PROJECT REPORT

VIRTUAL EYE-LIFEGUARD FOR SWIMMING POOLS TO DETECT ACTIVE DROWNING

1.INTRODUCTION

1.1 Project Overview

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. 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. 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.

1.2 Purpose

A meticulous system is to be implemented along the swimming pools to save human life. 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.

2.LITERATURE SURVEY

2.1 Existing Problem

Beginners, in particular, frequently struggle to breathe underwater, resulting in respiratory issues and, eventually, a drowning disaster. Drowning causes a higher mortality rate worldwide while causing no harm to children. Children under the age of six are found to have the highest global drowning fatality rates. These types of deaths rank third among all unexpected deaths worldwide, with approximately 1.2 million incident each year.

2.2 Reference

NAME OF THE PAPER: A novel drowning detection method for safety of swimmers

NAME OF THE AUTHOR: Ajil Roy, Dr. K Srinivasan, A.H. Kam J.Wang, Shardul Sanjay Chavan, Sanket Tukaram Dhake, Shubham Virendra Jadhav, Prof. Johnson Mathew.

Literature Survey

S.no	Paper Title	Year of publication	Journal or Conference name	Authors	Theme of the Paper	Inference
1.	An automatic drowning detection surveillance system for challenging outdoor pool environments	2003	Computer Vision, 2003. Proceedings. Ninth IEEE International Conference.	A.H. Kam J.Wang	Automatic drowning detection surveillance system	Understanding Automatic drowning detection.
2.	Drowning Detection System using LRCN Approach	2022	Convergence in Technology Mumbai, India	Shardul Sanjay Chavan, Sanket Tukaram Dhake, Shubham Virendra Jadhav, Prof. Johnson Mathew	Drowning detection using LRCN	Understanding Approach of drowning using LRCN.
3.	A novel drowning detection method for safety of swimmers	2018	Proceedings of the National Power Systems Conference (NPSC) - 2018, December 14-16, NIT Tiruchirappalli, India	Ajil Roy, Dr. K Srinivasan National Institute of Technology Tiruchirappalli, India	Drowning detection for safety of swimmers	Understanding the safety measures provided by drowning detection
4.	Automated drowning detection and security in swimming pool	2017	International Research Journal of Engineering and Technology (IRJET)	A KANCHANA, KAVYA G.R, KAVITHA C, SOUMYASHREE V, SALILA HEGDE	Security in Drowning detection	Understanding the security measures provided by drowning detection.

2.3 Problem Statement Definition

Ideation Phase
Define the Problem Statements

Date	19 September 2022
Team ID	PNT2022TMID32738
Project Name	Virtual Eye – Lifeguard for swimming pools to detect active drowning
Maximum Marks	2 Marks

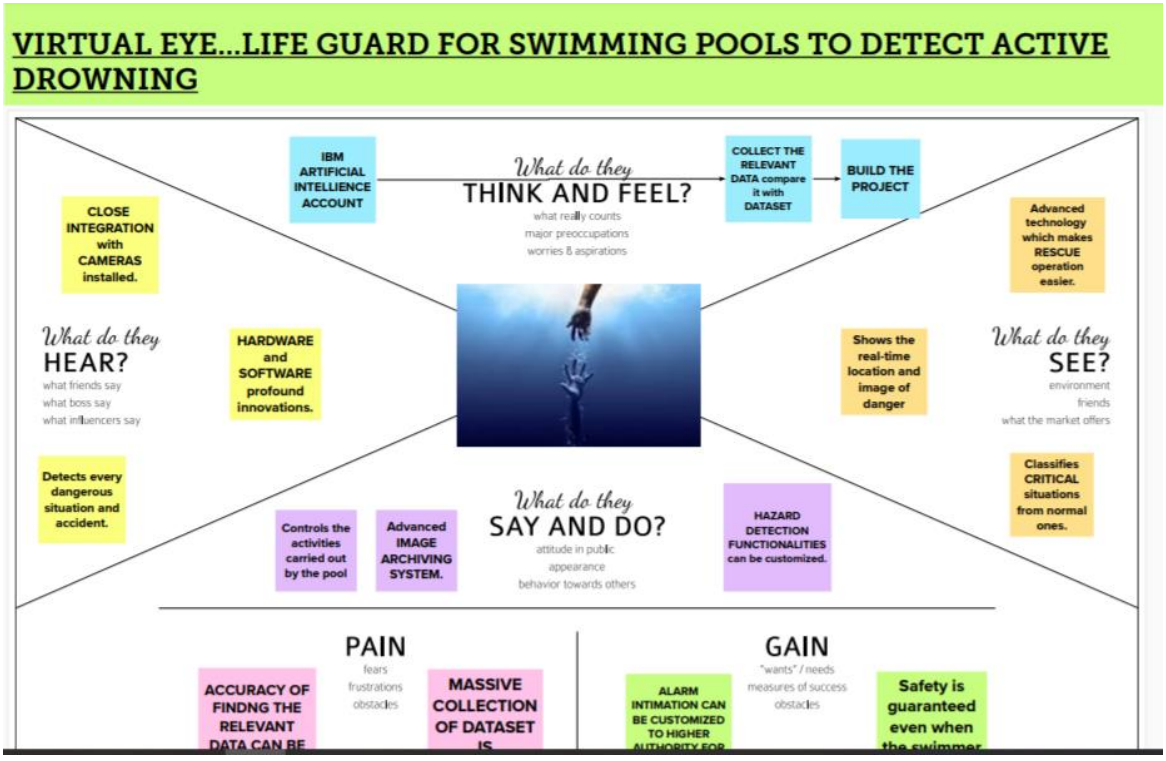
Customer Problem Statement:

Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	Pool house owner	save lives of drowning people	There is no way of detecting and alerting drowning	there is no efficient way to transfer warning signal	helpless
PS-2	Swimming pool attendant	Keep an eye on & save swimmers	It is a difficult task to keep an eye on the swimmers	It is a difficult task to keep an eye on all the at the same time	Under pressure

3.IDEATION AND PROPOSED SOLUTION

3.1 Empathy

Map Canvas An empathy map is a collaborative tool teams can use to gain a deeper insight into their customers. Much like a user persona, an empathy map can represent a group of users, such as a customer segment. The empathy map was originally created by Dave Gray and has gained much popularity within the agile community. An Empathy Map consists of four quadrants. The four quadrants reflect four key traits, which the user demonstrated/possessed during the observation/research stage. The four quadrants refer to what the user: Said, Did, Thought, and Felt.



3.2 Ideation and Brainstorming

Ideation is the process where you generate ideas and solutions through sessions such as Sketching, Prototyping, Brainstorming, Brainwriting, Worst Possible Idea, and a wealth of other ideation techniques. Ideation is also the third stage in the Design Thinking process Brainstorming is a method of generating ideas and sharing knowledge to solve a particular commercial or

technical problem, in which participants are encouraged to think without interruption. Brainstorming is a group activity where each participant shares their ideas as soon as they come to mind.

3.3 Proposed Solution

Your 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 your proposed solution by briefly describing

