VIRTUAL EYE - LIFEGUARD FOR SWIMMING POOLS TO DETECT ACTIVE DROWNING

NALAIYA THIRAN PROJECT BASED LEARNING

On

PROFESSIONAL READINESS FOR INNOVATION, EMPLOYABILITY AND ENTREPRENEURSHIP

A PROJECT REPORT

JONES STEVE WALKER P 19106048

PRAVEEN D 19106085

NISHANTH DANIEL J 19106079

PRASATH S 19106084

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

HINDUSTHAN COLLEGE OF ENGINEERING AND TECHNOLOGY

Approved by AICTE, New Delhi, Accredited with 'A' Grade by NAAC

(An Autonomous Institution, Affiliated to Anna University, Chennai)

COIMBATORE – 641 032

November 2022

Hindusthan College of Engineering And Technology

Approved by AICTE, New Delhi, Accredited with 'A' Grade by NAAC **(An Autonomous Institution, Affiliated to Anna University, Chennai)** Valley Campus, Pollachi Highway, Coimbatore – 641 032

INTERNAL MENTOR:

Dr. P. K. Ramya jayakumar

Associate Professor

Department of Electronics and Communication Engineering Hindusthan College of Engineering and Technology, Coimbatore - 641 032

INDUSTRY MENTOR

SWATHI

IBM

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that project report **"VirtualEye - Life Guard for Swimming Pools to Detect Active Drowning"** is the bonafide work of **"JONES STEVE WALKER P, PRAVEEN D, PRASATH S, NISHANTH DANIEL J,"** who carried out the project work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr.P.Vijayalakshmi

HEAD OF THE DEPARTMENT

Professor and Head,

Department of ECE,

Hindusthan College of Engineering

and Technology,

Coimbatore - 641 032

SIGNATURE

Dr.P.K.Ramya Jayakumar

SUPERVISOR

Associate Professor,

Department of ECE,

Hindusthan College of Engineering

and Technology,

Coimbatore - 641 032.

Submitted for Project Viva-Voice conducted on …….

—---------- ----------

INTERNAL EXAMINER **EXTERNAL EXAMINER**

TABLE OF CONTENT

INTRODUCTION

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 analyzes the position of swimmers to assess the probability of drowning, if it is higher than an alert will be generated to attract lifeguards' attention.

1.1 PROJECT OVERVIEW

Designed for those who have to guarantee every day the safety in public and intensive-use pools, VirtualEye LifeGuard 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. LifeGuard is a drowning detection system that detects every dangerous situation and accident. The VirtualEye software works in close integration with the cameras installed in the pool to continuously scan the pool. Thanks to this combination of hardware, software and profound innovations, today LifeGuard represents excellence in drowning detection. When it comes to swimmers in trouble, every second counts. LifeGuard makes itself heard loud and clear in case of danger. The built-in notification system produces alarms within 10 seconds on smartwatches, phones, flashing lights and other configurable devices. In addition, VirtualEyes advanced technology can provide real-time location and image of the danger, making rescue operations easier.

The VirtualEye LifeGuard system is able to record all the activities in the pools and to classify critical situations from normal ones in order to keep track of what happened. Thanks to its advanced image archiving system, LifeGuard meets the legislative requirements for the protection of personal data.The protection of swimmers is ensured to all facilities by a vigilance provided by personnel assigned to control the activities carried out in the pools. These controls have several critical points. The biggest problem is the difficulty in seeing the bottom of the pool. VirtualEye LifeGuard is specifically designed to provide support to lifeguards in the supervision of swimmers. It offers an additional level of safety and integrates seamlessly into rescue operations.

1.2 PURPOSE

Designed for those who have to guarantee every day the safety in public and intensive-use pools, VirtualEye LifeGuard 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.

When it comes to swimmers in trouble, every second counts. AngelEye LifeGuard makes itself heard loud and clear in case of danger. The built-in notification system produces alarms within 10 seconds on smartwatches, phones, flashing lights and other configurable devices. In addition, AngelEye's advanced technology can provide real-time location and image of the danger, making rescue operations easier.

The protection of swimmers is ensured to all facilities by a vigilance provided by personnel assigned to control the activities carried out in the pools. These controls have several critical points. The biggest problem is the difficulty in seeing the bottom of the pool. AngelEye LifeGuard is specifically designed to provide support to lifeguards in the supervision of swimmers. It offers an additional level of safety and integrates seamlessly into rescue operations

3. LITERATURE SURVEY

Video surveillance can be used as a tool for monitoring and security. Observing public and private sites has increasingly become a very sensitive issue. The visual monitoring capabilities can be employed in many different locations to help people live more safely. Videobased surveillance systems are designed and installed in places such as railway stations, airports, and even dangerous environments. Image processing, pattern recognition and machine-vision based methods are efficient ways for real-time intelligent monitoring of the objects or events of interest. The existing surveillance systems deliver valued information in monitoring of large areas. Applying intelligence in video surveillance systems allows real time monitoring of places, people and their activities. The tracking approach can change with varying targets and can change from a single camera to multiple camera configurations. Tracking methods in video surveillance use different parameters such as objects' motion, position, path of movement and velocity, biometrics such as skin color and many more.

