hover:bg-blue-700 text-white font-bold py-2
px-4 rounded focus:outline-none focus:shadow-outline"
                type="submit" value="Login">
            <a class="inline-block align-baseline font-bold text-sm text-blue-500
hover:text-blue-800"
                href="{{url_for('forgotpass')}}">
                Forgot Password?
            </a>
        </div>
    </form>
</div>
{% endblock %}
```

LOGOUT PAGE :

```
{% extends 'base.html' %}


{% block main %}
<div class="flex-col items-center">
    <p class="font-semibold text-2xl m-8" style="text-align: center">Successfully
Logged Out!</p>
    <p class="font-semibold text-l mb-4 text-green-500" style="text-align:
center">Login for more information</p>
    <button
        class="block m-auto bg-blue-500 hover:bg-blue-700 text-white font-bold py-2
px-4 rounded-full">Login</button>
</div>
{% endblock %}
```

PREDICTION PAGE :

```
{% extends 'base.html' %}

{% block navbar %}
<a href="{{url_for('home')}}" class="block mt-4 lg:inline-block lg:mt-0 text-gray-400
hover:text-white mr-4">
    Home
</a>
<a href="{{url_for('logout')}}"
    class="block mt-4 lg:inline-block lg:mt-0 text-gray-400 hover:text-white mr-4">
    Logout
</a>
{% endblock %}

{% block main %}
<p class="font-bold text-3xl px-8 pt-8 underline" style="text-align:
center">VirtualEye - Life Guard for Swimming
    Pools to Detect Active Drowning</p>
<div class="grid grid-cols-2">
    <div class="flex-col bg-white shadow-xl rounded px-8 pt-10 pb-8 m-8 mr-4">
        <p class="m-auto text-xl font-semibold mb-4">Problem</p>
        <p>
            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="flex-col bg-white shadow-xl rounded px-8 pt-10 pb-8 m-8 ml-4">
        <img class="h-56 m-auto" src="{{url_for('static',
filename='img/background.jpg')}}">
        <button
            class="bg-blue-500 hover:bg-blue-700 text-white font-bold py-2 px-4
rounded-full block m-auto mt-4 mb-4">Click
            me! For a demo</button>
        <p style="text-align: center">{{ Prediction }}</p>
    </div>
</div>
{% endblock %}
```

REGISTER PAGE :

```
{% extends 'base.html' %}

{% block main %}
{% if status == 'Success' %}
<div class="m-4 p-4 text-sm text-green-700 bg-green-100 rounded-lg dark:bg-green-200 dark:text-green-
800" role="alert">
  <span class="font-medium">Success!</span> Your registration was successful.
</div>
{% endif %}
{% if status == 'Failed' %}
<div class="m-4 p-4 text-sm text-yellow-700 bg-yellow-100 rounded-lg dark:bg-yellow-200 dark:text-
yellow-800" role="alert">
  <span class="font-medium">Oops!</span> Looks like you have already registered.
</div>
{% endif %}
<div class="md:flex md:justify-center mb-6 mt-20 flex-wrap">
  <form class="bg-white shadow-xl rounded px-8 pt-10 pb-8 mb-4" method="post">
    <img class="w-24 h-24 m-auto rounded-full mb-8" src="{{url_for('static', filename='img/logo.png')}}"
alt="Virutal Eye Logo">
    <div class="mb-4">
      <label class="block text-gray-700 text-sm font-bold mb-2" for="name">
        Name
      </label>
      <input
        class="shadow appearance-none border rounded w-full py-2 px-3 text-gray-700 leading-tight
focus:outline-none focus:shadow-outline"
        id="name" type="text" name="Name">
    </div>
    <div class="mb-4">
```

```
            <label class="block text-gray-700 text-sm font-bold mb-2" for="username">
                Email ID
            </label>
            <input
                class="shadow appearance-none border rounded w-full py-2 px-3 text-gray-700 leading-tight
focus:outline-none focus:shadow-outline"
                id="username" type="email" name="Email ID">
        </div>
        <div class="mb-6">
            <label class="block text-gray-700 text-sm font-bold mb-2" for="password">
                Password
            </label>
            <input
                class="shadow appearance-none border rounded w-full py-2 px-3 text-gray-700 mb-3 leading-tight
focus:outline-none focus:shadow-outline"
                id="password" type="password" name="Password">
        </div>
        <div class="mt-6">
            <input
                class="bg-blue-500 hover:bg-blue-700 text-white font-bold py-2 px-4 rounded focus:outline-none
focus:shadow-outline"
                type="submit" value="Register">
            <a class="inline-block align-baseline font-bold text-sm text-blue-500 hover:text-blue-800"
                href="{{url_for('login')}}">
                Have login?
            </a>
        </div>
    </form>
</div>
{% endblock %}
```

GITHUB LINK :

https://github.com/IBM-EPBL/IBM-Project-25249-1659956102

REFERENCES:-

[1]Foresti, Gian Luca, Petri Mähönen, and Carlo S. Regazzoni, eds. Multimedia video-based surveillance systems: Requirements, Issues and Solutions. Vol.
573. Springer Science & Business Media, 2012.

[2] Jones, Graeme A., Nikos Paragios, and Carlo S. Regazzoni, eds. Video-based surveillance systems: computer vision and distributed processing. Springer Science & Business Media, 2012.

[3] Conde, Cristina, et al. "HoGG: Gabor and HoG-based human detection for surveillance in non-controlled environments." Neurocomputing 100 (2013): 19-30.

[4] Wang, Xiaogang. "Intelligent multi-camera video surveillance: A review." Pattern recognition letters
34.1 (2013): 3-19.

\[8]Zhang, Chi, Xiaoguang Li, and Fei Lei. "A Novel Camera-Based Drowning Detection Algorithm." Advances in Image and Graphics Technologies. Springer Berlin Heidelberg, 2015. 224-233.

[9] Fei, Lei, Wang Xueli, and Chen Dongsheng. "Drowning Detection Based on Background Subtraction." Embedded Software and Systems, 2009. ICESS'09. International Conference on. IEEE, 2009.

[10] Kharrat, Mohamed, et al. "Near drowning pattern detection using neural network and pressure information measured at swimmer's head level." Proceedings of the Seventh ACM International Conference on Underwater Networks and Systems. ACM, 2012.