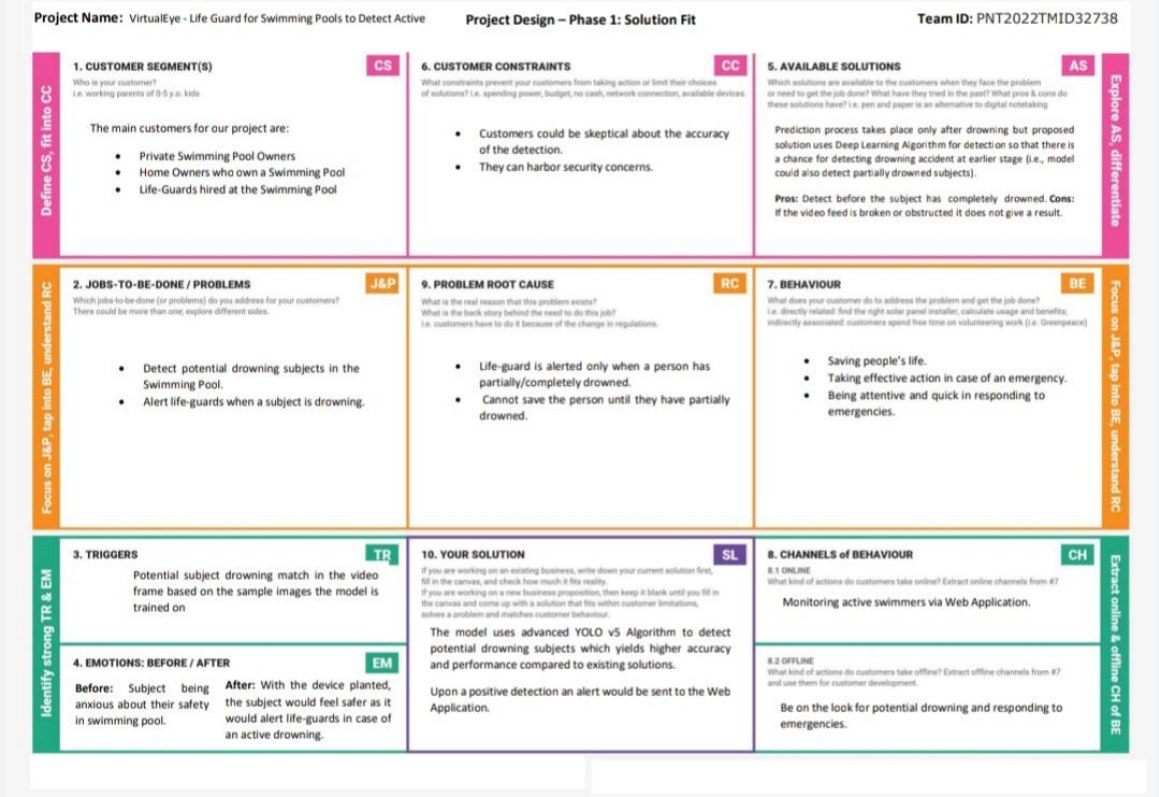
Proposed Solution Template

Date: 29th September 2022
Team ID: PNT2022TMID32738
Project Name: VirtualEye - Life Guard for Swimming Pools to Detect Active Drowning.
Team Leader: Harsheetha RC
Team Members: Harini VS,Harshini G, Jenifar Sheeba A

S. No.	Parameter	Description
1.	Problem Statement	If someone is drowning while swimming in a pool, immediate assistance is required so that the individual can continue swimming without the risk of dying.
2.	Idea / Solution description	We came up with a solution that detects drowning people with help of deep learning and computer vision techniques
3.	Novelty / Uniqueness	The proposed system uses a state-of-the-art object detection model to detect a drowning person in real-time with the highest degree of accuracy.
4.	Social Impact / Customer Satisfaction	This ensures the safety of all swimmers and promotes a safe environment for swimming in swimming pools.
5.	Business Model (Revenue Model)	Subscription model - The subscription business model is a business model in which a customer must pay a recurring price at regular intervals for access to a product or service.
6.	Scalability of the Solution	Since this is a cost-effective model, it can be implemented in all swimming pools.

3.4 Problem Solution Fit

Problem-solution fit is a term used to describe the point validating that the base problem resulting in a business idea really exists and the proposed solution actually solves



4.REQUIREMENT ANALYSIS

4.1 Functional Requirement

Functional requirements are product features or functions that developers must implement to enable users to accomplish their tasks. So, it's important to make them clear both for the development team and the stakeholders. Generally, functional requirements describe system behavior under specific conditions.

Project Design Phase-II
Solution Requirements (Functional & Non-functional)

Date	03 October 2022
Team ID	PNT2022TMID32738
Project Name	Virtual eye – lifeguard for swimming pools for active drowning
Maximum Marks	4 Marks

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Installation	Needed to be fixed under the water in the swimming pool
FR-2	Detection	Either horrified or in unconscious
FR-3	Audio	Ask for help or stay quiet if the person is unconscious
FR-4	Support	Take swim tubes or take the help of rescuer
FR-5	Prior Alert	Send alert message to the lifeguard
FR-6	Pulse rate sensor	Detect the pulse rate of a swimmer

4.2 Non-Functional Requirements

Non functional requirements are requirements that define 'how' the app must perform a certain function. In essence, they are the quality attributes of an app that define the user experience of the app. They are also known as non- behavioral requirements and are to be implemented according to their priority to the app function. This makes them flexible to an extent, making it possible to skip a few in case of time, budget or technology constraints.

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	To ensure the safety of each and every person present in the pool. A Lifeguard should be present all the time in the pool.
NFR-2	**Security**	Lifeguards should be aware of the alert message to save the life of the swimmer
NFR-3	**Reliability**	Virtual eye lifeguard triggers an immediate prior alarm if a swimmer is in peril, helping to avoid panic even in critical situations.
NFR-4	**Performance**	The performance of the tool works better than available tools
NFR-5	**Availability**	Equipment and accessories include lifesaver rings, inflatable vests, life hooks, spine boards, rescue tubes, and a first aid kit.
NFR-6	**Scalability**	Virtual eye lifeguard detects potential drowning and promptly notifies you. It features the latest artificial intelligence technology and adapts to the needs of the user.