The tracking must be robust and overcome occlusion and noise which are common problems in monitoring. One important environment that the need for monitoring systems is crucially sensed is the swimming pool. Each year many people including children are drowned or very close to drowning in the depths of the swimming pools, and the life guards are not trained well enough to handle these problems.This raises the need for having a system that will automatically detect the drowning person and alert the lifeguards of such danger. Real-time detection of a drowning person in swimming pools is a challenging task that requires an accurate system. The challenge is due to the presence of water ripples, shadows and splashes and therefore detection needs to have high accuracy. The wearable drowning monitor device can detect drowning accidents and alarm. The device has seven main modules, including microprocessor, power module, SD memory card module, LED warning module, acceleration sensor module, water pressure sensor module. When swimming the human arm must constantly wave in the water, if drowning, arm motion of floating is significantly reduced, and if falling into the water, almost motionless. According to the physiological response of human drowning, it can detect drowning accidents by recording arm motion real-time through wearable wrist accelerometer devices. This accelerometer is packed with embedded functions with edible user programmable options, configurable to two interrupt pins. The pressure sensor is installed to judge whether the human body is in the water. The red LED is used for drowning warning. One blue LED is used to get the work status of the device which will flash every few seconds in order to save precious energy. Because LED light emitting angle is generally relatively small, 5 red LED lights of upward and around direction are installed to make LED alarm signal caller. Two keys are designed for the demo device

2.1 EXISTING PROBLEM

Lei Fei, Wang Xueli, Chen Dongsheng, proposed a background subtraction method for drowning detection and swimmer identification using visual surveillance in their research paper. This method fails to reflect real background accurately thus restricting model accurate shape detection of moving objects. It also fails to reflect sudden background changes.

Ajil Roy, Dr. K. Srinivasan, proposed drowning detection using RFID-based swimming goggles, however, this model also fails to overcome the limitation of accuracy since the water sensor is not placed very close to the mouth and nose. But this model successfully overcomes limitations of video surveillance-based drowning detection systems like the need for high power computing devices.

Chi Zhang, Xiaoguang Li, Fei Lei, proposed ?A Novel Camera-Based Drowning Detection Algorithm? using input video sequences obtained from underwater cameras. In this case, to detect drowning swimmers an implementable real-time detection system with high accuracy will be needed.

2.2 REFERENCE

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

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

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

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

[5] Gudyś, Adam, et al. "Tracking people in video sequences by clustering feature motion paths." Computer Vision and Graphics. Springer International Publishing, 2014. 236-245.

[6] Vezzani, Roberto, Davide Baltieri, and Rita Cucchiara. "People re identification in surveillance and forensics: A survey." ACM Computing Surveys (CSUR) 46.2 (2013): 29.

[7] Bierens, Joost, and Andrea Scapigliati. "Drowning in swimming pools." Microchemical journal 113 (2014): 53-58.

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

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

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

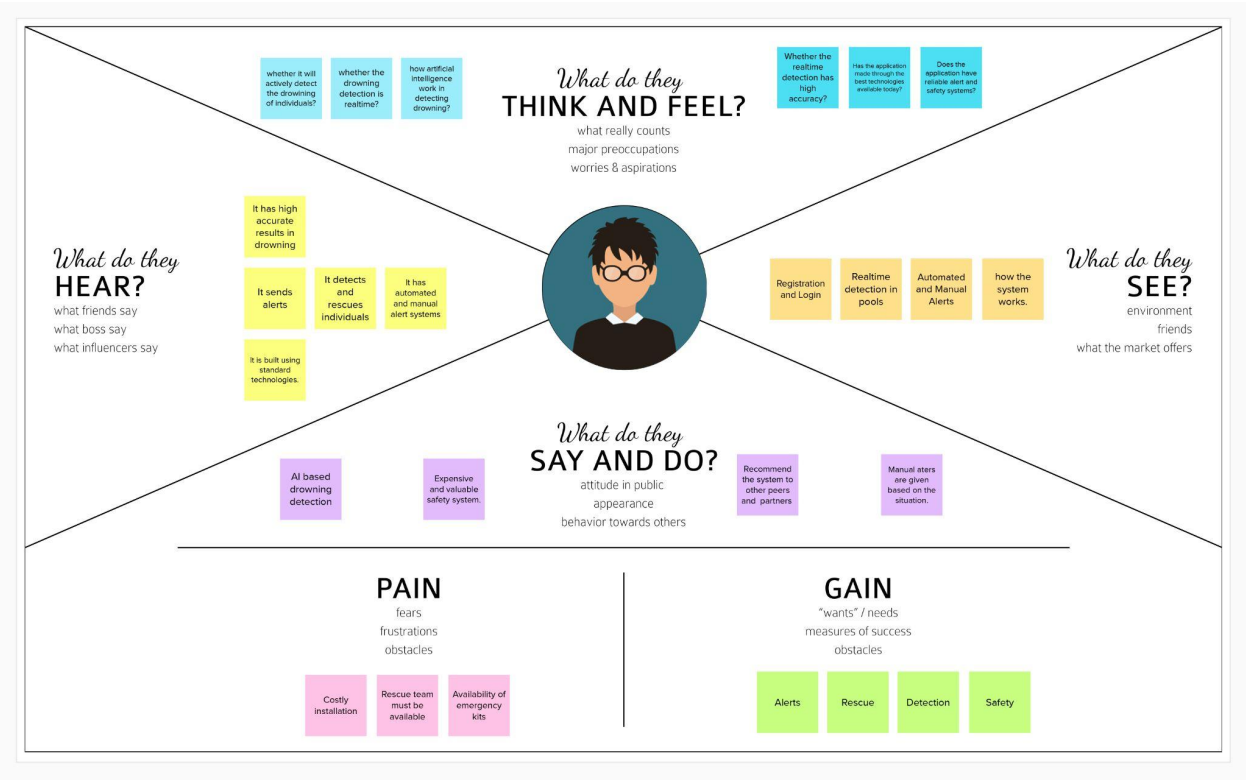
2.3 Problem statement Definition

Problem statement	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 analyzes the position of swimmers to assess the probability of drowning, if it is higher than an alert will be generated to attract lifeguards' attention
I am (USER)	A user, detect the drowning person in Swimming pools to alert the lifeguard
I'm trying to	Analyzing the body movement to Detect drowning people using 16 cameras. And alert to life guard
But	I am not sure if I should alert the life guards he will rescue people

