FINAL DELIVERABLE PROJECTDOCUMENTATION

Date	11 November 2022
Team ID	PNT2022TMID30844
Project Name	VirtualEye-Lifeguard for Swimming Pools to Detect the Active Drowning

Submitted by

Team Leader : MURALITHARAN Y

Team Member : AATHITHYAN J

Team Member : JAYASURYA J

Team Member : MANOJ M

Team Member : SURIYA S

In partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

COMPUTER SCIENCE AND ENGINEERING

ANNA UNIVERSITY:CHENNAI 600025

CHAPTER-1

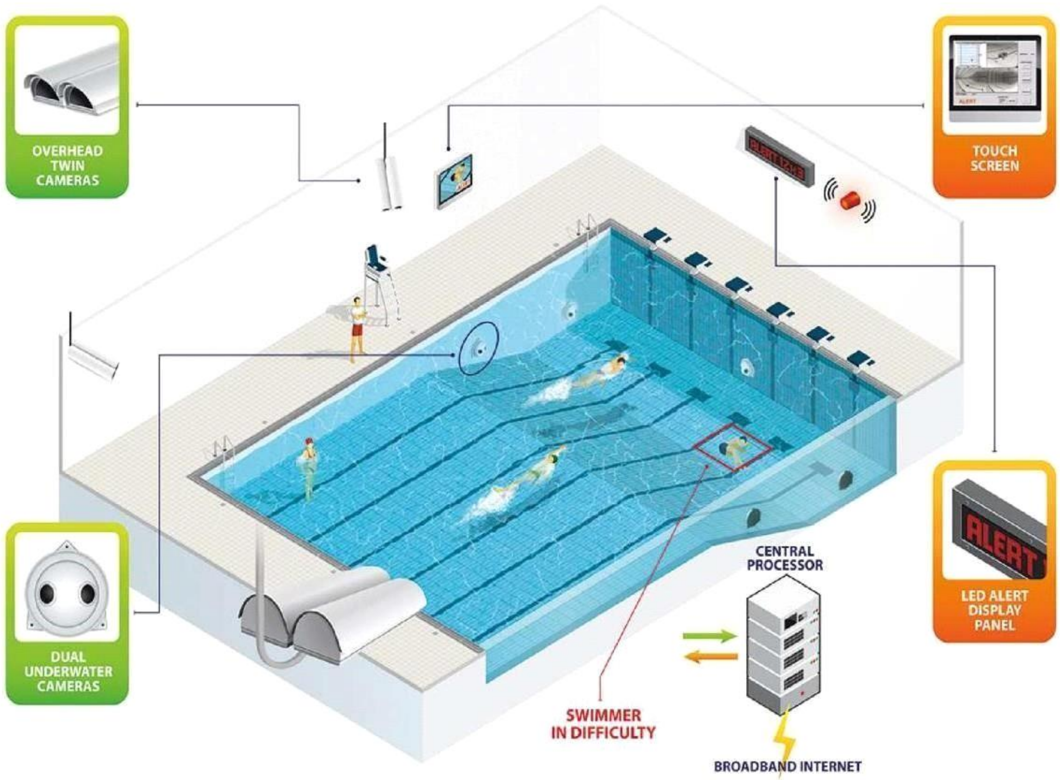
INTRODUCTION

Recently, there has been growing interest around the topic of drowning detection systems (DDS) in the sport and leisure industry both across the UK and globally. Advancements in technology, coupled with the importance of pool safety, has led to its growing prominence, with mention of DDS now in documents such as HSG179 - the latest UK standards document for health and safety in swimming pools (Health and Safety Executive, 2018). However, the topic is a debated area for various reasons explored in this review.

Whilst there are plenty of academic articles dedicated to the technology and design behind these products in the fields of biometrics, computer science and electronic engineering, there is limited academic research investigating their application to real-world scenarios. Furthermore, there is uncertainty around their use alongside traditional lifeguarding; whether international testing standards (ISO standards) are robust enough; and general risks affecting the effectiveness of these products. This includes factors such as water clarity, high pool occupancy, lighting, glare and attractions such as water slides and wave machines. These concerns alongside the lack of research and high installation costs have resulted in a reluctance by some operators to incorporate DDS into their pools. This signifies the importance of independent research into DDS. intends to support the move towards the shared goal of improved pool safety.

This piece will begin with an overview of the different definitions of DDS, followed by an explanation of the aims and methodology of this review. It will then discuss what the current DDS standards are alongside legislation and guidance available around DDS, and provide a summary of the shared responsibilities towards the effective operation of DDS. Following this, the literature review will examine the co-existence between DDS and traditional lifeguarding, provide an analysis of its impact so far, and conclude with recommendations on the direction of future DDS research.

Project Overview



Purpose

>> Establish and outline what is known on Drowning Detection Systems.
>> Evaluate the current literature on Drowning Detection Systems, including their use in indoor pool environments along with interaction with traditional lifeguarding.
>> Better understand where DDS are positioned in the health and safety landscape of indoor swimming pools.

The value that can be generated from these aims stem from the recognition that currently, there are no published documents drawing together all the current DDS research. The literature review aims to contribute as independent research in this field and hopes to signpost the potential future direction of DDS research.

CHAPTER-2

LITERATURE SURVEY

The differing definitions of DDS, most outline three defining elements:

- surveillance,
- detection of a pool user in difficulty, and
- raising an alarm

For example, ISO_20380 (the document published by the International Organization for Standardization (2017) outlining the international safety requirements and test standards for DDS) defines the technology as an ‘automated system including means for digitizing series of images of people in the pool basin, means for comparing and analyzing digitized images and decision means for setting off and sending an alarm to trained staff when a detection occurs’. In comparison, there are broader definitions that are inclusive of other technologies that focus on the surveillance aspect, for example, ‘DDS is used to describe various electronic systems that are designed to assist with the surveillance of swimmers within the water of a swimming pool’ (Sport England, 2011). This definition would include CCTV that helps give lifeguards an underwater view but does not have the capacity to detect a pool user in difficulty or raise an alarm. For this to be effective, staff would have to make sure the CCTV is being monitored at all times, making the staff experience with this very different to the experience of using a DDS falling under the first definition. It is important to distinguish what exactly constitutes a DDS as there are different areas of responsibility required from different actors involved in the effective operation of DDS, which will be examined in chapter 4. For this literature review, research has focused on the definition used by the ISO and other sources that incorporate all three elements of surveillance, detection and alarm raising.

