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

A Project report submitted in partial fulfilment of 7th semester in degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Team ID: PNT2022TMID49865

JONATHAN SANJEEV PRAKESH	**(950619104028)**
P. BALA KRISHNAN	**(950619104012)**
J. DILAKSHAN	**(950619104017)**
M. MOHAMED IBRAHIM	**(950619104041)**

DEPARTMENT OFCOMPUTER SCIENCE ENGINEERING

EINSTEIN COLLEGE OF ENGINEERING,

TIUNELVELI-627 012

BONAFIDE CERTIFICATE

Certified this Report, for the project **"VIRTUAL EYE - LIFE GUARD FOR SWIMMING POOLS TO DETECT ACTIVE DROWNING"** is the bonafide work of "**JONATHAN SANJEEV PRAKESH** (950619104028), **P. BALA KRISHNAN** (950619104012), **J. DILAKSHAN** (950619104017), **M. MOHAMED IBRAHIM** (950619104041)", who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported here in does not form part of any other thesis or dissertation on the basis of which a degree or award was co-offered on the earlier occasion on this or any other candidate.

SIGNATURE

DR.T. SURESH THANGAKRISHNAN

HEAD OF THE DEPARTMENT

COMPUTER SCIENCE AND ENGINEERING

EINSTEIN COLLEGE OF ENGINEERING

TIRUNELVELI-12

ACKNOWLEDGEMENT

We have successfully completed the project with blessings showered on us by God, the Almighty, a project of this nature needs co-operation and support from many for successful completion.

We express our heartfelt thanks to **Mr. A. MATHIVANAN, BE., M. sc. (Agri)**, Managing Trustee of Einstein college of Engineering, Tirunelveli, for his mortal support and device.

Our thanks to **Prof. A. AMUTHAVANAN, BE., M.S (USA). B.L,** Chairman of our college for making necessary arrangements to do this project.

Our heartly thanks to **Prof. EZHILVANAN, MBA.,** Secretary of our college for making necessary arrangements to do this project.

We wish to express our gratitude to **Dr. VELAYUTHAM, M.E, Ph.D., FIE.** Principal for the support he provided us to carry out this project successfully.

We are very much thankful to **Dr.T. SURESH THANKAKRISHNAN**, Head of the Department, Computer Science Engineering who is always a constant of inspiring us.

We extend our sincere thanks to all the teaching and non-teaching staff members and our family members, friends for their help in completing this project.

ABSTRACT

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.

CHAPTER - 1

1. INTRODUCTION

1.1 PROJECT OVERVIEW

Swimming is one of the best exercises that helps people 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.

1.2 PURPOSE

The system is not designed to replace a lifeguard or other human monitor, but to act as an additional tool. “It helps the lifeguard to detect the underwater situation where they cannot easily observe.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

Safety is paramount in all swimming pools. The current systems expected to address the problem of ensuring safety at swimming pools have significant problems due to their technical aspects, such as underwater cameras and methodological aspects such as the need for human intervention in the rescue mission. The use of an automated visual-based monitoring system can help to reduce drownings and assure pool safety effectively. This study introduces a revolutionary technology that identifies drowning victims in a minimum amount of time and dispatches an automated drone to save them. Using convolutional neural network (CNN) models, it can detect a drowning person in three stages. Whenever such a situation like this is detected, the inflatable tube-mounted self-driven drone will go on a rescue mission, sounding an alarm to inform the nearby lifeguards. The system also keeps an eye out for potentially dangerous actions that could result in drowning. This system's ability to save a drowning victim in under a minute has been demonstrated in prototype experiments' performance evaluations.

2.2 REFERENCES

- Nasrin Salehi, Maryam Keyvanara and Seyed Amirhassan Monadjemmi, University of Isfahan, Faculty of Computer Engineering, Department of Artificial Intelligence, Isfahan, 81744, Iran. (For survey review - 1).

- A Kanchana, Kavya G.R, Kavitha C and Soumyashree V - Students and Salila Hegde - Associate Professor, Department of Electronics and Communication, NIE-IT, Mysuru. (For survey review - 2).

2.3 PROLEM STATEMENT DEFINITION

REVIEW-1:

Title of the Paper:

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

Problem Description:

Safety in swimming pools is a crucial issue. In this paper, a real time drowning detection method based on HSV colour space analysis is presented which uses prior knowledge of the video sequences to set the best values for the colour channels. Our method uses a HSV thresholding mechanism along with Contour detection to detect the region of interest in each frame of video sequences. 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. The presented algorithm for this system is tested on several video sequences recorded in swimming pools in real conditions and the results are of high accuracy with a high capability of tracking individuals in real time. According to the evaluation results, the number of false alarms generated by the system is minimal and the maximum

alarm delay reported by the system is 2.6 sec which can relatively be reliable compared to the acceptable time for rescue and resuscitation.

REVIEW-2:

Title of the Paper:

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

Problem Description:

Swimming pool drowning monitoring system based on video technology is mostly reported in the literature. There are three kinds drowning monitoring system according to the different position of the camera. One is that the camera is mounted on the underwater swimming pool wall, then monitor underwater swimmer status. A limitation of this equipment is that if too many swimmers, the occlusion problem arises. The other is that the camera is mounted upon the water, and monitors the Swimmer posture change. The reflection and refraction of light in air-water interference will affect the image quality, and drowning man feature this method detected is not easy to distinguish swimmers and divers obviously. The third is a combination of the two, underwater camera and aerial camera matched, monitoring the swimmer posture. This system needs constant observation which is the main disadvantage.