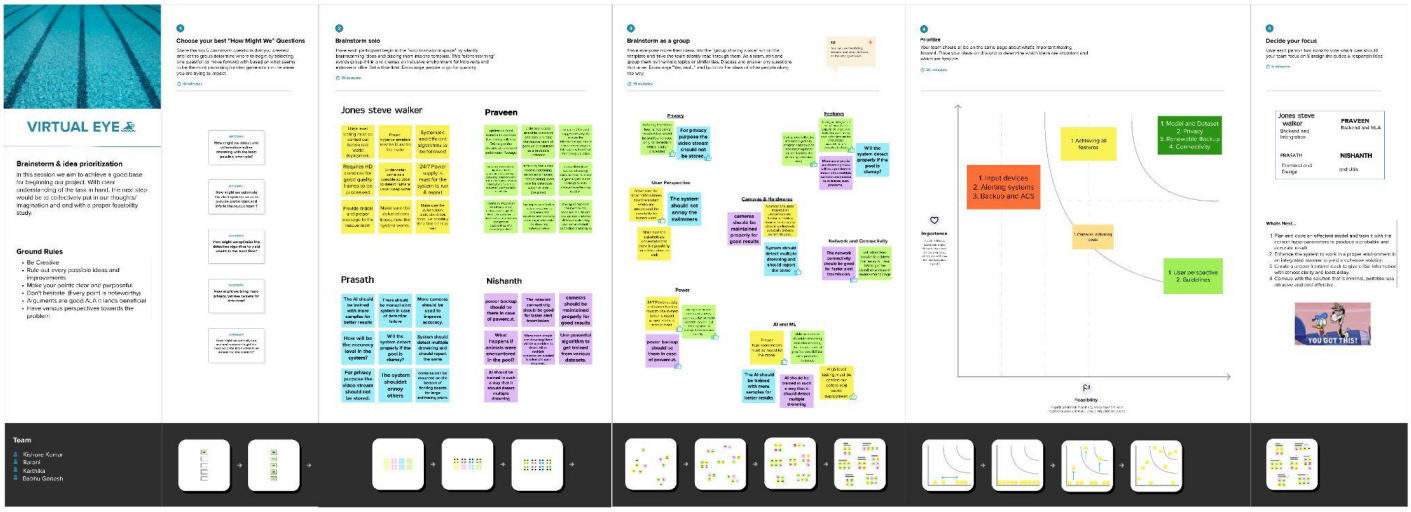
Because	It might cause many death in Swimming Pools increasing death rate
Which makes me feel	Children and beginner Swimmers may affect and lead to death. It will reduce by an VirtualEye cameras

3. IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION & BRAINSTORMING



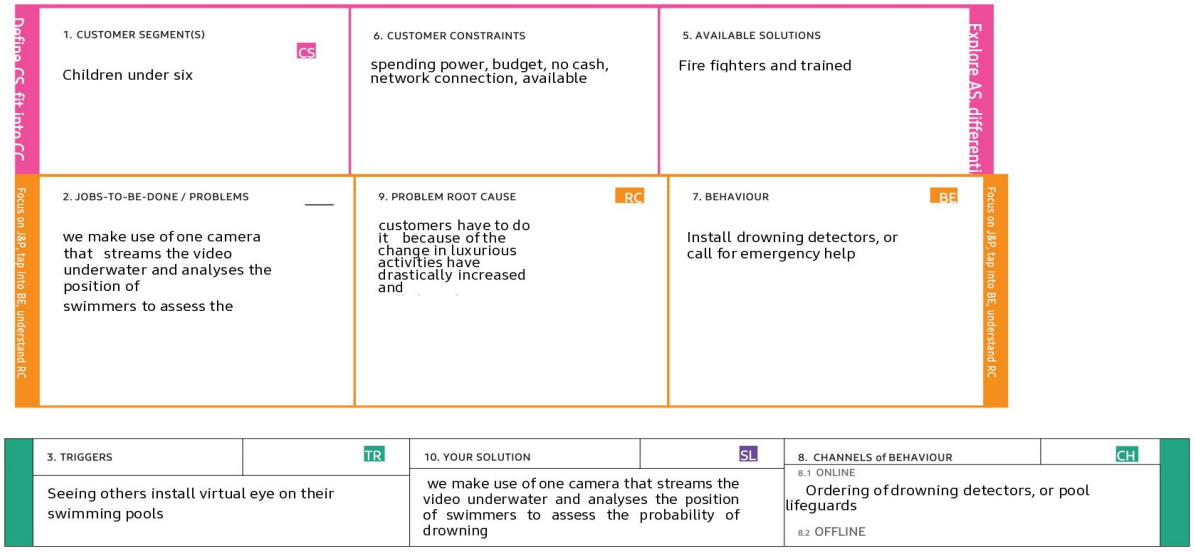
3.3 PROPOSED SOLUTION

S.No	Parameter	Description
1	Proposed Statement (Problem to be solved)	To detect drowning people using AI & ML algorithm
2	Idea / Solution Description	We will develop a motion detection system using AI. Using 16 cameras to Analyze body movement. And alert to life guards
3	Novelty / Uniqueness	It reduces death rate in Swimming Pools
4	Social Impact / Customer Satisfaction	It will save from death in Swimming pools
5	Business Model (Revenue Model)	Artificial Intelligence it will detect it will Safe guide like from drowning in Swimming Pools
6	Scalability of the solution	Using 16 cameras fixing around swimming pools walls and programming with AI algorithm