Existing Problem

Whilst literature on DDS mostly agrees on areas such as the risks and issues associated with DDS performance, there are other areas where sources offer differing points of view, for example, DDS and their co- existence with lifeguards. There is debate around whether DDS can be helpful or harmful towards lifeguarding practices and how DDS may change the landscape of traditional lifeguarding, as well as some disagreement on whether they serve as justification for reducing lifeguard numbers. The term ‘blended lifeguarding’ or ‘modern lifeguarding’ has been newly coined to describe the concept of traditional lifeguarding practices being blended with technology for drowning detection (Swimming Pool Scene, 2017).Currently, there is little qualitative or quantitative research analyzing theexperiences of lifeguards themselves relating to this concept.

PROBLEM STATEMNET DEFINITION

- Swimming is one of the best exercises that helps people to reduce stressin this urban lifestyle. Swimming pools are found larger in number in hotels.

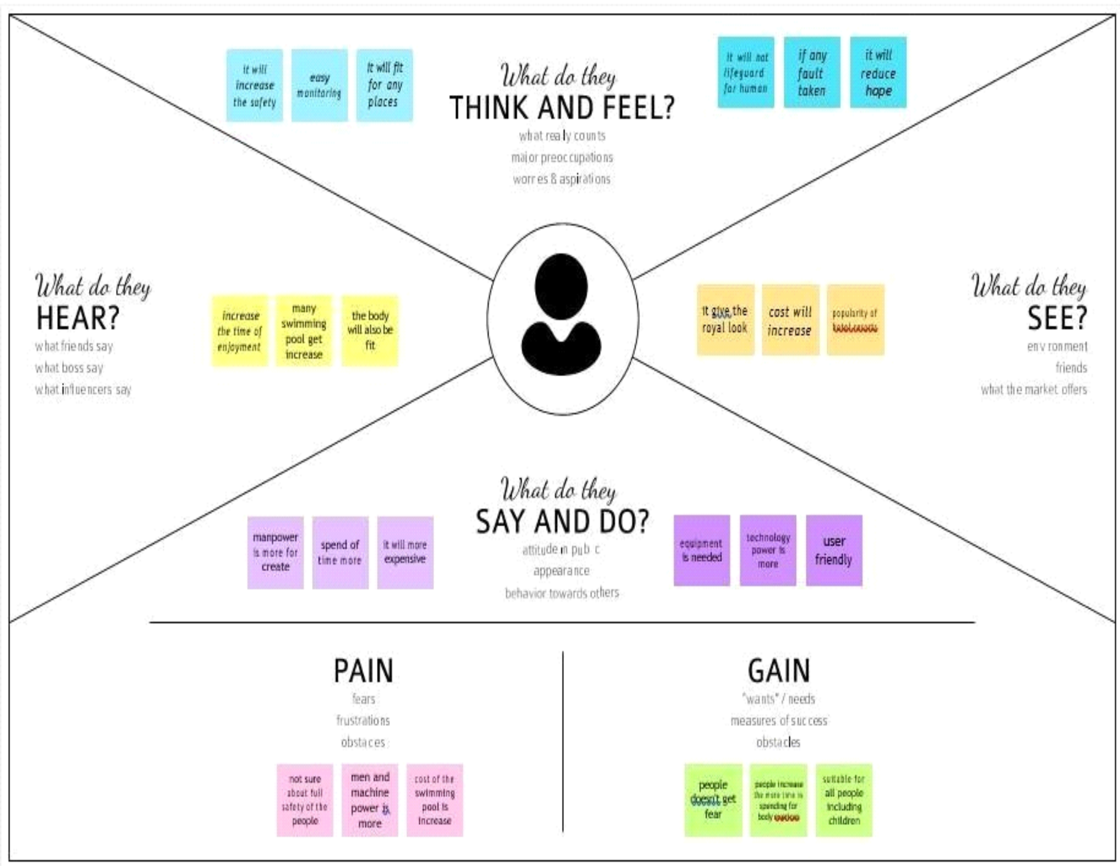
- Applying the CNN algorithm to the dataset.Beginners, especially, often feel it difficult to breathe underwater which causes breathing trouble which in turn causes a drowning accident.

- To overcome this conflict, a meticulous system is to be implemented along theswimming pools to save human life.