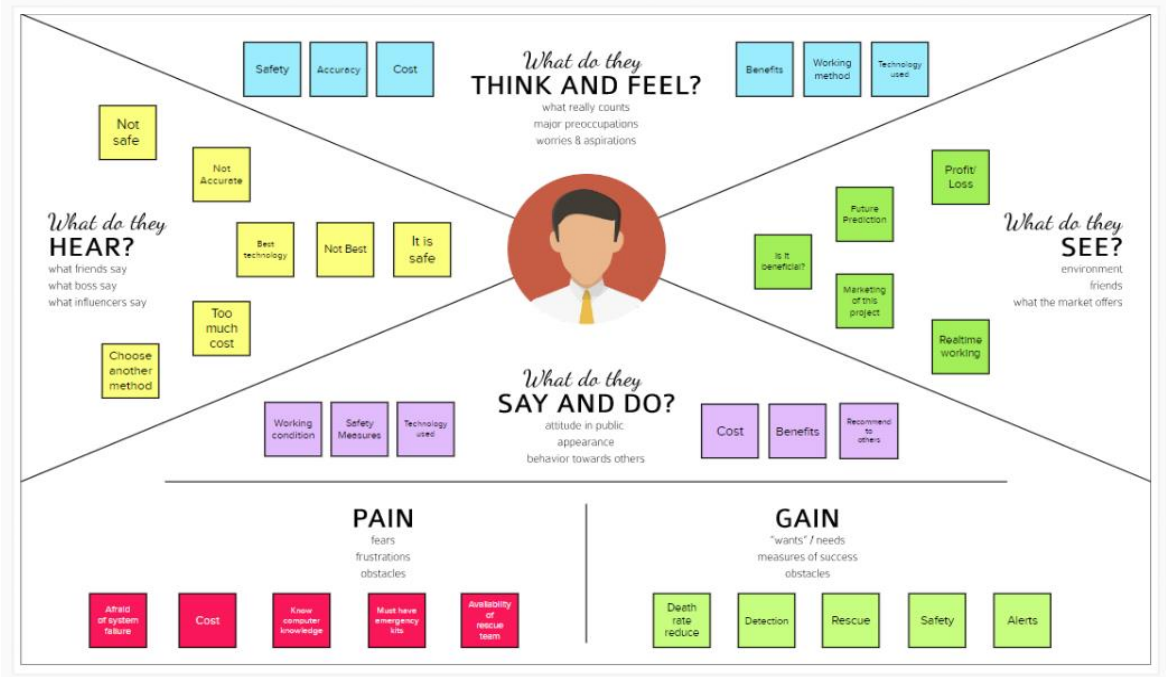
CHAPTER - 3

3.IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS

An empathy map is a simple, easy -to-digest visual that captures knowledge about a user's behaviour and attitude. It is a useful tool to helps teams better understand the users. Creating an effective solution requires understanding the true problem and the person to is experiencing it. The exercise of creating the map helps participants consider thinks from the user's perspective along with his or her goals and challenges.

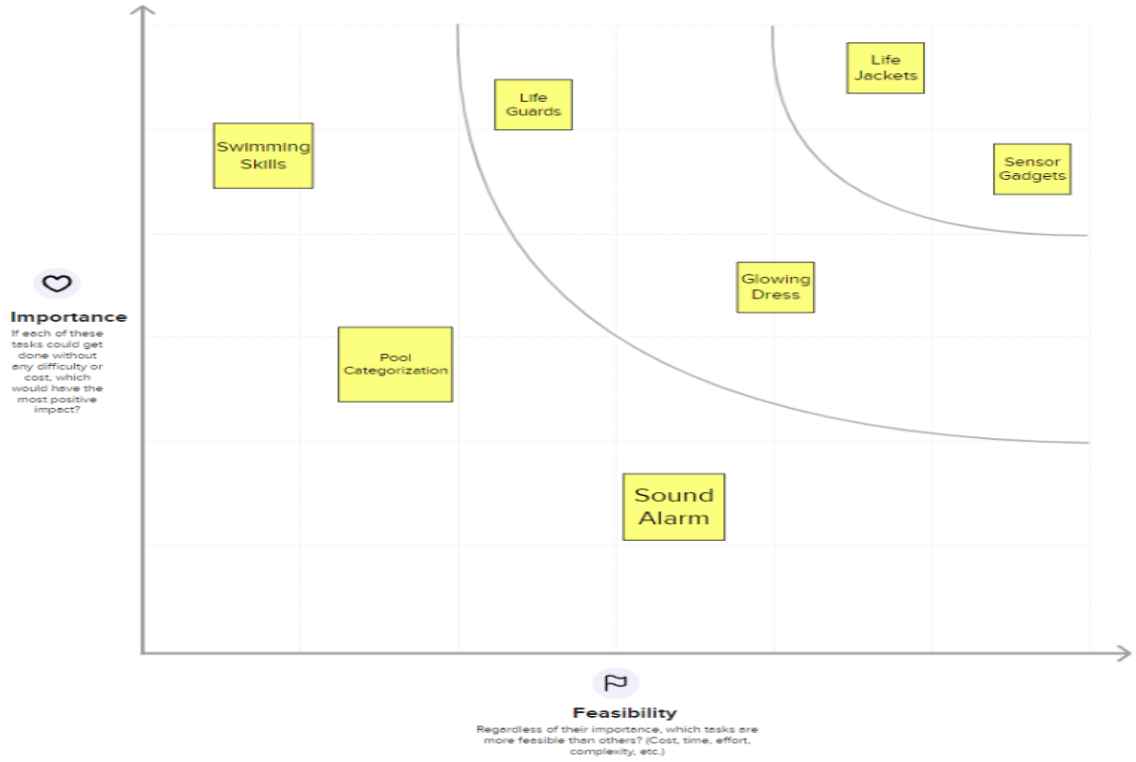
Virtual Eye Empathy map



3.2 IDEATION & BRAINSTORMING

Ideation refers to the whole creative process of coming up with and communicating new ideas. It can take many different forms, from coming up with a totally new idea to combining multiple existing ideas to create a new process or organizational system. ideation is similar to a practice known as brainstorming.

Brainstorming is a group problem-solving method that involves the spontaneous contribution of creative ideas and solutions. This technique requires intensive, freewheeling discussion in which every member of the group is encouraged to think aloud and suggest as many ideas as possible based on their diverse knowledge.



3.3 PROPOSED SOLUTION

Proposed solution should relate the current situation to a desired result and describe the benefits that will accrue when the desired result is achieved. So, begin proposed solution by briefly describing this desired result.

S.No	Parameter	Description
1.	Problem statement (problem to be solved)	Detection of active drowning in swimming pools and alert the lifeguards and to save in the person who is drowning.
2.	Idea/solution description	When the swimmer is drowning the camera deduct it and alert the lifeguards by producing alarm sound.
3.	Novelty/uniqueness	Deep learning for deduction.alarm for alert.automatic life jackets for safety measure.
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 there satisfaction.
5.	Business model (revenue model)	It model can be recommended to other swimming pools if the working of the model leads to the more safety of the customers.
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 life jacket plays a major role in saving the customer from drowning.