3.4 PROBLEM SOLUTION FIT

Project Title: VirtualEye - Life Guard for Swimming Pools to Detect Active Drowning
Team ID: PNT2022TMID10134

Project Design Phase-I - Solution Fit Template



4 REQUIREMENT ANALYSIS

4.1 Function Requirement

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 requirement

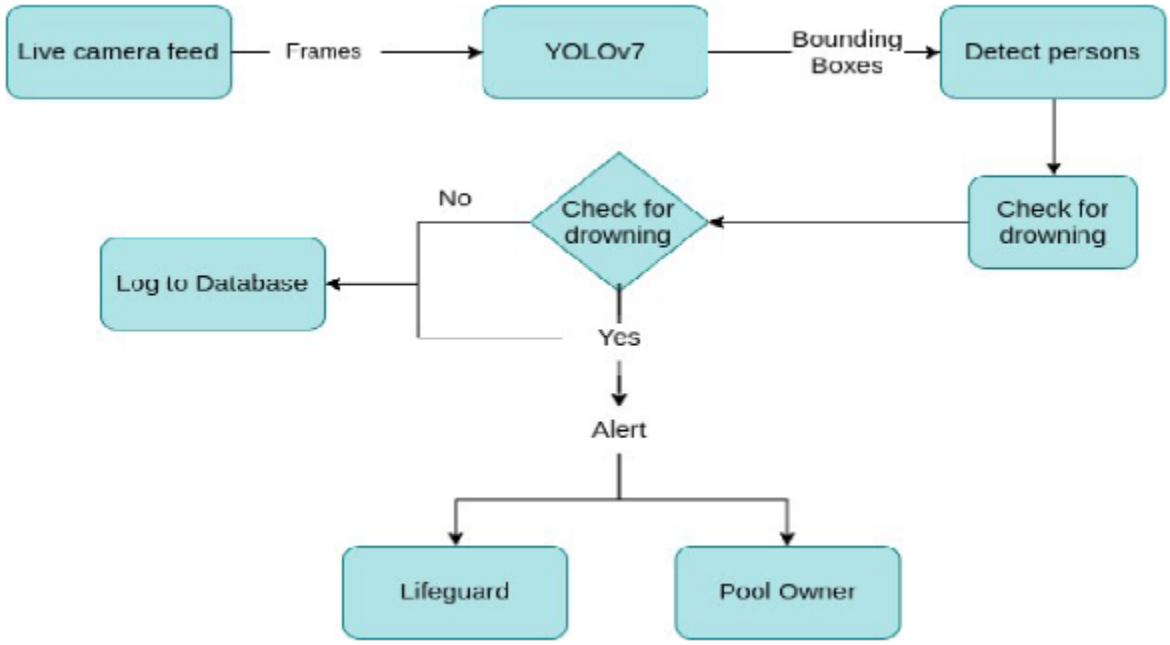
Following are the non-functional requirements of the proposed solution.

NFR.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.

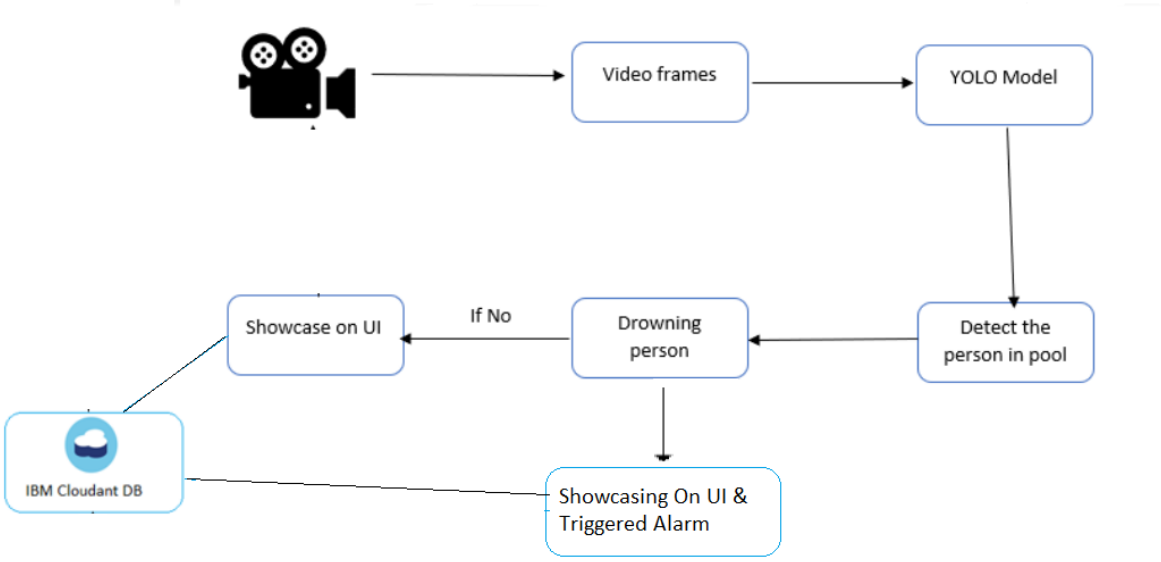
NFR-2	Security	Lifeguards should be aware of the alert message to save the life of the swimmer
NFR-3	Reliability	Virtual eye lifeguard triggers an immediate prior alarm if a swimmer is in peril, helping to avoid panic even in critical situations.
NFR-4	Performance	The performance of the tool works better 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