5.PROJECT DESIGN

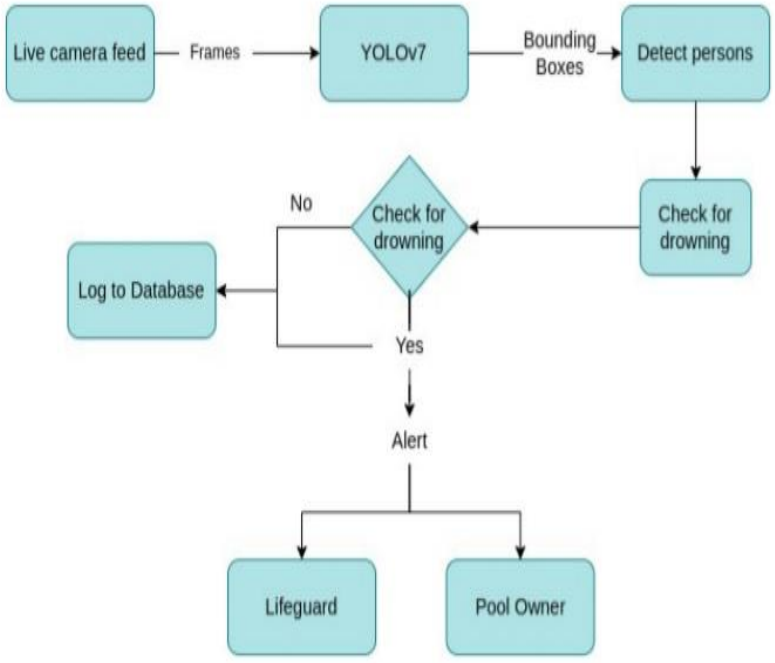
5.1 Data Flow Diagram

Project Design Phase-II
Data Flow Diagram & User Stories

Date	15 October 2022
Team ID	PNT2022TMID32738
Project Name	VirtualEye - LifeGuard for Swimming Pools to Detect Active Drowning
Maximum Marks	4 Marks

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.



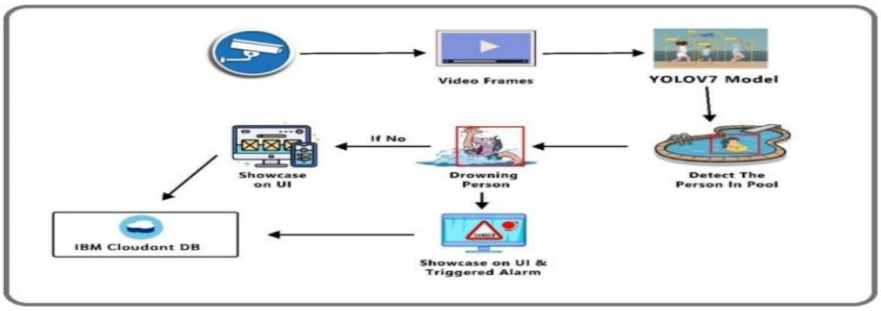
5.2 Solution and Technical Architecture

Project Design Phase – I
Solution Architecture

Date	19 September 2022
Team ID	PNT2022TMID32738
Project Name	VIRTUAL EYE - LIFE GUARD FOR SWIMMING POOLS TO DETECT ACTIVE DROWNING
Maximum Marks	4 Marks

VIRTUAL EYE - LIFE GUARD FOR SWIMMING POOLS TO DETECT ACTIVE DROWNING
Solution Architecture:
Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- **Find the best tech solution to solve existing business problems.**
- **Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.**
- **Define features, development phases, and solution requirements.**
- **Provide specifications according to which the solution is defined, managed, and delivered.**



5.3 User Stories

User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Pool owner)	Installation	USN-1	As a pool owner, I can install the cameras and set up the drowning detection system	I can connect the cameras to the cloud-hosted software	High	Sprint-1
	Detecting the drowning persons	USN-2	As a user, I can find the drowning persons by using the drowning detection system	I would receive an alert if a person is drowning	High	Sprint-1
	Notify the lifeguard	USN-3	As a user, I can notify the lifeguard when the system detects a drowning person	I can set up an alarm that would notify the lifeguard	High	Sprint-2
Customer (Lifeguard)	Rescue people	USN-4	As a user, I can rescue the drowning persons from the pool	I can save the drowning person	High	Sprint-2
Customer (Swimmers)	Safety	USN-5	As a user, I can swim without the fear of drowning	I can swim safely with the help of the system and the lifeguard	Medium	Sprint-2
Customer Care Executive	Contact	USN-6	resolve technical issues	I can contact the customer care executive to resolve any issues	Medium	Sprint-3
Adminitsrator	Dashboard	USN-7	Management of the drowning detection system and database management.	I can access the system's logs and any other data instantly	High	Sprint-4