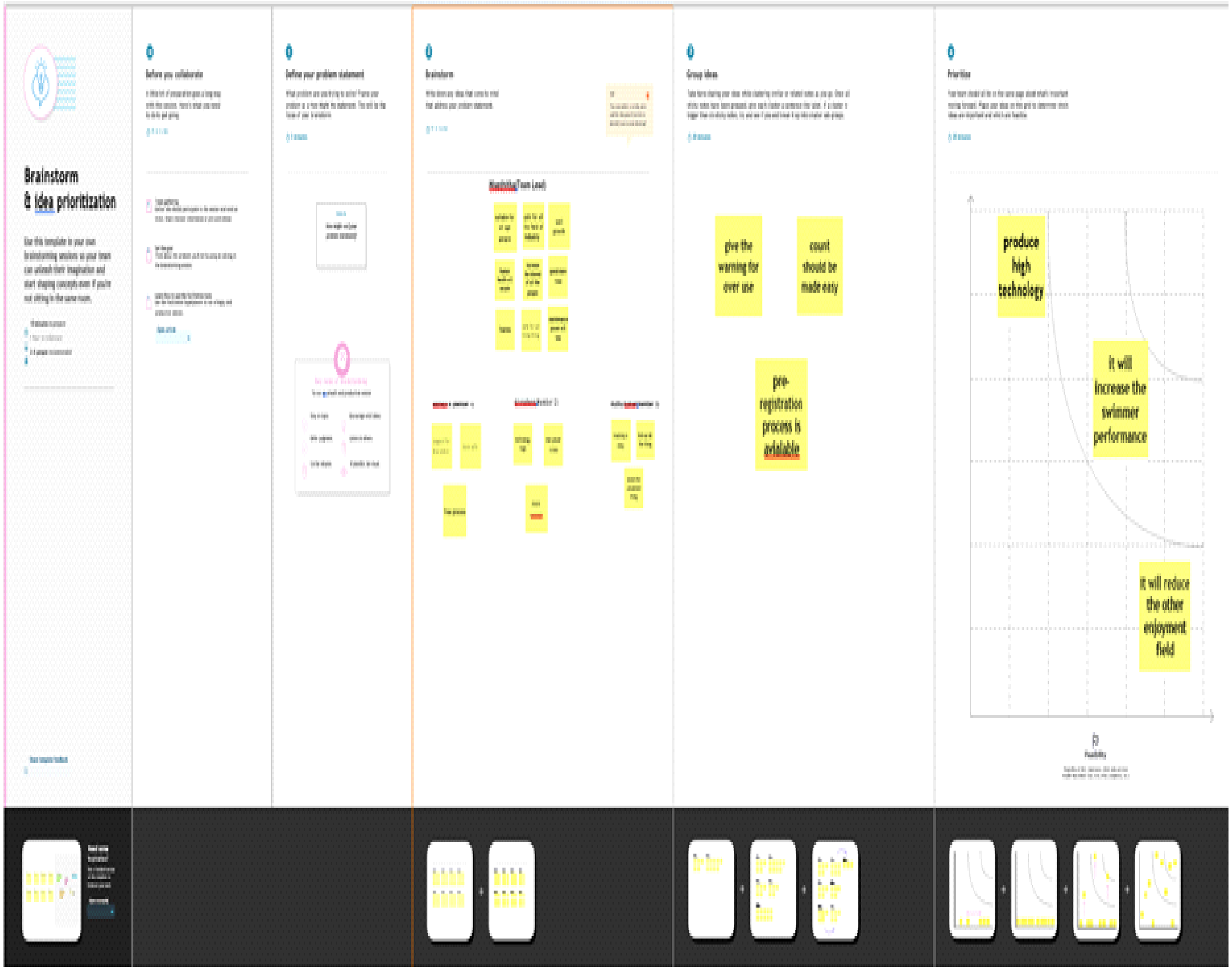
CHAPTER-33.IDEATION & PROPOSED SOLUTION

- **EMPATHY MAP CANVAS**

Virtual-Lifeguard For Swimming Pools to Detect the Active .Drowning



- **IDEATION & BRAINSTORMING**



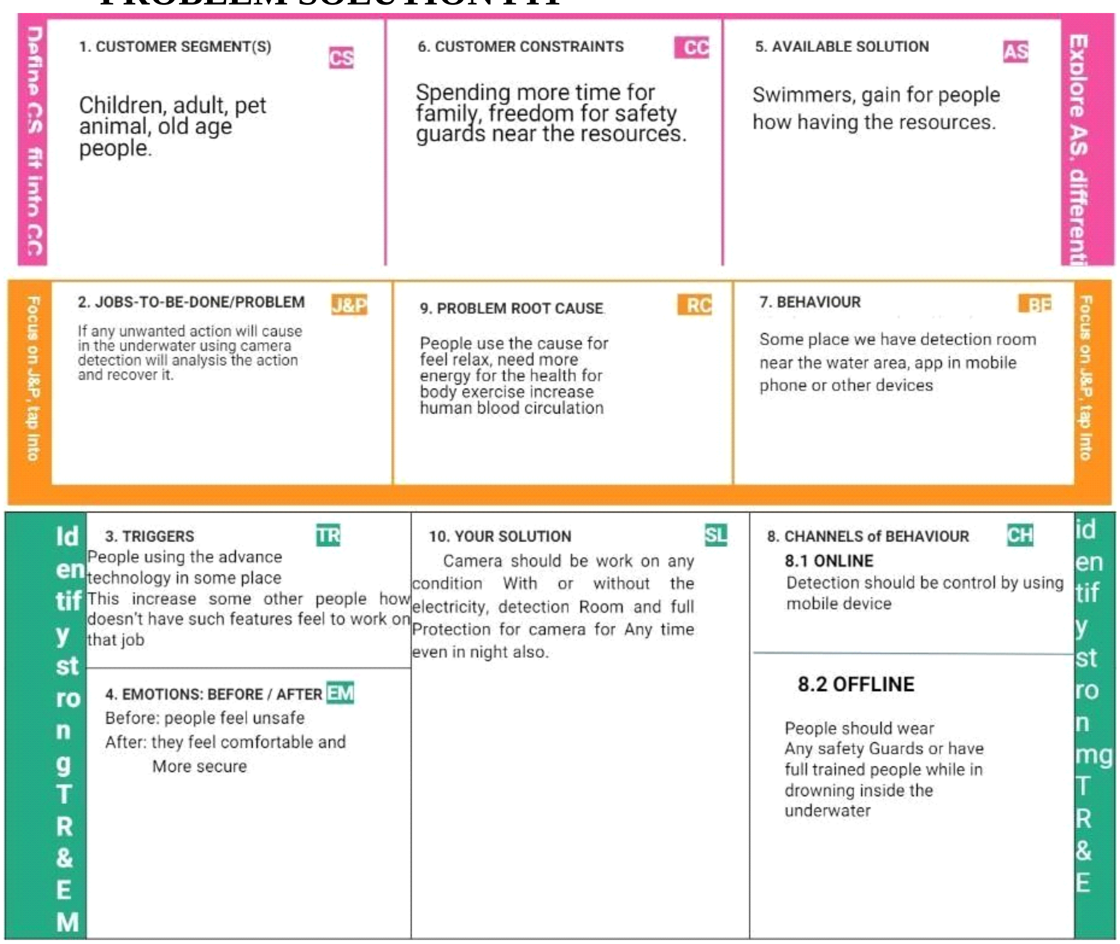
- **PROPOSED SOLUTION**

Proposed Solution Template:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	People use the swimming for enjoyment, health Exercise but for all age of the people pool is really dangerous we need lifeguard, in duty swimming pools are very dangerous in the underwater.
2.	Idea / Solution description	In this project, using Artificial intelligence technology, using the camera help we can detect the people action and positions and also we check breathing level of the people inside the underwater and use of any alarms system we can detect the some of them are in the problem
3.	Novelty / Uniqueness	The uniqueness of the our system is track the

		people position and body condition in the drowning using YOLO Algorithm. It is fast and very speed in the detection
4.	Social Impact / Customer Satisfaction	In world most of them are unexcepted cause very serious death in the underwater not only in the city but most occurs in the rural area in the public places (well, lakes) we should avoid the accident in the underwater drowning
5.	Business Model (Revenue Model)	In the software field this well increase good income. Safety innovation in the swimming related issues this makes attractive for end users to use our software product
6.	Scalability of the Solution	IBM cloud server will collect all the data and stored in the server. This will more safe and secure

- **PROBLEM SOLUTION FIT**



CHAPTER-4

- **REQUIREMENT ANALYSIS**