5.2 Solution & Technical Architecture



5.3 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 rescue the drowning persons from the pool	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
Administrator		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 & SCHEDULING

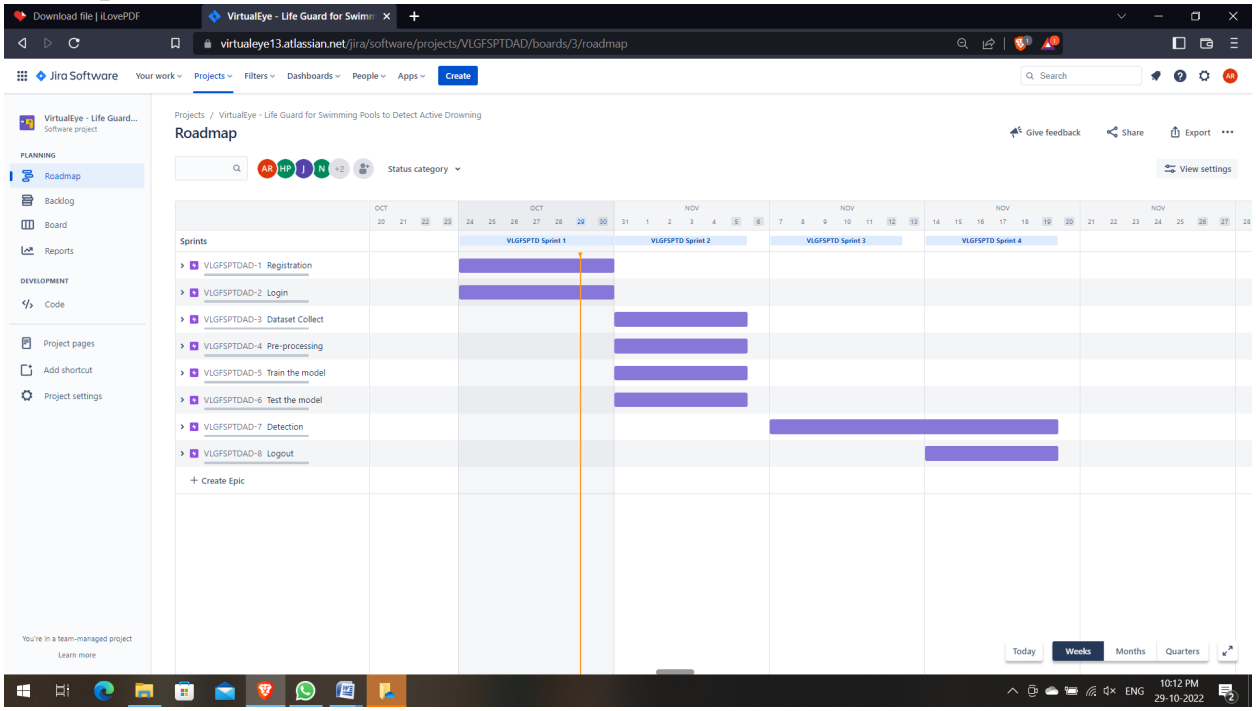
6.1. Sprint Planning & Estimation

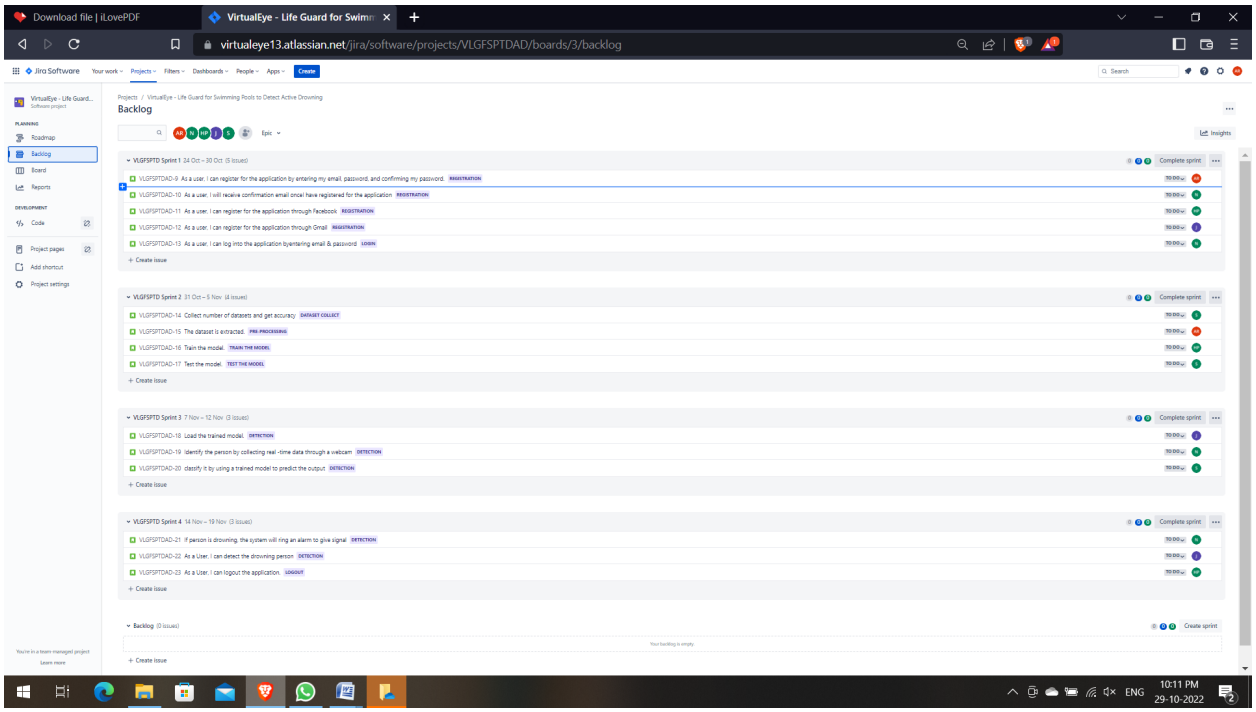
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a lifeguard , I can register for the application by entering my email, password, and confirming my password.	2	High	Jones Praveen Nishanth Prasath
Sprint 1	User confirmation	USN-2	As a lifeguard, I will receive the confirmation mail once I have registered for the application	2	Medium	Jones Praveen Prasath Nishanth
Sprint-1	Login	USN-3	As a lifeguard , I can log the application by entering email & password	2	High	Jones Nishanth Prasath Praveen
Sprint-3	Coding (Accessing datasets)	USN-1	Coding is a set of instructions used to manipulate information so that a certain input results in a particular output.	2	High	Praveen Jones Prasath Praveen
Sprint-4	Application building	USN-1	As a Lifeguard , It will show the current Information of the swimming pool	1	Medium	Prasath Praveen Jones Nishanth