6.PROJECT PLANNING AND SCHEDULING

6.1 Sprint Planning and Estimation

Project Planning Phase
Project Planning (Product Backlog, Sprint Planning, Stories, Story points)

Date	18 October 2022
Team ID	PNT2022TMID32738
Project Name	Virtual eye - Life Guard for Swimming Pools to Detect Active Drowning
Maximum Marks	8 Marks

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	I can register for the application by entering my phone number.	1	High	Harini VS
		USN-2	I will receive confirmation OTP once I have registered for the application.	2	Low	Jenifar Sheeba A
		USN-3	I can also register for the application through Gmail	2	Medium	Harshini G
	Login	USN-4	I can login into the application by entering email or phone number & password.	1	High	Harsheetha RC
		USN-5	In prediction page, the data uploaded will help the user to detect the drowning movements	2	Medium	Harshini G
Sprint-1	Dataset collection	USN-6	The dataset collected will give high accuracy on the drowning details of the person.	2	High	Harsheetha RC
Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint-2	Data Pre-processing	USN-7	The dataset is extracted and is used to train the model.	4	High	Harini VS
	Train the model	USN-8	We will train the model.	8	High	Jenifar Sheeba A
		USN-9	We will test the model.	6	High	Harshini G
Sprint-3	Detection	USN-10	The tested model will be loaded.	3	High	Harsheetha RC
		USN-11	To identify the person by collecting real-time data.	5	Medium	Harini VS
		USN-12	The data collected at present is checked with the pre-fed data.	8	High	Jenifar Sheeba A
Sprint-4	Alert	USN-13	When the abnormal movement is detected the system will ring an alarm to notify the lifeguard to rescue the person.	7	High	Harshini G
		USN-14	We will be able to detect the drowning person.	3	Medium	Harsheetha RC
Sprint-4	Logout	USN-15	User can logout of the application.	2	Low	Jenifar Sheeba A

Project Tracker, Velocity & Burndown Chart: (4 Marks)

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	10	6 Days	24 Oct 2022	29 Oct 2022		
Sprint-2	18	6 Days	31 Oct 2022	05 Nov 2022		
Sprint-3	16	6 Days	07 Nov 2022	12 Nov 2022		
Sprint-4	12	6 Days	14 Nov 2022	19 Nov 2022		

Velocity:

For Sprint-1 the Average Velocity (AV) is:

AV = Sprint Duration / velocity = 10 / 6 = 1.6

For Sprint-2 the Average Velocity (AV) is:

AV = Sprint Duration / velocity = 18 / 6 = 3.0

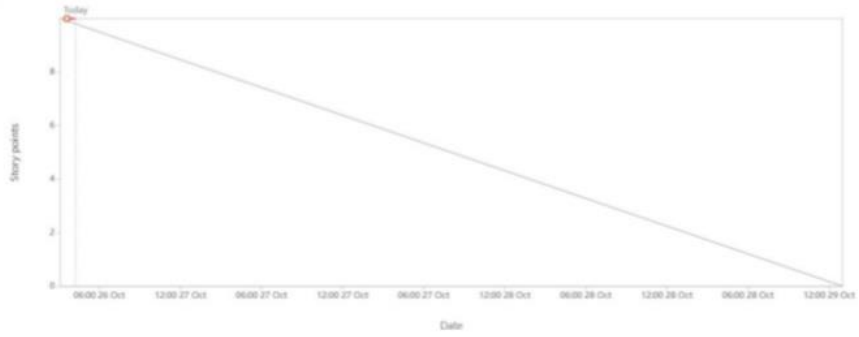
For Sprint-3 the Average Velocity (AV) is:

AV = Sprint Duration / velocity = 16 / 6 = 2.6

For Sprint-4 the Average Velocity (AV) is:

AV = Sprint Duration / velocity = 12/ 6 = 2.0

BURNDOWN CHART



6.2 Sprint Delivery Schedule

Project Planning Phase

Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Date	21 October 2022
Team ID	PNT2022TMID32738
Project Name	VirtualEye - Life Guard for Swimming Pools to Detect Active Drowning
Maximum Marks	8 Marks

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a lifeguard , I can register for the application by entering my email, password, and confirming my password.	2	High	Harini VS
Sprint-1		USN-2	As a lifeguard, I can register for the application through Gmail	1	Medium	Jenifar Sheeba A
Sprint-1	User Confirmation	USN-3	As a lifeguard, I will receive confirmation email once I have registered for the application	1	High	Harshini G
Sprint-1	Login	USN-4	As a lifeguard, I can log into the application by entering email & password	2	High	Harsheetha RC