- **FUNCTIONAL REQUIREMENT**

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Installation	Install the camera inside the underwater, connect necessary app in the phone or otherdevice
FR-2	Detection	Near swimming pool area use detectionroom for monitor or use IBM cloud forstorage purpose of the details
FR-3	Audio	Give the alert signal for the people enterinto the underwater and leaving into underwater

FR-4	Support	Extra support from the lifeguard if any person pulse rate will decrease inside the water
FR-5	Prior alert	Extreme level problem should be occursgive the alert signal for the entire pool

- **NON-FUNCTIONAL REQUIREMENT**

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	A Lifeguard should be present in all the time near pool
NFR-2	**Security**	Alert message or signal should be give by the lifeguard of swimmer
NFR-3	**Reliability**	Triggers if any immediate needs of theswimmer inside the pool
NFR-4	**Performance**	If any unwanted position changes and thepulse rate will decrease this will detect it.
NFR-5	**Availability**	Equipment and other requirement should bechecked by the lifeguards
NFR-6	**Scalability**	Virtual eye lifeguard detects potential drownings and it should be notifies you.

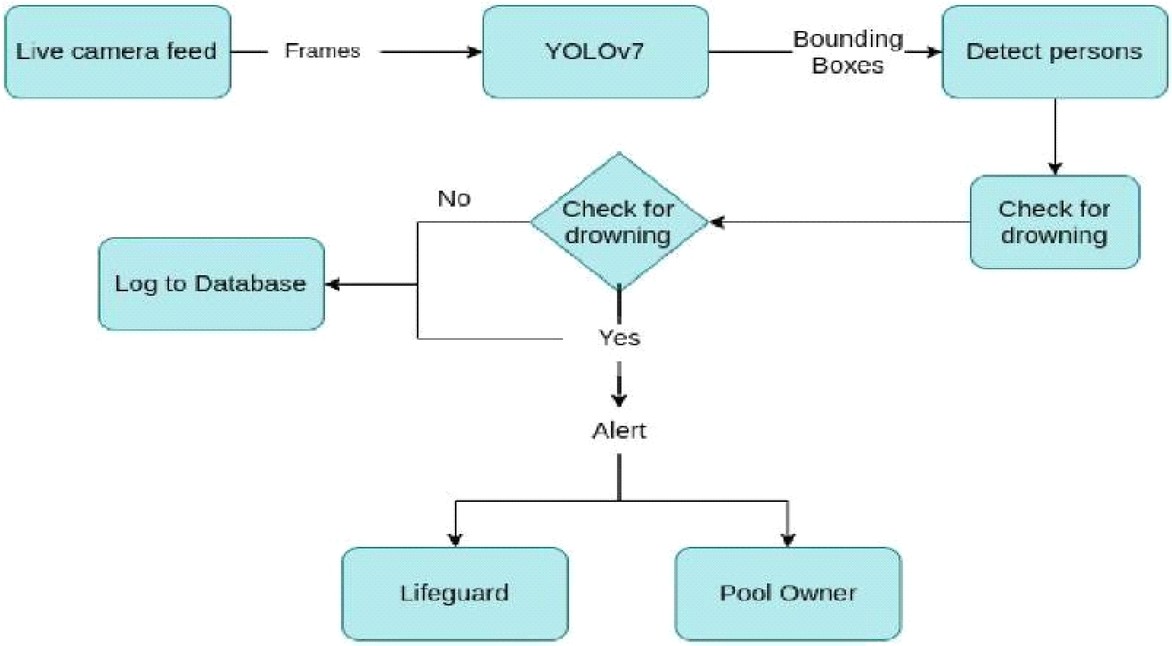
CHAPTER-5

- **PROJECT DESIGN**

- **DATAFLOW DIAGRAMS**

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 theinformation, and where data is stored

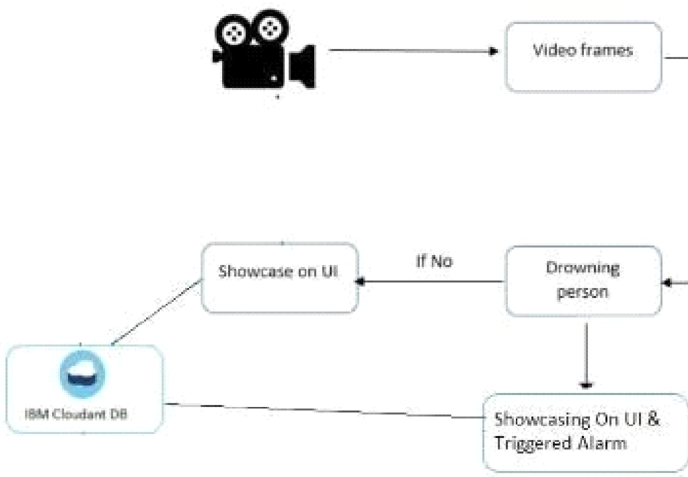


• **SOLUTION & TECHNICAL ARCHITECTURE**

Solution Architecture:

• To find underwater movement while person in drowning they have any Problem or anything else we will find the solution using the Artificial Intelligence (AI) detection technology.