3.4 PROBLEM SOLUTION FIT

CHAPTER - 4

4. REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENTS

FR No.	Functional requirement (Epic)	Sub requirement (story/sub-task)
Fr - 1	Authentication	Email id or Username Password
Fr - 2	Confirmation of Authentication	Confirmation via Email Confirmation via OTP
Fr - 3	Authorization levels	User Administrator
Fr - 4	External interfaces	Camera Sound alarm Automatic life jacket Device held by life guards
Fr - 5	Demo	Video Pictures
Fr - 6	Reservation	Rules and regulation Pool for who knows swimming No. of adults No. of children Time period
Fr - 7	Payment	Credit card UPI payment QR scan
Fr - 8	Feedback	Feedback through from

4.2 NON-FUNCTIONAL REQUIREMENT

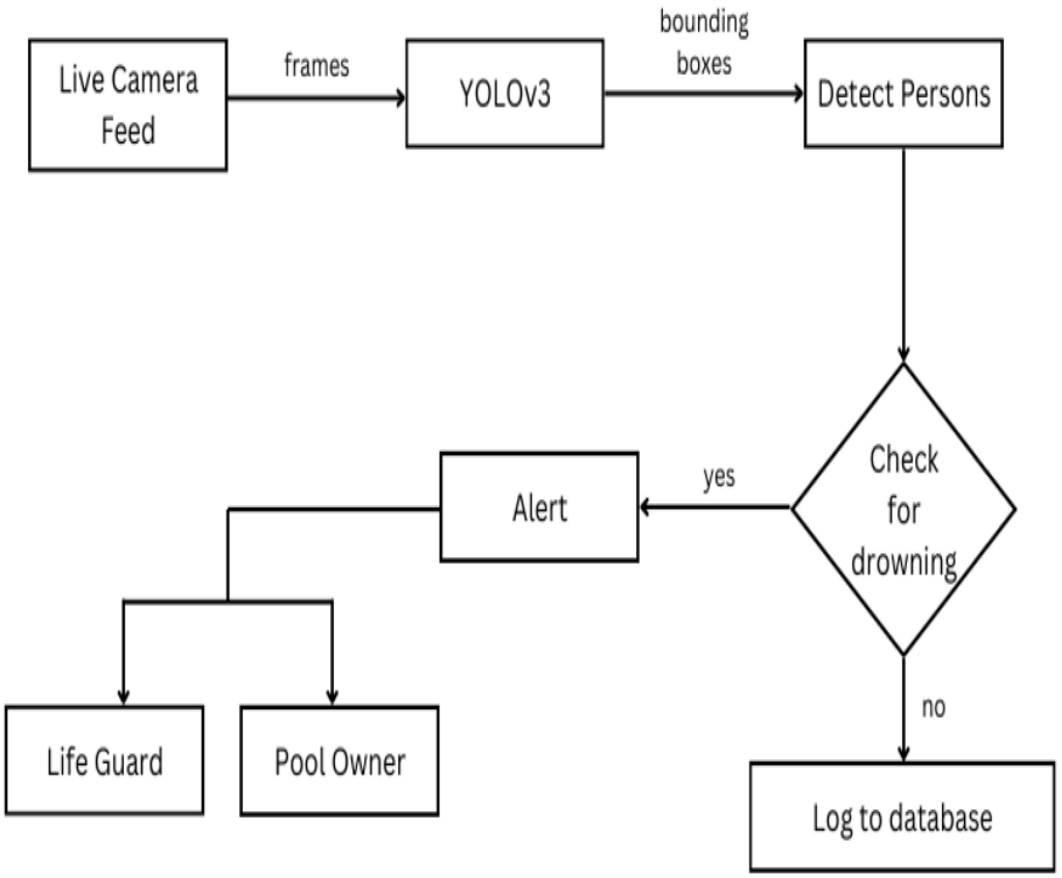
NFR. No.	Non-Functional Requirement	Description
NFR - 1	Usability	Simple and easy to use
NFR - 2	Security	Unauthorized access is denied for security purpose.
NFR - 3	Reliability	After deduction of active drowning, it immediately alerts the life guard by sound alarm and location of drowning person and it also triggers automatic life jacket.
NFR - 4	Performance	If the performance is high, the percentage of saving the drowning person will be high
NFR - 5	Availability	Alarm should be triggered till the drowning person is saved. Registration page should be user friendly and available in all the time emergency kits must be available in all time.
NFR -6	Scalability	Registration can be done in any time. Supporting more users at the same time working of camera, alarm and automatic life jackets should be consistent and fast

CHAPTER - 5

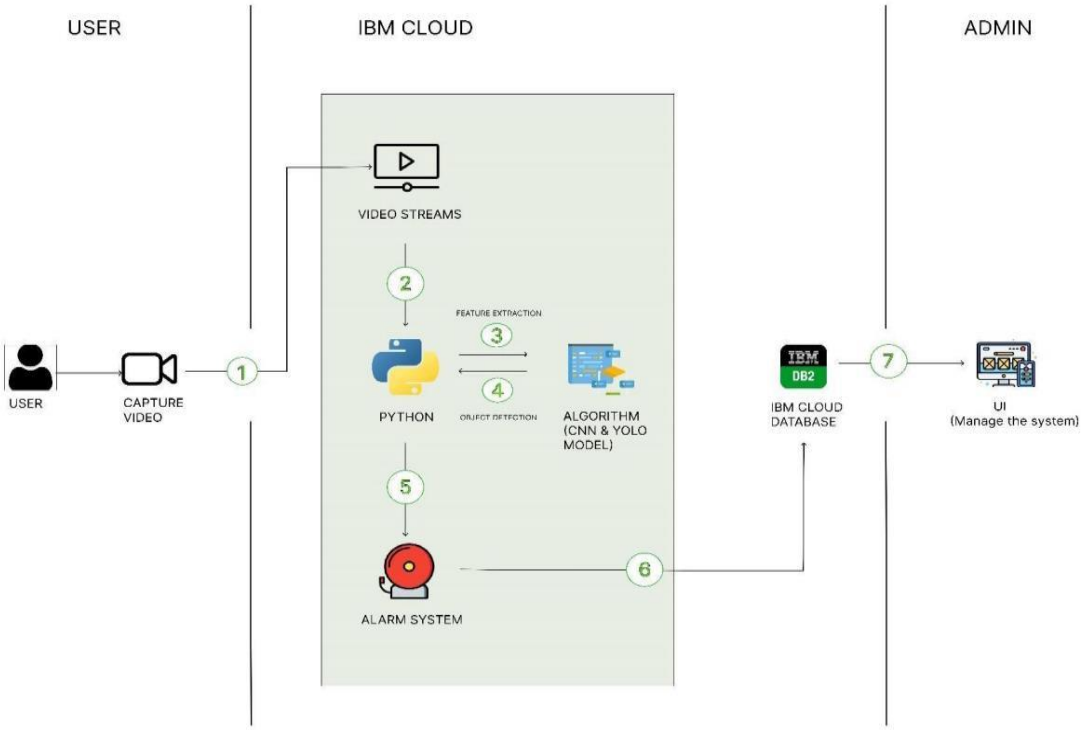
5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAMS

A Data flow diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically.it shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2 TECHNICAL ARCHITECTURE



5.3 USER STORIES

Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Pool owner)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
	Login	USN-3	As a user, I can log into the application by entering email & password	I can login & access the dashboard	Medium	Sprint-1
	Installation	USN-4	As a user, I will install the camera and setup the drowning detection system	I can install the camera and detection system	High	Sprint-1
	Detection	USN-5	As a user, I can detect the drowning person using drowning detection system	I can detect the drowning persons	High	Sprint-1
	Life Guards	USN-6	As an owner, I will appoint life guards for the swimming pool for rescuing drowning person	I will appoint life guards for rescue	High	Sprint-2
	Alert	USN-7	As a user, I will get alert sounds when the swimmer is drowning	I will be alerted if the swimmer is drowning	Medium	Sprint-2
Customer Care Executive	contact	USN-8	As an executor, I can solve technical issues in the system	I can solve technical issues in the system	Medium	Sprint-3
Administrator	Database	USN-9	As an administrator, I can manage the database	I can manage the system logs and other data	High	Sprint-4
		USN-10	As an administrator, I can manage the drowning detection system	I can manage the system logs and other data	High	Sprint-4

CHAPTER - 6

6. PROJECT PLANNING & SCHEDULING

6.1 SPRINT PLANNING & ESTIMATION

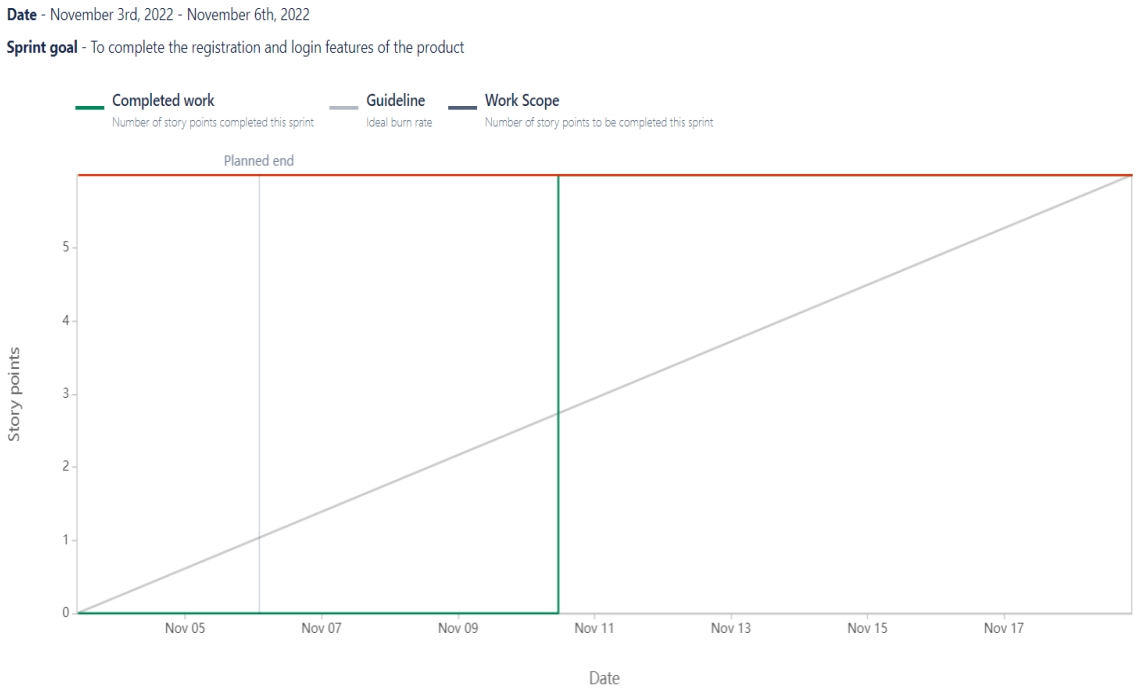
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	3	High	Dilakshan
Sprint-1	Registration	USN-2	As a user, I will receive confirmation email once I have registered for the application	2	High	Bala Krishnan
Sprint-1	Login	USN-3	As a user, I can log into the application by entering email & password	1	Low	Mohamed Ibrahim
Sprint-2	Life Guards	USN-4	As a user, I can add life guards to help and protect the swimmers	3	Medium	Dilakshan
Sprint-2	Detection	USN-5	As a user, I can detect the drowning persons by the detection system software	8	High	Jonathan
Sprint-3	Alert	USN-6	As a user, I will get alert sound if a person is drowning	3	High	Bala Krishnan
Sprint-3	Customer Care	USN-7	As a user, I can solve the problems of the customers if they need help	2	Medium	Bala Krishnan
Sprint-4	Administration	USN-8	As a user, I can monitor the logs and other data	5	High	Mohamed Ibrahim
Sprint-4	Logout	USN-9	As a user, I can logout from the dashboard	1	Low	Jonathan

6.2 SPRINT DELIVERY SCHEDULE

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	6	4 Days	03 Nov 2022	06 Nov 2022	6	06 Nov 2022
Sprint-2	11	5 Days	06 Nov 2022	10 Nov 2022	11	10 Nov 2022
Sprint-3	5	5 Days	10 Nov 2022	14 Nov 2022	5	14 Nov 2022
Sprint-4	6	5 Days	14 Nov 2022	18 Nov 2022	6	18 Nov 2022

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

6.3 REPORTS FROM JIRA

Road Map:

CHAPTER - 7

7. CODING & SOLUTION

7.1 PRE-TRAINED MODEL

Object Detection Model:

```
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 = 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.c
fg'
        weights_file_name = 'yolov4.weights'
        weights_url =
'https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_
optimal/yolov4.weights'
        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)
```