Project Tracker, Velocity & Burndown Chart: (4 Marks)

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	20	4 Days	24 Oct 2022	27 Oct 2022	20	29 Oct 2022
Sprint-2	20	5 Days	28 Oct 2022	01 Nov 2022	20	04 Nov 2022
Sprint-3	20	8 Days	02 Nov 2022	09 Nov 2022	20	11 Nov 2022
Sprint-4	20	9 Days	10 Nov 2022	18 Nov 2022	20	19 Nov 2022

6.3 Reports from JIRA

Jira helps teams plan, assign, track, report, and manage work and brings teams together for everything from agile software development and customer support to start-ups and enterprises

7.CODING & SOLUTIONING

7.1 Feature 1

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. 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.

Software Requirements:

- tensorflow Keras
- IBM Cloudant Flask
- OpenCVPython
- imutils
- flask
- progress bar
- play sound

Hardware Requirement:

- Processor - Intel core i5
- Hard Disk Space - Minimum 100GB
- RAM - 4GB
- Display -14.1"colour monitor (LCD,CRT or LED)
- Clock Speed - 1.67 GHz

BASE.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" />
    <link rel="stylesheet" href="/static/styles.css" />
    <title>DeepEye</title>
  </head>
  <body>
    <nav>
      <h1>VirtualEye</h1>
      <ul>
        <li><a href="/">Home</a></li>
        {% if request.cookies.get("isLoggedIn") == "True" %}
        <li><a href="/dashboard">Dashboard</a></li>
        <li><a href="/logout">Logout</a></li>
        {% else %}
        <li><a href="/login">Login</a></li>
        <li><a href="/register">Register</a></li>
        {% endif %}
      </ul>
    </nav>
    <hr />
 {% block content %}{% endblock %}

    <footer>VirtualEye &copy; . All rights reserved.</footer>
  </body>
</html>
```

REGISTER.html

```
{% extends "base.html" %} {% block content %}

<section class="cont">
  {% if bad %}
  <div class="message">{{ message }}</div>
  {% else %}
  <div class="message" style="color: green">{{ message }}</div>
```

```
  {% endif %}
  <form class="form" method="post" action="/afterreg">
    <h1 class="head">Register</h1>
    <input type="text" name="name" class="inp" placeholder="username" />
    <input type="email" name="email" class="inp" placeholder="email" />
    <input type="password" name="password" class="inp"
placeholder="password" />
    <input type="submit" class="inp-btn" value="Register" />
  </form>
</section>

{% endblock %}
```

LOGIN.html

```
{% extends "base.html" %} {% block content %}

<section class="cont">
  <div class="message">{{ message }}</div>
  <form class="form" method="post" action="/afterlogin">
    <h1 class="head">Login</h1>
    <input type="email" name="_id" class="inp" placeholder="email" />
    <input type="password" name="psw" class="inp" placeholder="password" />
    <input type="submit" class="inp-btn" value="Login" />
  </form>
</section>