• Usually, such systems can be developed by installing more than 16 cameras underwater and ceiling and analyzing the video feeds to detect any anomalies. AS a POC we make use of one camera that streams the video underwater and analyses the positionof swimmers to assess the probability of drowning, if it is higher then an alert will be generated to attract lifeguards' attention.



• 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)	Installation	USN-1	Install the camera inside the underwater, connect necessaryapp in the phone or other device	I can cameras to the IBM cloudDB	High	Sprint-1
Customer (Lifeguard	Secure thepeople	USN-2	As a user, I can secure the drowning personsfrom the pool	I can save thedrowning person	High	Sprint-1
Customer (swimmers)	safety	USN-3	As a user, I can swim inside the underwater without fear of the Drowning	I can swim safely	medium	Sprint-2
Customer care (Executive)	Contact	USN-4	As a user, I Can resolve if any problem occurs with any device	I can contact the customer care	Medium	Sprint-3

			technically	executiveto resolve anyissues		
Administrator	Dashboard	USN-5	Management ofthe drowning detection systemand database management	I can accessthe system's logs and any other data instantly	High	Sprint-4

CHAPTER-6

SPRINT PLANNING & ESTIMATION

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as onPlanned End Date)	Sprint Release Date(Actual)
Sprint-1	8	6 Days	24 Oct 2022	29 Oct 2022	6	29 Oct 2022
Sprint-2	14	6 Days	31 Oct 2022	05 Nov 2022	12	05 Nov 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	• Nov 2022

SPRINT DELIVERY SCHEDULE

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	VLGFSP-1	As a user, I can register for theapplication by entering my email, password, andconfirmingmy password.	2	High	Nivethitha
Sprint-1	Registration	VLGFSP-2	As a user, I will receive confirmationemail onceI have registered for the application	1	High	Muthukumar
Sprint-1	Registration	VLGFSP -3	As a user, I can register for theapplication through Facebook	2	Low	Azarudeen
Sprint-1	Registration	VLGFSP -4	As a user, I can register for theapplicationthrough Gmail	2	Medium	Abinaya
Sprint-1	Login	VLGFSP -6	As a user, I can log into the application byentering	1	High	Nivethitha

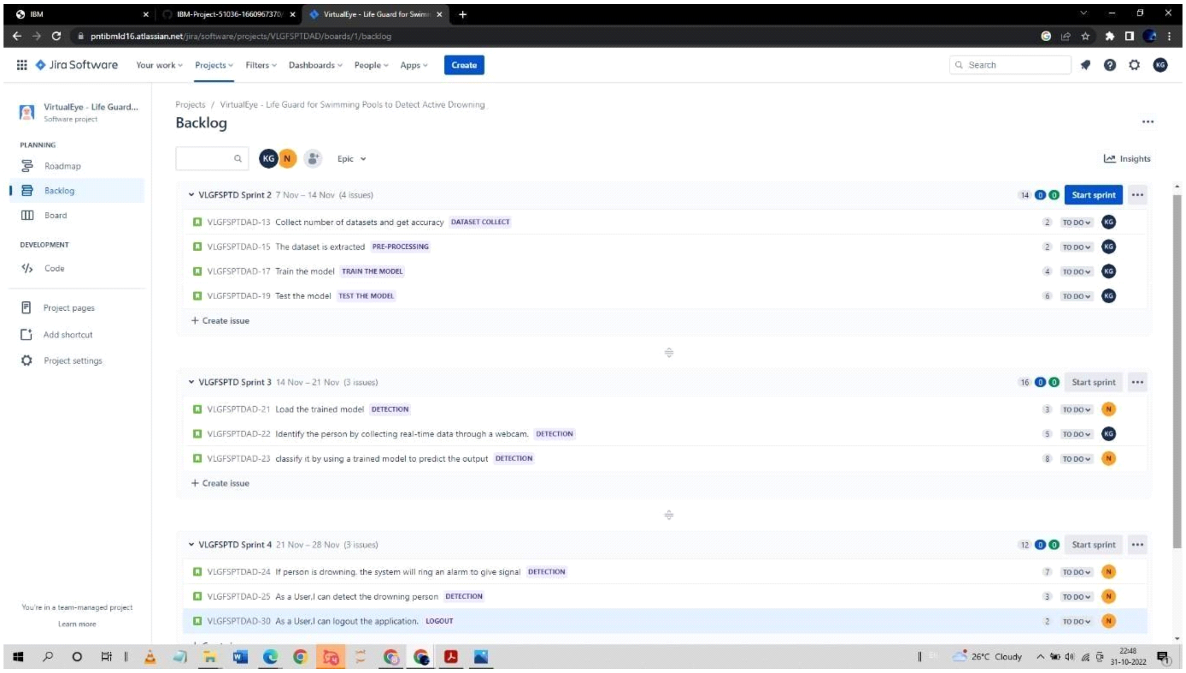
			email & password			
Sprint-2	Dataset Collect	VLGFSP -11	Collect number of datasets and getaccuracy	2	Medium	Muthukumar
Sprint-2	Pre-processing	VLGFSP -12	The dataset is extracted	2	High	Azarudeen
Sprint-2	Train the model	VLGFSP -13	Train the model.	4	High	Abinaya
Sprint-2	Test the model	VLGFSP -14	Test the model	6	High	Nivethitha
Sprint-3	Detection	VLGFSP -15	Load the trained model.	3	High	Muthukumar
Sprint-3	Detection	VLGFSP -16	Identify the person by collecting real- time datathrough a webcam.	5	Medium	Azarudeen
Sprint-3	Detection	VLGFSP -16	classify it by using a trained modelto predictthe output	8	High	Abinaya
Sprint-4	Detection	VLGFSP -17	If person is drowning, the systemwill ring analarm to give signal	7	High	Nivethitha
Sprint-4	Detection	VLGFSP -18	As a User,I can detect the drowning person.	3	Medium	Muthukumar
Sprint-4	Logout	VLGFSP -19	As a User,I can logout the application.	2	Low	Azarudeen