CHAPTER - 8

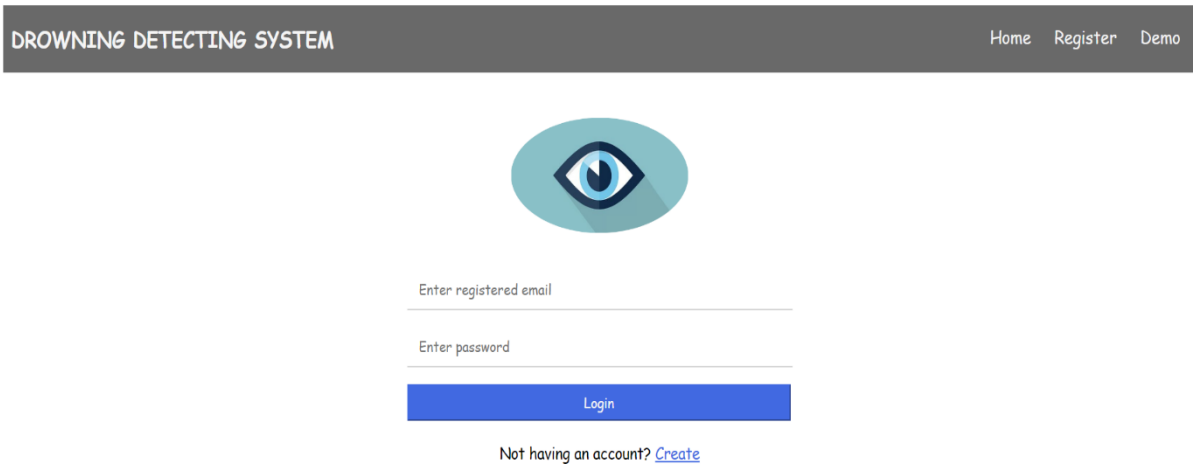
TESTING

8.1 TEST CASES

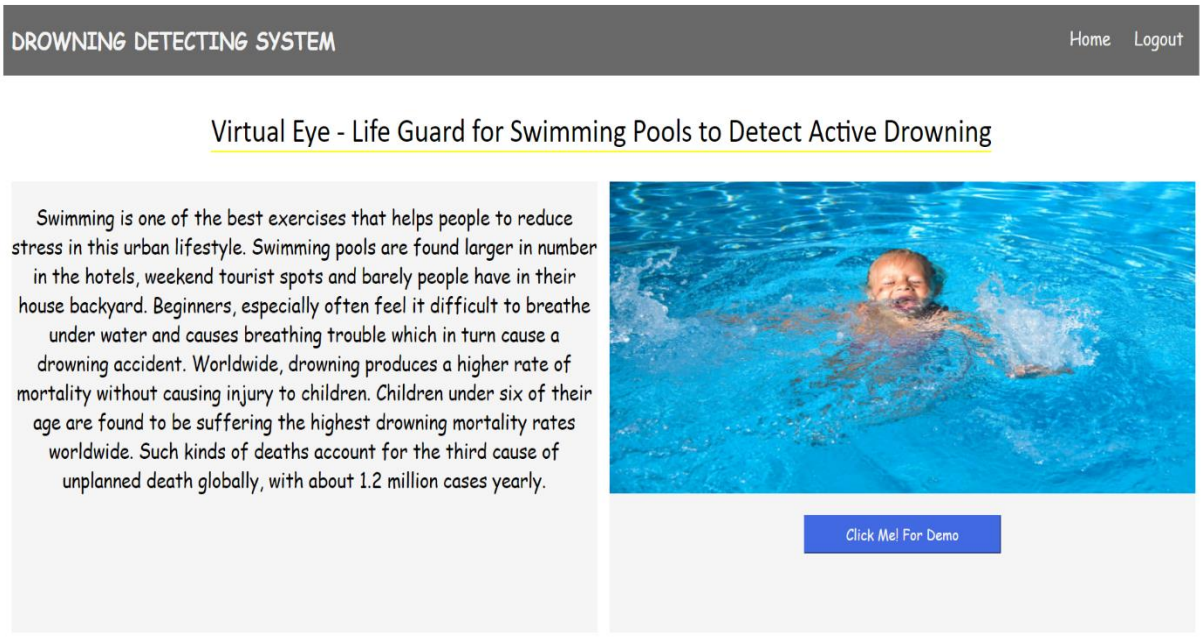
- **Case 1:** Verify user can able to see login page
- **Case 2:** Verify user can able to login to the application
- **Case 3:** Verify user can create or register an account
- **Case 4:** Verify detection of drowning person
- **Case 5:** Verify user can see alert message
- **Case 6:** Verify user can logout

8.2 USER ACCEPTANCE TESTING

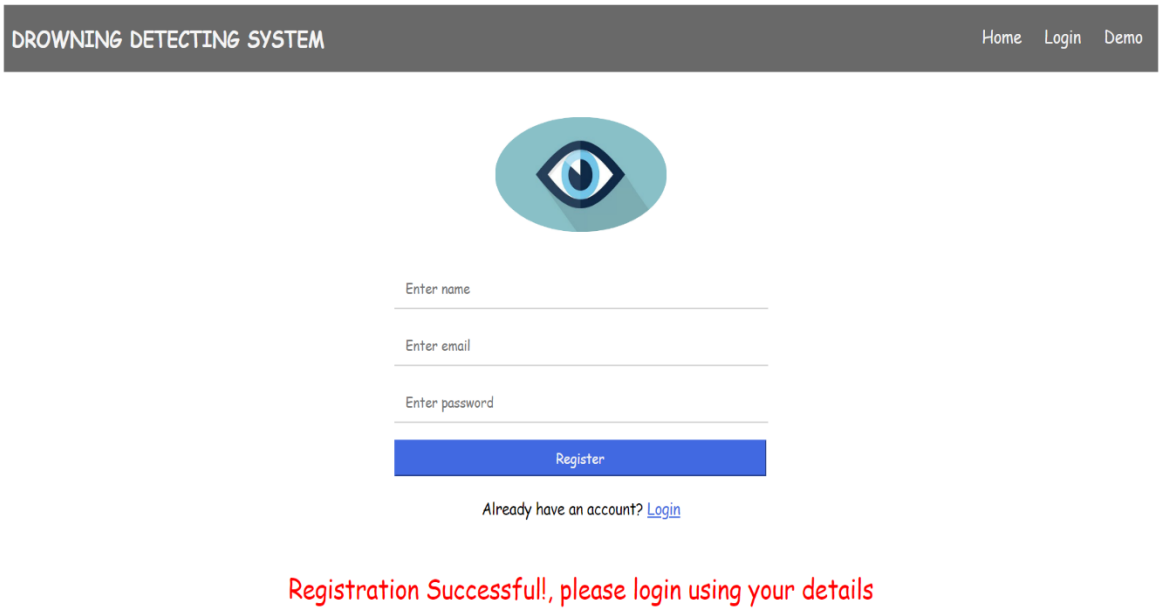
Case 1:



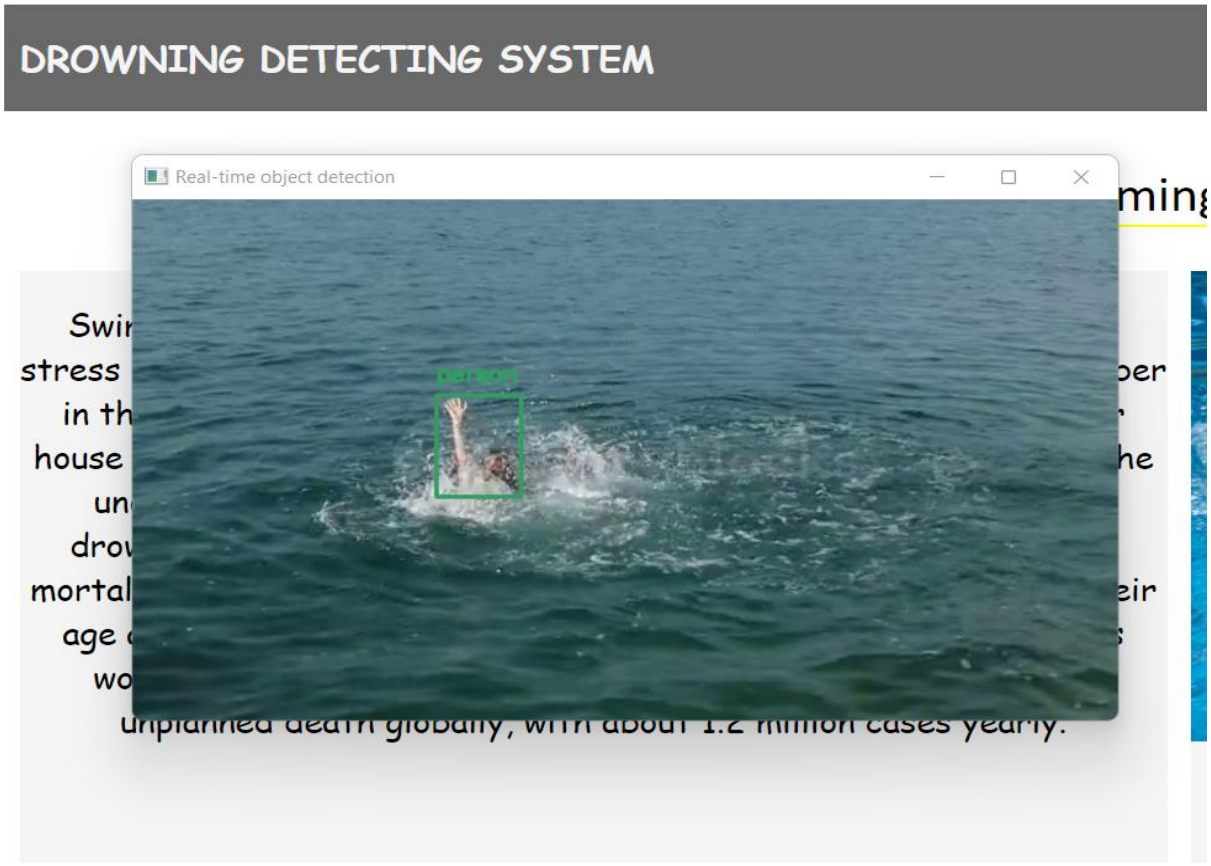
Case 2:


DROWNING DETECTING SYSTEM
Home
Logout
Virtual Eye - Life Guard for Swimming Pools to Detect Active Drowning
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 the hotels, weekend tourist spots and barely people have in their house backyard. Beginners, especially often feel it difficult to breathe under water and causes breathing trouble which in turn cause 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.
Click Me! For Demo

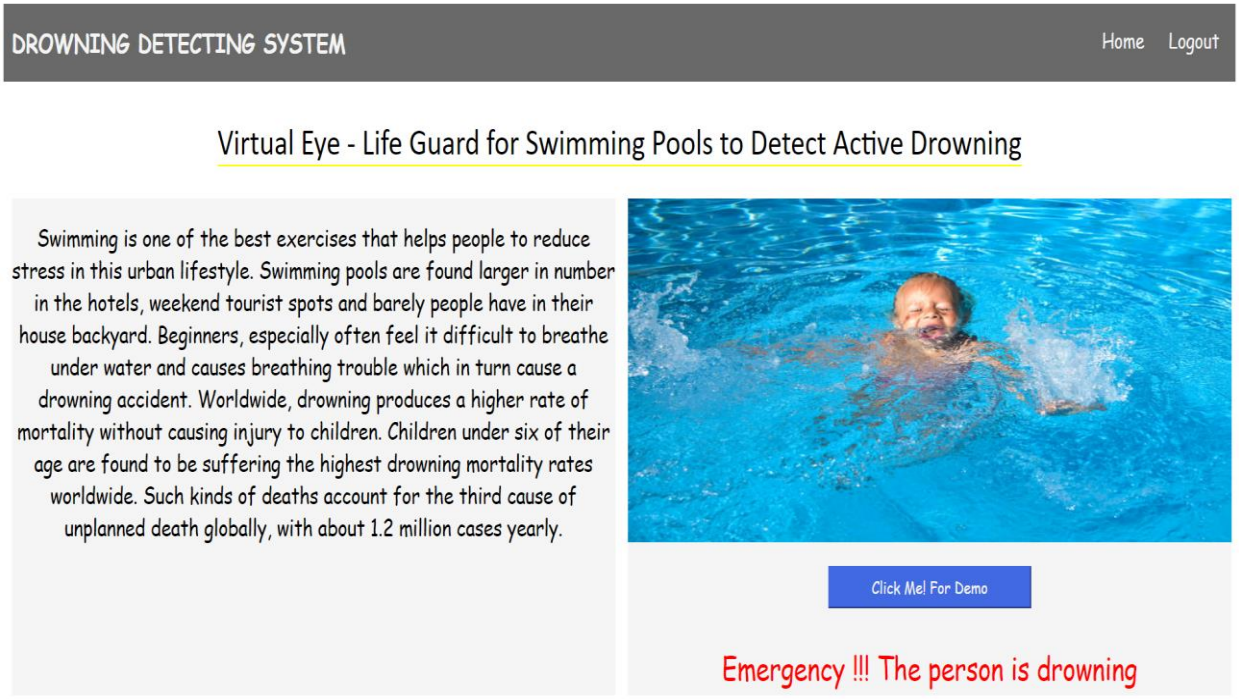
Case 3:


DROWNING DETECTING SYSTEM
Home
Login
Demo
Enter name
Enter email
Enter password
Register
Already have an account? Login
Registration Successful!, please login using your details

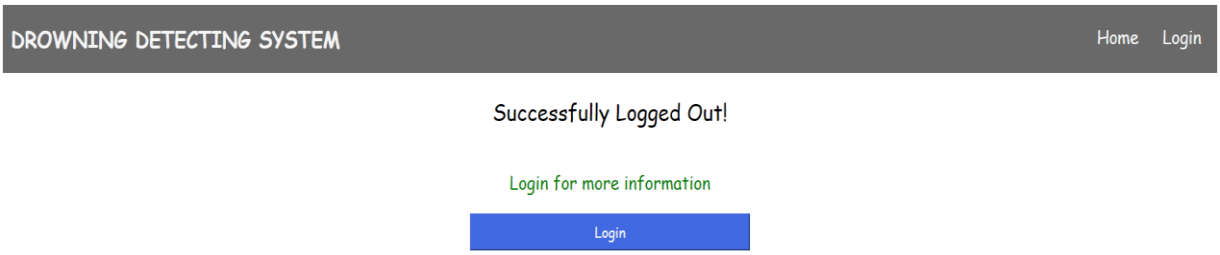
Case 4:


DROWNING DETECTING SYSTEM
Real-time object detection
person

Case 5:


DROWNING DETECTING SYSTEM
Home
Logout
Virtual Eye - Life Guard for Swimming Pools to Detect Active Drowning
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 the hotels, weekend tourist spots and barely people have in their house backyard. Beginners, especially often feel it difficult to breathe under water and causes breathing trouble which in turn cause 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.
Click Me! For Demo
Emergency !!! The person is drowning

Case 6:


DROWNING DETECTING SYSTEM
Home
Login
Successfully Logged Out!
Login for more information
Login

CHAPTER - 9

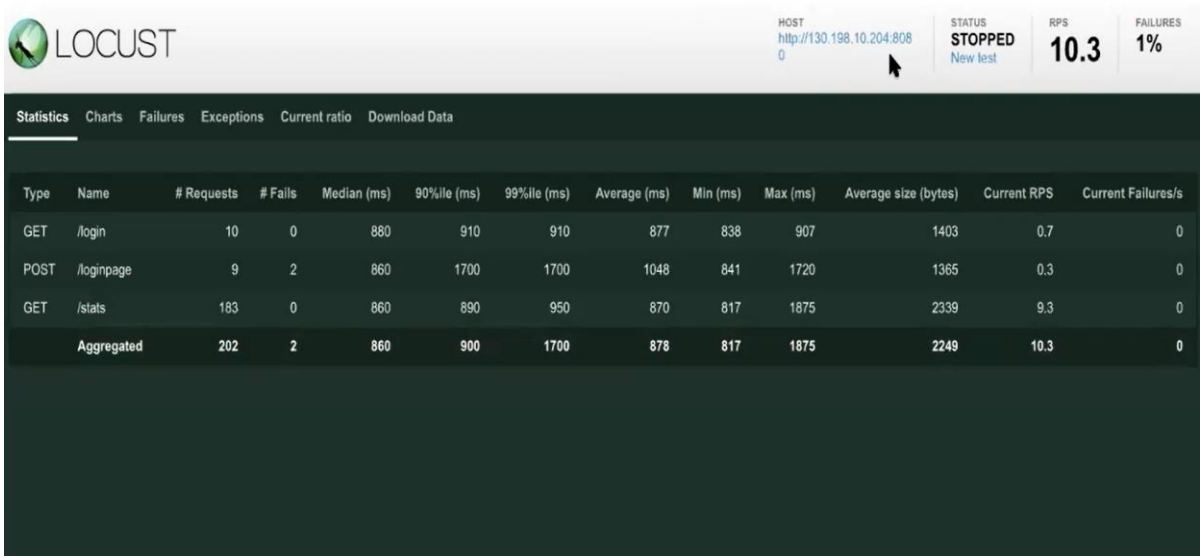
RESULT

9.1 PERFORMANCE MERITES

Performance testing comes under quality assurance checks of a software/application in which speed, capacity and stability are the major checks.

S. No.	Parameter	Values
1.	Model Summary	Object Detection – 95 %
2.	Accuracy	Training Accuracy – 93 % Validation Accuracy – 90 %
3.	Confidence Score (Only Yolo Projects)	Class Detected – 98 % Confidence Score – 94 %

Screenshot:



CHAPTER - 10

ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- It represents an additional level of safety and protection for swimmers.
- Prevents drowning accidents by improving the rescue time of the lifeguards.
- It increases visibility in areas of the pool that are difficult for lifeguards to see, such as the bottom and corners.
- It can be installed in any type of pool and construction variant.

DISADVANTAGES:

- The cost of analysis is very expensive.
- The testing and analysis take some time.

CHAPTER - 11

CONCLUSION

Consistently numerous people, including kids, are suffocated or near suffocating in the deeps of the swimming pools, and the lifeguards are not prepared all around to deal with these issues. In this manner raises the necessities for having a framework that will thus recognize the suffocating people and alert the lifeguards at such hazard. It can be installed in International standardized schools where classes are held for training kids.

CHAPTER - 12

FUTURE SCOPE

Availability of better data set, modern methodologies, and technologies with high computational power accompanied by high-quality surveillance cameras, will help to improve the accuracy of drowning detection & even can be used in adverse conditions. After the implementation of all these essentials, this system also can be used on sea beaches for drowning detection.