{% endblock %}
```

7.2 Feature 2

To design a system in an economically viable and easily accessible way that acts as a virtual eye to detect the drowning person in the swimming pool and alert the lifeguard using alarms to save the drowning person. The system can be deployed in house,hotels,resorts,and swimmingpoolcenters.The result is predicated in real-time, thus it can be used in emergency situations.

8.TESTING

8.1 Testcase

				Date	09-Nov-22			
				Team ID	PNT2022TMID32738			
				Project Name	Virtual Eye – Life Guard for Swimming Pool to Detect Active Drowning			
				Maximum Mark	4 marks			
Test case ID	**Feature Type**	**Component**	**Test Scenario**		**Steps To Execute**	**Test Data**	**Expected Result**	**Actual Result**
LoginPage_TC_O_001	Functional	Home Page	Verify user is able to see the Login		1.Enter URL and click go 2.Click on My Account dropdown button 3.Verify login/Singup popup displayed or not	login. Html	Login/Signup popup should display	Working as expected
LoginPage_TC_O_002	Functional	Home Page	Verify the UI elements in Login/Signup		1.Enter URL and click go 2.Click on My Account dropdown button 3.Verify login/Singup popup with below UI elements: a.email text box b.password text box c.Login button d.New customer? Create account link e.Last password? Recovery password link	login. Html	Application should show below UI elements: a.email text box b.password text box c.Login button with orange colour d.New customer? Create account link e.Last password? Recovery password link	Working as expected
LoginPage_TC_O_003	Functional	Home page	Verify user is able to log into web		1.Enter URL and click go 2.Click on My Account dropdown button 3.Enter Valid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: hhhj38@gmail password: hhj38	User should navigate to user account homepage	working as expected
LoginPage_TC_O_004	Functional	Login page	Verify user is able to log into web		1.Enter URL and click go 2.Click on My Account dropdown button 3.Enter inValid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: hhhj38@gmail password: hhj38	Application should show 'Incorrect email or password ' validation message.	working as expected
PredictionPage_TC_005	Functional	Prediction Page	Page should display whether the person is drowning or not		1. Camera should take pictures of people swimming in pools 2. It should predict the probability of drowning 3. It should show a bounding box displaying the probability Of drowning	image Of people drowning	Generate a alert to life guard if people are drowning	working as expected

8.2 User Acceptance testing

1.Defect Analysis This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Acceptance Testing
UAT Execution & Report Submission

Date	03 November 2022
Team ID	PNT2022TMID32738
Project Name	Virtual Eye – Life Guard for Swimming Pools to Detect Active Drowning
Maximum Marks	4 Marks

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [ProductName] project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	1	4	2	3	10
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	2	2	1	1	6
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	2	2	1	5
Totals	6	11	10	7	34

2.Test case Analysis

This report shows the nmber of test case that have been passed,failed,untested

3.Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	2	0	0	2
Client Application	2	0	1	1
Security	1	0	0	1
Outsource Shipping	1	0	0	1
Exception Reporting	2	0	1	1
Final Report Output	1	0	0	1
Version Control	1	0	0	1

9.RESULTS

9.1Performance Metrices

Project Development Phase
Model Performance Test

DATE	10 NOV 2022
TEAM ID	PNT2022TMID32738
PROJECT NAME	Virtual Eye – Life Guard for Swimming Pools to Detect Active Drowning
MAXIMUM MARKS	10 MARKS

Model Performance Testing:

Project team shall fill the following information in the model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Model detecting the drowning person	
2.	Accuracy	Training Accuracy - 30 Validation Accuracy - 44	
3.	Confidence Score (Only Yolo Projects)	Class Detected - Yes Confidence Score - 50	

10. ADVANTAGES & DISADVANTAGES

Advantages

- The use of deep learning gives accurate results after training the model.
- YOLOv3 model is fast and can process up to 45 frames per second.
- Recording of events.
- An additional level of security.

Disadvantages

- YOLO has low recall value and struggles to detect very close objects.
- Designed for whom has to guarantee every day the safety in public and intensive use pools, this life guard detects potential drownings and promptly notifies you.
- It features the latest artificial intelligence technology and adapts to the needs of the user.
- It's the ultimate drowning detection system for those who demand the ultimate in safety.

11. CONCLUSION

In this project, we have developed a deep learning system using YOLOv3 model to predict if a person is drowning or not. The system is connected to IBM cloud services the user can access through a web application along with the alarm feature system to notify the lifeguard.

12.FUTURE SCOPE

The project can be further extended by deploying multiple cameras underwater to improve accuracy of prediction. The processing speed of the model can be improved to produce the result faster

13.APPENDIX

Source code

```python
# from crypt import methods
from distutils.log import debug
from email import message
from gzip import BadGzipFile
from itertools import dropwhile
# from signal import alarm
from sqlite3 import connect
import cvlib as cv
from cvlib.object_detection import draw_bbox
import cv2
import time
import numpy as np

from werkzeug.utils import secure_filename
from playsound import playsound

import os
from dotenv import load_dotenv, find_dotenv

# from .utils import download_file

from flask import Flask, request, render_template, redirect, url_for, make_response

from cloudant.client import Cloudant

load_dotenv(find_dotenv())

client = Cloudant.iam(os.getenv("IBM_CLOUDANT_KEY"), os.getenv("IBM_CLOUDANT_USER"), connect=True)

my_database = client.create_database("my_database")

app = Flask(__name__)
```

```python

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


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

@app.route("/prediction")
def prediction():
    if request.cookies.get("isLoggedIn") == "True":
        return render_template("prediction.html")
    else:
        return render_template("login.html", message="You must be logged in first!")

@app.route("/dashboard")
def dashboard():
    if request.cookies.get("isLoggedIn") == "True":
        return render_template("dashboard.html")
    else:
        return render_template("login.html", message="You must be logged in first!")