Use the below template to create product backlog and sprint schedule

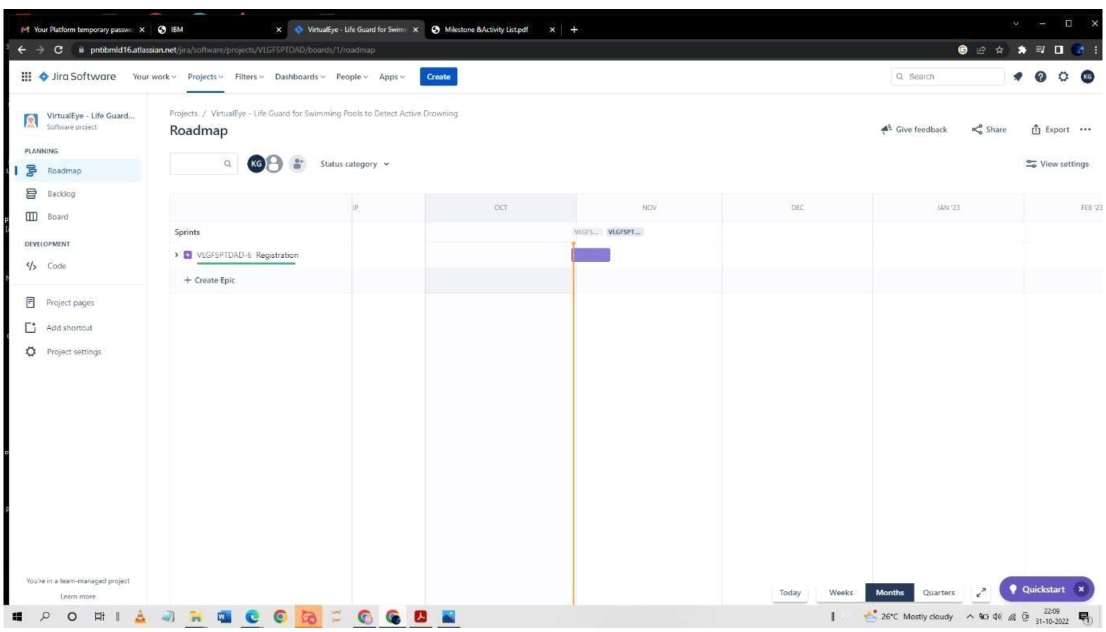
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	VLGFSP-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Kishore Kumar
Sprint-1	Registration	VLGFSP-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Barani
Sprint-1	Registration	VLGFSP -3	As a user, I can register for the application through Facebook	2	Low	Karthika
Sprint-1	Registration	VLGFSP -4	As a user, I can register for the application through Gmail	2	Medium	Babhu Ganesh
Sprint-1	Login	VLGFSP -6	As a user, I can log into the application by entering email & password	1	High	Barani
Sprint-2	Dataset Collect	VLGFSP -11	Collect number of datasets and get accuracy	2	Medium	Karthika
Sprint-2	Pre-processing	VLGFSP -12	The dataset is extracted	2	High	Kishore Kumar
Sprint-2	Train the model	VLGFSP -13	Train the model.	4	High	Babhu Ganesh

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-2	Test the model	VLGFSP -14	Test the model	6	High	Kishore Kumar
Sprint-3	Detection	VLGFSP -15	Load the trained model.	3	High	Babhu Ganesh
Sprint-3	Detection	VLGFSP -16	Identify the person by collecting real-time data through a webcam.	5	Medium	Barani
Sprint-3	Detection	VLGFSP -16	classify it by using a trained model to predict the output	8	High	Karthika
Sprint-4	Detection	VLGFSP -17	If person is drowning, the system will ring an alarm to give signal	7	High	Karthika
Sprint-4	Detection	VLGFSP -18	As a User,I can detect the drowning person.	3	Medium	Babhu Ganesh
Sprint-4	Logout	VLGFSP -19	As a User,I can logout the application.	2	Low	Barani