CHAPTER – 13

APPENDIX

SOURCE CODE:

TEMPLATES:

login.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=0.1">
<title>login</title>
<link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename = 'index.css') }}">
</head>
<body>
<div class="Navbar">
<div class="head">
<h2>DROWNING DETECTING SYSTEM</h2>
</div>
<div class="options">
<p><a class="a" href="/index">Home</a></p>
<p><a class="a" href="/register">Register</a></p>
<p><a class="a" href="https://youtu.be/4CFhNMzbS24">Demo</a></p>
</div>
</div>
<div class="img">
<div>
<img id="img2" src="{{ url_for('static', filename =
'images/eyelogo.png') }}">
</div><br><br>
<div class="img">
<form action="" method="post">
<input class="inp" type="text" name="email" placeholder="Enter
registered email"><br><br>
<input class="inp" type="varchar" name="pwd" placeholder="Enter
password"><br><br>
<input class="b1" type="submit" name="login" value="Login">
</form>
</div>
<p class="p">Not having an account? <a class="a1"
href="/register">Create</a></p>
{% if pred %}
<p class="p0">{{pred}}</p>
```

```
{% endif %}
</div>
</body>
</html>
```

register.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=0.1">
<title>register</title>
<link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename = 'index.css') }}">
</head>
<body>
<div class="Navbar">
<div class="head">
<h2>DROWNING DETECTING SYSTEM</h2>
</div>
<div class="options">
<p><a class="a" href="/">Home</a></p>
<p><a class="a" href="/login">Login</a></p>
<p><a class="a" href="https://youtu.be/4CFhNMzbS24">Demo</a></p>
</div>
</div>
<div class="img">
<div>
<img id="img2" src="{{ url_for('static', filename =
'images/eyelogo.png') }}">
</div><br><br>
<div class="img">
<form action="" method="post">
<input class="inp" type="text" name="name" placeholder="Enter
name"><br><br>
<input class="inp" type="email" name="email" placeholder="Enter
email"><br><br>
<input class="inp" type="varchar" name="pwd" placeholder="Enter
password"><br><br>
<input class="b1" type="submit" name="register" value="Register">
</form>
</div>
<p class="p">Already have an account? <a class="a1"
href="/login">Login</a></p>
{% if pred %}
<p class="p0">{{pred}}</p>
```

```
{% endif %}
</div>
</body>
</html>
```

prediction.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=0.1">
<title>prediction</title>
<link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename = 'index.css') }}">
</head>
<body>
<div class="Navbar">
<div class="head">
<h2>DROWNING DETECTING SYSTEM</h2>
</div>
<div class="options">
<p><a class="a" href="/index">Home</a></p>
<p><a class="a" href="/logout">Logout</a></p>
</div>
</div>
<div class="img3">
<p class="p3">Virtual Eye - Life Guard for Swimming Pools to Detect
Active Drowning</p>
</div>
<div class="contentbox">
<div class="cbox1">
<p class="p4">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 the hotels, weekend tourist spots and barely people
have in their house backyard. Beginners, especially often feel it
difficult to breathe under water and causes breathing trouble which in
turn cause 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.</p>
</div>
<div class="cbox2">
<img class="image1" src="{{ url_for('static', filename =
'images/drown.jpg') }}"><br><br>
```

```
<div class="b4">
<a href="/result"><input class="b3" type="button" name="demo"
value="Click Me! For Demo"></a>
</div>
{% if pred %}
<p class="p0">{{pred}}</p>
{% endif %}
</div>
</div>
</body>
</html>
```

PYTHON CODE:

```
import cvlib as cv
from cvlib.object_detection import draw_bbox
import cv2
import time
import numpy as np
from playsound import playsound
from flask import Flask, render_template, request

from cloudant.client import Cloudant

client = Cloudant.iam(account_name="eb433409-a69f-477a-9c62-
b277c3cd23dc-
bluemix",api_key="i7viH79onAkYy7kVE9A6ArJtvaXsHLB0brw_ovxJbw1i",connect
= True)

db = client.create_database('my_database')

app = Flask(__name__)

# home page
@app.route('/')
@app.route('/index')
def home():
    return render_template('index.html')

# registration page
@app.route('/register', methods=['POST','GET'])
def register():
    if request.method == 'GET':
        return render_template('register.html')

    name = request.form['name']
    email = request.form['email']
```

```
    password = request.form['pwd']

    data = {
        'email': email,
        'name': name,
        'pwd': password
    }
    print(data)

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

    docs = db.get_query_result(query)
    print(docs)

    print(len(docs.all()))

    if (len(docs.all())==0):
        url = db.create_document(data)

        # response = request.get(url)
        return render_template('register.html', pred =
"Registration  Successful!, please login using your details")
    else:
        return render_template('register.html', pred = "Already
registered!, please login using your details")

# login page
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'GET':
        return render_template('login.html')

    username = request.form.get('email')
    password = request.form.get('pwd')
    print(username, password)

    query = {'email': {'$eq': username}}

    docs = db.get_query_result(query)
    print(docs)

    print(len(docs.all()))

    if(len(docs.all())==0):
        return render_template('login.html', pred = "Invalid
username!")
    elif(username == docs[0][0]['email'] and password ==
docs[0][0]['pwd']):
```

```
        return render_template('prediction.html')
    else:
        return render_template('login.html', pred = "Invalid
Password!")

# logout page
@app.route('/logout')
def logout():
    return render_template('logout.html')

# detection
@app.route('/result', methods=['POST','GET'])
def res():
    webcam =cv2.VideoCapture('drowning.mp4')
    if not webcam.isOpened():
        print("Could not open webcam")
        exit()
    t0 = time.time()

    centre0 = np.zeros(2)
    isDrowning = False

    # loop through frames
    while webcam.isOpened():
        # read frame from webcam
        status, frame = webcam.read()

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

        if (len(bbox)>0):
            bbox0 =bbox[0]

            centre = [0,0]

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

            # vertical and horizontal movement variables
            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
```

```
            if(bbox[0][0]<215 and bbox[0][1]>140 and bbox[0][2]<290 and
bbox[0][3]<210):
                isDrowning = True

            print('bbox:', bbox, 'centre:', centre, 'centre0:',
centre0)
            print('Is he drowning:', isDrowning)

            centre0 = centre

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

        cv2.imshow("Real-time object detection", out)
        if(isDrowning == True):
            playsound('alarm.wav')
            webcam.release()
            cv2.destroyAllWindows()
            return render_template('prediction.html', pred = "Emergency
!!! The person is drowning")

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

    webcam.release()
    cv2.destroyAllWindows()

""" Running the application """
if __name__ == "__main__":
    app.run(debug=True)
```

TEAM ID: PNT2022TMID49865

GITHUB LINK: https://github.com/IBM-EPBL/IBM-Project-47532-1660800090

PROJECT DEMO LINK:

youtube: https://youtu.be/9YDec6hUY84

google drive:

https://drive.google.com/file/d/1vxVZL_Rvfk2nupDW-fymswPpKY9UZA4n/view?usp=sharing