@app.route('/upload', methods = ['POST'])
def upload_file():
    if request.cookies.get("isLoggedIn") == "True":
        if request.method == 'POST':
            f = request.files['video']
            f.save(os.path.join(os.path.dirname(os.path.abspath(__file__)), 'static/uploads', secure_filename(f.filename)))
            return render_template("prediction.html", message="File upload success, Processing stream...", bad=False)
```

```
            return render_template("prediction.html", message="File upload success, Processing stream...", bad=False)
    else:
        return render_template("login.html", message="You must be logged in first!")

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


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

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

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

    print(len(docs.all()))

    if(len(docs.all()) == 0):
        url = my_database.create_document(data)
        return render_template("register.html", message="Registration Successfull, Please login using your credentials", ba
    else:
        return render_template("register.html", message="You are already a member, please login using your credentials", ba
```

```


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


@app.route("/afterlogin", methods=["POST"])
def afterlogin():

    user = request.form["_id"]
    passw = request.form["psw"]
    print(user, passw)

    query = {"_id": {"$eq": user}}

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

    print(len(docs.all()))

    if(len(docs.all()) == 0):
        resp = make_response(render_template("login.html", message="The email is not found!"))
        return resp
    else:
        if((user == docs[0][0]["_id"]) and passw == docs[0][0]["psw"]):
            resp = make_response(redirect(url_for("dashboard")))
            resp.set_cookie('isLoggedIn',"True")
            return resp
        else:
            print("Invalid User")

```

```python

@app.route("/logout")
def logout():
    if request.cookies.get("isLoggedIn") == "True":
        resp = make_response(render_template("login.html", message="You have logged out successfully!"))
        resp.set_cookie('isLoggedIn', '', expires=0)
        return resp
    else:
        return render_template("login.html", message="You must be logged in first!")

@app.route("/result", methods=["GET", "POST"])
def res():

    if request.cookies.get("isLoggedIn") == "True":

        webcam = cv2.VideoCapture("static/drowning.mp4")

        if not webcam.isOpened():
            print("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):
                bbox0 = bbox[0]
```

```python
                bbox0 = bbox[0]
                centre = [0,0]

                centre = [(bbox0[0]+bbox0[2])/2, (bbox0[1]+bbox0[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, "Centre: ", centre, "Centre0: ", centre0)
                print("Is he drowning: ", isDrowning)

                centre0 = centre


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

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

```
                print("Is he drowning: ", isDrowning)

                centre0 = centre


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

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

                if(isDrowning == True):
                    playsound("http://localhost:5000/static/sound3.mp3")

                    webcam.release()
                    cv2.destroyAllWindows()

                    return render_template("prediction.html", message="Emergency!!! The Person is Drowning", bad=True)

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

        webcam.release()
        cv2.destroyAllWindows()

        return render_template("prediction.html")
    else:
        return render_template("login.html", message="You must be logged in first!")

@app.route("/result-upload", methods=["GET", "POST"])
def resUpload():

    if request.cookies.get("isLoggedIn") == "True":
```

```
        # print(request.files["video"])

        # file = request.files['video']
        # file.save(secure_filename(file.filename))


        #webcam = cv2.VideoCapture("static/uploads/"+ file.filename +".mp4")
        webcam = cv2.VideoCapture("static/uploads/drowning.mp4")

        if not webcam.isOpened():
            print("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):
                bbox0 = bbox[0]
                centre = [0,0]

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

                hmov = abs(centre[0]-centre0[0])
                vmov = abs(centre[1]-centre0[1])

                x = time.time()
```

```
                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, "Centre: ", centre, "Centre0: ", centre0)
                print("Is he drowning: ", isDrowning)

                centre0 = centre


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

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

                if(isDrowning == True):

                    webcam.release()
                    cv2.destroyAllWindows()

                    playsound("http://localhost:5000/static/sound3.mp3")

                    return render_template("prediction.html", message="Emergency!!! The Person is Drowning")

                if(cv2.waitKey(1) & 0xFF ==  ord("q")):
```

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

                    playsound("http://localhost:5000/static/sound3.mp3")

                    return render_template("prediction.html", message="Emergency!!! The Person is Drowning")

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

        webcam.release()
        cv2.destroyAllWindows()

        return render_template("prediction.html")
    else:
        return render_template("login.html", message="You must be logged in first!")


if __name__ == '__main__':
    app.run(debug=True, static_url_path="static", static_folder='static', template_folder="templates")
```

GitHub & Project Demo Link

Github link-https://github.com/IBM-EPBL/IBM-Project-14619-1659587862

Demo link - https://youtu.be/ffMAK0K0D4A