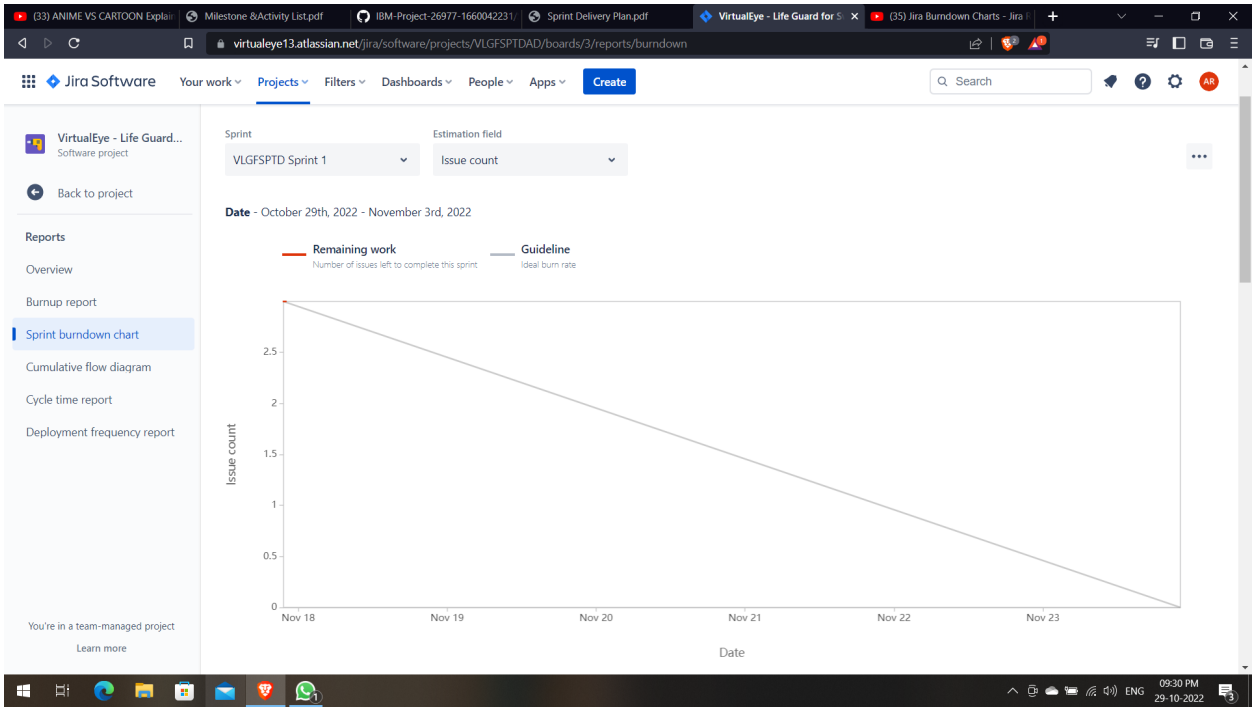
6.2 Sprint Delivery Schedule

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	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 form JIRA




Download file | iLovePDF
VirtualEye - Life Guard for Swimn
virtualeye13.atlassian.net/jira/software/projects/VLGFSPTDAD/boards/3/backlog
Backlog
VLGFSPTD Sprint 1
VLGFSPTD Sprint 2
VLGFSPTD Sprint 3
VLGFSPTD Sprint 4
Backlog
ENG
10:11 PM
29-10-2022


virtualeye13.atlassian.net/jira/software/projects/VLGFSPTDAD/boards/3/reports/burndown
Jira Software
Your work
Projects
Filters
Dashboards
People
Apps
Create
Search
VirtualEye - Life Guard...
Software project
Back to project
Reports
Overview
Burnup report
Sprint burndown chart
Cumulative flow diagram
Cycle time report
Deployment frequency report
You're in a team-managed project
Learn more
Sprint
VLGFSPTD Sprint 1
Estimation field
Issue count
Date - October 29th, 2022 - November 3rd, 2022
Remaining work
Number of issues left to complete this sprint
Guideline
Ideal burn rate
Issue count
2.5
2
1.5
1
0.5
0
Nov 18
Nov 19
Nov 20
Nov 21
Nov 22
Nov 23
Date
ENG
09:30 PM
29-10-2022

7. CODING & SOLUTIONING

7.1 Feature 1

7.2 Feature 2

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

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

def populate_class_labels():