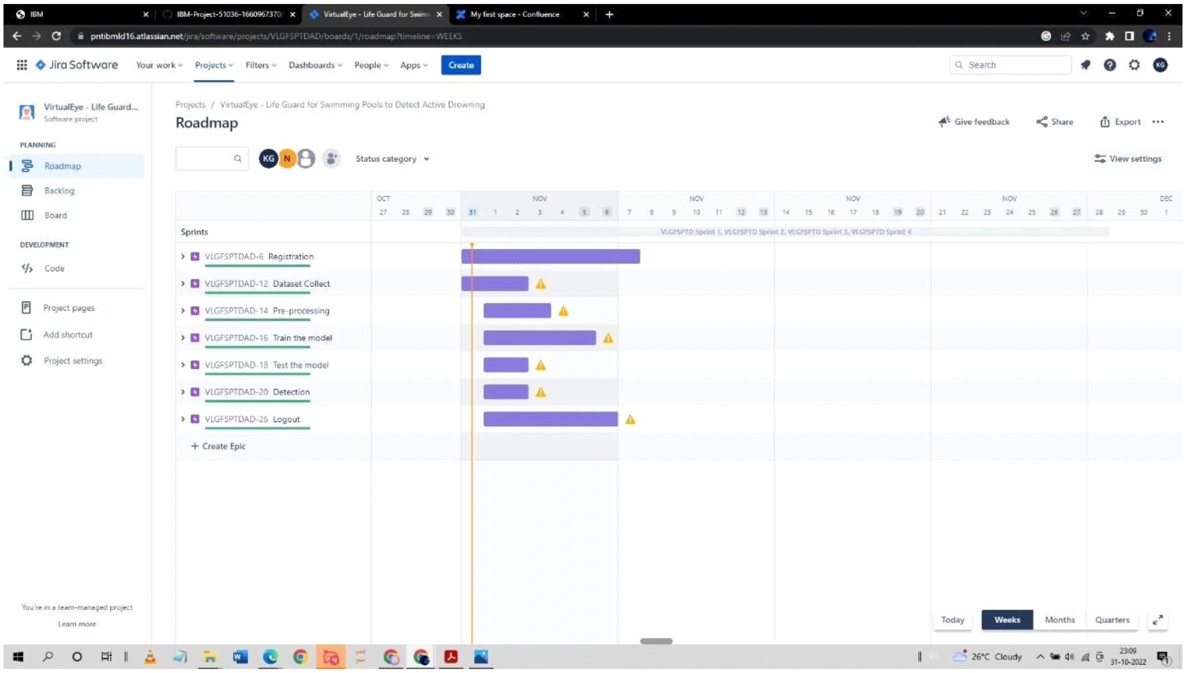
REPORT FROM JIRABacklog (scrum)



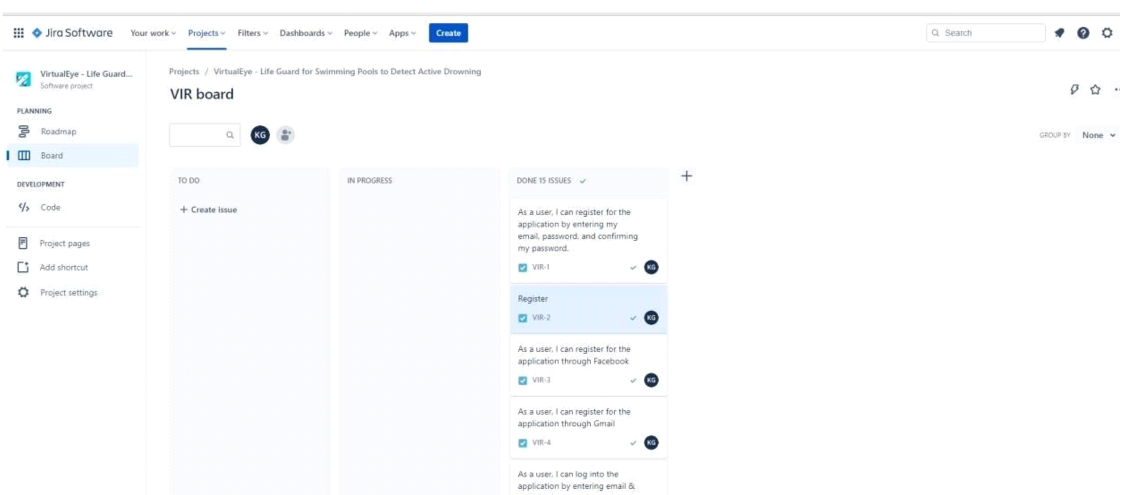
Roadmap



Chart



Board (Kanban)



CHAPTER-7

- **CODING & SOLUTION**
- **FEATURE 1**

[net]

Testing#batch=1

subdivisions=1# Training batch=64 subdivisions=16 width=608 height=608channels=3 momentum=0.9 decay=0.0005 angle=0saturation = 1.5 exposure = 1.5hue=.1

learning_rate=0.01 burn_in=1000 max_batches =500200policy=steps steps=400000,450000

scales=.1,.1

[convolutional] batch_normalize=1filters=32 size=3 stride=1

pad=1 activation=leaky

Downsample

[convolutional] batch_normalize=1filters=64 size=3 stride=2

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=32 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=64 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear#

Downsample

[convolutional] batch_normalize=1filters=128 size=3 stride=2

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=64 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=128 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=64 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=128 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear# Downsample

[convolutional] batch_normalize=1

filters=256size=3stride=2 pad=1 activation=leaky

[convolutional] batch_normalize=1filters=128 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=256 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=128 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=256 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=128 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=256 size=3

stride=1pad=1

activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=128 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=256 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=128 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=256 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=128 size=1 stride=1

pad=1 activation=leaky

[convolutional]

batch_normalize=1filters=256 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=128 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=256 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=128 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=256 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear#

Downsample

[convolutional] batch_normalize=1filters=512 size=3 stride=2

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=256 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=512 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=256 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=512 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=256 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=512 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=256 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=512 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=256 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=512 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=256 size=1 stride=1

pad=1 activation=leaky

[convolutional]

batch_normalize=1filters=512 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=256 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=512 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=256 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=512 size=3 stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear#

Downsample

[convolutional] batch_normalize=1filters=1024 size=3 stride=2pad=1

activation=leaky

[convolutional] batch_normalize=1filters=512 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=1024 size=3stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=512 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=1024 size=3stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=512 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=1024 size=3stride=1

pad=1

activation=leaky

[shortcut]from=-3

activation=linear

[convolutional] batch_normalize=1filters=512 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1filters=1024 size=3stride=1

pad=1 activation=leaky

[shortcut]from=-3

activation=linear ######################

[convolutional] batch_normalize=1filters=512 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1size=3stride=1 pad=1 filters=1024 activation=leaky

[convolutional] batch_normalize=1filters=512 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1size=3stride=1 pad=1 filters=1024

activation=leaky

[convolutional] batch_normalize=1filters=512 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1size=3stride=1 pad=1 filters=1024 activation=leaky

[convolutional]size=1stride=1

pad=1 filters=255activation=linear

[yolo]

mask = 6,7,8

anchors = 10,13, 16,30, 33,23, 30,61, 62,45, 59,119, 116,90,

156,198, 373,326

classes=80 num=9 jitter=.3 ignore_thresh = .7

truth_thresh = 1random=1

[route] layers = -4

[convolutional] batch_normalize=1filters=256 size=1 stride=1

pad=1 activation=leaky

[upsample]stride=2

[route]

layers = -1, 61

[convolutional]

batch_normalize=1filters=256 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1size=3stride=1 pad=1 filters=512 activation=leaky

[convolutional] batch_normalize=1filters=256 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1size=3stride=1 pad=1 filters=512 activation=leaky

[convolutional] batch_normalize=1filters=256 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1size=3stride=1 pad=1 filters=512 activation=leaky

[convolutional]size=1stride=1

pad=1 filters=255activation=linear

[yolo]

mask = 3,4,5

anchors = 10,13, 16,30, 33,23, 30,61, 62,45, 59,119, 116,90,

156,198, 373,326

classes=80 num=9 jitter=.3 ignore_thresh = .7

truth_thresh = 1random=1

[route] layers = -4

[convolutional] batch_normalize=1filters=128 size=1 stride=1

pad=1 activation=leaky

[upsample]stride=2

[route]

layers = -1, 36

[convolutional] batch_normalize=1filters=128 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1size=3stride=1 pad=1 filters=256 activation=leaky

[convolutional] batch_normalize=1filters=128 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1size=3stride=1

pad=1 filters=256activation=leaky

[convolutional] batch_normalize=1filters=128 size=1 stride=1

pad=1 activation=leaky

[convolutional] batch_normalize=1size=3stride=1 pad=1 filters=256 activation=leaky

[convolutional]size=1stride=1

pad=1 filters=255activation=linear

[yolo]

mask = 0,1,2

anchors = 10,13,   16,30,   33,23,   30,61,   62,45,   59,119,   116,90, 156,198, 373,326

classes=80 num=9 jitter=.3 ignore_thresh = .7

truth_thresh = 1random=1

- ## FEATURE 2

#import necessary packagesimportcv2

import os

import numpy as np

from .utils import download_file

initialize = Truenet

= None

dest_dir = os.path.expanduser('~') + os.path.sep + '.cvlib' + os.path.sep + 'object_detection' +os.path.sep + 'yolo' +os.path.sep + 'yolov3'

classes = None

#colors are BGR instead of RGB in pythonCOLORS = [0,0,255], [255,0,0]

def populate_class_labels():

#we are using a pre existent classifier which is more reliable and more efficient than one#we could makeusing only a laptop

#The classifier should be downloaded automatically when you run this scriptclass_file_name ='yolov3_classes.txt'

class_file_abs_path = dest_dir + os.path.sep + class_file_name

url = 'https://github.com/Nico31415/Drowning-Detector/raw/master/yolov3.txt'if notos.path.exists(class_file_abs_path):

download_file(url=url, file_name=class_file_name, dest_dir=dest_dir)f =open(class_file_abs_path, 'r')

classes = [line.strip() for line in f.readlines()]

return classes

def get_output_layers(net)

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

output_layers = [layer_names[i[0] - 1] for i in net.getUnconnectedOutLayers()]

return output_layers


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

global COLORSglobal classes

if classes is None:
classes = populate_class_labels()

for i, label in enumerate(labels):

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

if write_conf:
label += ' ' + str(format(confidence[i] * 100, '.2f')) + '%'
#you only need to points (the opposite corners) to draw a rectangle. These points#are stored in thevariable bbox
cv2.rectangle(img, (bbox[i][0],bbox[i][1]), (bbox[i][2],bbox[i][3]), color, 2)

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

return img

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

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

global classes global dest_dir
```

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

#this part of the script just downloads the YOLO filesconfig_file_name = 'yolov3.cfg'

config_file_abs_path = dest_dir + os.path.sep + config_file_name

weights_file_name = 'yolov3.weights'

weights_file_abs_path = dest_dir + os.path.sep + weights_file_name

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

if not os.path.exists(config_file_abs_path):

download_file(url=url, file_name=config_file_name, dest_dir=dest_dir)

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

if not os.path.exists(weights_file_abs_path):

download_file(url=url, file_name=weights_file_name, dest_dir=dest_dir)

global initializeglobal net

if initialize:

classes = populate_class_labels()

net = cv2.dnn.readNet(weights_file_abs_path, config_file_abs_path)initialize = False

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

net.setInput(blob)

outs = net.forward(get_output_layers(net))

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

for out in outs:

for detection in out: scores =detection[5:]

class_id = np.argmax(scores) max_conf = scores[class_id] ifmax_conf > confidence:

center_x = int(detection[0] * Width) center_y =int(detection[1] * Height)w = int(detection[2] *Width)

h = int(detection[3] * Height)x =center_x - w / 2

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

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

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

for i in indices:

i = i[0]

box = boxes[i]x = box[0]

y = box[1] w =box[2] h = box[3]

bbox.append([round(x), round(y), round(x+w), round(y+h)]) label.append(str(classes[class_ids[i]])) conf.append(confidences[i])

return bbox, label, conf

Github Link:

Link- Github

Demo HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-26977-1660042231/blob/main/Final%20Deliverables/alarm.mp3"HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-26977-1660042231/blob/main/Final%20Deliverables/alarm.mp3" HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-26977-1660042231/blob/main/Final%20Deliverables/alarm.mp3" HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-26977-1660042231/blob/main/Final%20Deliverables/alarm.mp3"HYPERLINK "https://github.com/IBM-

EPBL/IBM-Project-26977-1660042231/blob/main/Final%20Deliverables/alarm.mp3" HYPERLINK "https://github.com/IBM-EPBL/IBM-Project-26977-1660042231/blob/main/Final%20Deliverables/alarm.mp3" link