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

    return classes
```

7.2 Feature 2

The automated drowning detection system works on the principle of differential pressure. The system contains two fundamental modules: to begin with, the wristband consists of pressure sensors on the transmitter side. Second, the receiver module at the swimming pool site.

8. TESTING
8.1 Test Cases

Test case ID	Feature Type	Home Page	Test Scenario	Steps TO Execute	Test	Expected Result	Actual Result
LoginPage_ TC_001	Functional	Home Page	Verify user is able to see the Login/Signup popup when user clicked on My account button	I.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
LoginPage_ TC_002	Functional	Home Page	Verify the UI elements in Login/Signup popup	I.Enter tJRL and dick go 2.Click on My Account dropdown 3.Verify login/Singup popup with below UI elements:	Login.html	Application should show below elements: a.email text box b.password text box c.Login button with orange color d. New customer? Create account link e.Last password? Recovery password link	

LoginPage_ TC_003	Functional	Home Page	Verify user is able to log into application with Valid credentials	I.Enter URL and dick go 2.Click on My Account dropdown 3.Enter Valid username/e mail in Email text 4.Enter valid password in password text box 5. Click On in button	Username:la x@gmail password: lax26	User should navigate to prediction homepage	Working as
LoginPage_ TC_004	Functional	Login page	Verify user is able to log into application with Invalid credential	1, Enter URL and click go 2.Click on My Account button 3.Enter Invalid username/e mail in Email text box 4.Enter valid password in password text box 5.Click on •n button	Username:la x password:la x26	Application should show 'Incorrect email or password ' validation message.	Working as
LoginPage_ TC_004	Functional	Login page	Verify user is able to log into application with Invalid credential	I-Enter URL and click go 2.Click On My Account dropdown 3.Enter Valid username/e mail in Email text box 4.Enter Invalid password in password text box 5.Click on in button	username:la x26@mail password:la x26	Application should show •Incorrect email or password ' validation message.	Working as
LoginPage_ TC_005	Functional	Login page	Verify user is able to	I.Enter URL and click go	username:la x26@mail	Application should show	Working as

			into application with lnValid credentials	2.Click on My Account dropdown 3.Enter Invalid username/e mail in Email text box 4. Enter Invalid password in password text box 5. Click on I in button	password:18 03	'Incorrect email or password ' validation message.	
LoginPage_ TC_006	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 lifeguard if people are drowning	Working as

10. ADVANTAGES & DISADVANTAGES

Advantages:

This system don’t have to wait until life guard comes to rescue because it has uplifting mesh.

This is a very fast process.

More effective and cost Efficient than previous other models

Disadvantages:

Internet connection is necessary to use GPS or sending alert messages. Sometimes to send messages SIM balance may be required.

11. CONCLUSION.

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

12. FUTURE SCOPE

Availability of better dataset, 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.

The swimmers in the pool are detected and tracked using the Pi camera. As soon as the swimmer remains under a certain level for more than a determined time, Raspberry Pi will calculate that swimmer's position, path of movement and send an order to the linear stage. The linear stage will uplifts the swimmer and Meanwhile, a warning message will signal the life guard of imminent danger. With such systems, the number of drownings would be reduced. For future development, the system is currently being improved by attaching an infrared LED to the swimmer's vest.

13. APPENDIX

Source code

```
#import necessary packages
import cv2
import os
import numpy as np
from .utils import download_file
initialize = True
net = None
dest_dir = os.path.expanduser('~') + os.path.sep + '.cvlib' + os.path.sep +
'object_detection' + os.path.sep + 'yolo' + os.path.sep + 'yolov3'
classes = None
#colors are BGR instead of RGB in python
COLORS = [0,0,255], [255,0,0]
def populate_class_labels():
#we are using a pre existent classifier which is more reliable and more
efficient than one
#we could make using only a laptop
#The classifier should be downloaded automatically when you run this
script
class_file_name = 'yolov3_classes.txt'
class_file_abs_path = dest_dir + os.path.sep + class_file_name
url = 'https://github.com/Nico31415/Drowning
Detector/raw/master/yolov3.txt'
if not os.path.exists(class_file_abs_path):
download_file(url=url, file_name=class_file_name, dest_dir=dest_dir)
f = open(class_file_abs_path, 'r')
classes = [line.strip() for line in f.readlines()]
return classes
def get_output_layers(net):
#the number of output layers in a neural network is the number of
possible
#things the network can detect, such as a person, a dog, a tie, a
phone...
layer_names = net.getLayerNames()
output_layers = [layer_names[i[0] - 1] for i in
net.getUnconnectedOutLayers()]
return output_layers
def draw_bbox(img, bbox, labels, confidence, Drowning, write_conf=False):
global COLORS
global classes
if classes is None:
classes = populate_class_labels()
```

```
for i, label in enumerate(labels):
#if the person is drowning, the box will be drawn red instead of blue
if label == 'person' and Drowning:
color = COLORS[0]
label = 'DROWNING'
else:
color = COLORS[1]
if write_conf:
label += ' ' + str(format(confidence[i] * 100, '.2f')) + '%'
#you only need to points (the opposite corners) to draw a rectangle.
These points
#are stored in the variable bbox
cv2.rectangle(img, (bbox[i][0],bbox[i][1]), (bbox[i][2],bbox[i][3]),
color, 2)
cv2.putText(img, label, (bbox[i][0],bbox[i][1]-10),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)
return img
def detect_common_objects(image, confidence=0.5, nms_thresh=0.3):
Height, Width = image.shape[:2]
scale = 0.00392
global classes
global dest_dir
#all the weights and the neural network algorithm are already
preconfigured #as we are using YOLO
#this part of the script just downloads the YOLO files
config_file_name = 'yolov3.cfg'
config_file_abs_path = dest_dir + os.path.sep + config_file_name
weights_file_name = 'yolov3.weights'
weights_file_abs_path = dest_dir + os.path.sep + weights_file_name
url = 'https://github.com/Nico31415/Drowning
Detector/raw/master/yolov3.cfg'
if not os.path.exists(config_file_abs_path):
download_file(url=url, file_name=config_file_name, dest_dir=dest_dir)
url = 'https://pjreddie.com/media/files/yolov3.weights'
if not os.path.exists(weights_file_abs_path):
download_file(url=url, file_name=weights_file_name,
dest_dir=dest_dir)
global initialize
global net
if initialize:
classes = populate_class_labels()
net = cv2.dnn.readNet(weights_file_abs_path, config_file_abs_path)
initialize = False
blob = cv2.dnn.blobFromImage(image, scale, (416,416), (0,0,0), True,
crop=False)
net.setInput(blob)
```

```
outs = net.forward(get_output_layers(net))
class_ids = []
confidences = []
boxes = []
for out in outs:
for detection in out:
scores = detection[5:]
class_id = np.argmax(scores)
max_conf = scores[class_id]
if max_conf > confidence:
center_x = int(detection[0] * Width)
center_y = int(detection[1] * Height)
w = int(detection[2] * Width)
h = int(detection[3] * Height)
x = center_x - w / 2
y = center_y - h / 2
class_ids.append(class_id)
confidences.append(float(max_conf))
boxes.append([x, y, w, h]) indices = cv2.dnn.NMSBoxes(boxes, confidences, confidence, nms_thresh)
bbox = []
label = []
conf = []
for i in indices:
i = i[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
```

Utils.py

```
import requests
import progressbar as pb
import os
def download_file(url, file_name, dest_dir):
if not os.path.exists(dest_dir):
os.makedirs(dest_dir)
full_path_to_file = dest_dir + os.path.sep + file_name
if os.path.exists(dest_dir + os.path.sep + file_name):
return full_path_to_file
print("Downloading " + file_name + " from " + url)
try:
```

```
r = requests.get(url, allow_redirects=True, stream=True)
except:
print("Could not establish connection. Download failed")
return None
file_size = int(r.headers['Content-Length'])
chunk_size = 1024
num_bars = round(file_size / chunk_size)
bar = pb.ProgressBar(maxval=num_bars).start()
if r.status_code != requests.codes.ok:
print("Error occurred while downloading file")
return None count = 0
with open(full_path_to_file, 'wb') as file:
for chunk in r.iter_content(chunk_size=chunk_size):
file.write(chunk)
bar.update(count)
count +=1
return full_path_to_file
App.py:
import time
import cv2
import numpy as np
from cloudant.client import Cloudant
from flask import Flask, request, render_template, redirect, url_for
from playsound import playsound
import cvlib as cv
from cvlib.object_detection import draw_bbox
# Loading the model
# Authenticate using an IAM API key
client = Cloudant.iam('5a1ffd26-d995-410e-af77-546fb6498fd8-
bluemix','5rUgrIfVeYtIyTqJ0hAvTOvIYvJDNiKlr-sDUHQRcLnN', connect=True)
# Create a database using an initialized client
my_database = client.create_database('my_database')
app=Flask(__name__)
#default home page or route
@app.route('/')
def index():
return render_template('index.html')
@app.route('/index.html')
def home():
return render_template("index.html")
#registration page@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], # Setting _id is optional
'name': x[0],
'psw':x[2]
}
print(data)
query = {'_id': {'$eq': data['_id']}}
docs = my_database.get_query_result(query)
print(docs)
print(len(docs.all()))
if(len(docs.all())==0):
url = my_database.create_document(data)
#response = requests.get(url)
return render_template('register.html', pred="Registration
Successful, please login using your details")
else:
return render_template('register.html', pred="You are already a
member, please login using your details")
#login page
@app.route('/login')
def login():
return render_template('login.html')
@app.route('/afterlogin',methods=['POST'])
def afterlogin():
user = request.form['_id']
passw = request.form['psw']
print(user,passw)
query = {'_id': {'$eq': user}}
docs = my_database.get_query_result(query)
print(docs)
print(len(docs.all()))
if(len(docs.all())==0):
return render_template('login.html', pred="The username is not
found.")
else:
if((user==docs[0][0]['_id'] and passw==docs[0][0]['psw'])):
return redirect(url_for('prediction'))
else: print('Invalid User')
@app.route('/logout')
def logout():
return render_template('logout.html')
@app.route('/prediction')
def prediction():
return render_template('prediction.html')
@app.route('/result',methods=["GET","POST"])
```

```
def res():
webcam = cv2.VideoCapture('drowning.mp4')
if not webcam.isOpened():
print("Could not open webcam")
exit()
t0 = time.time() #gives time in seconds after 1970
#variable dcount stands for how many seconds the person has been standing
still for
centre0 = np.zeros(2)
isDrowning = False
#this loop happens approximately every 1 second, so if a person doesn't
move,
#or moves very little for 10seconds, we can say they are drowning
#loop through frames
while webcam.isOpened():
# read frame from webcam
status, frame = webcam.read()
if not status:
print("Could not read frame")
exit()
# apply object detection
bbox, label, conf = cv.detect_common_objects(frame)
#simplifying for only 1 person
#s = (len(bbox), 2)
if(len(bbox)>0):
bbox0 = bbox[0]
#centre = np.zeros(s)
centre = [0,0]
#for i in range(0, len(bbox)):
#centre[i]
=[(bbox[i][0]+bbox[i][2])/2,(bbox[i][1]+bbox[i][3])/2 ]
centre =[(bbox0[0]+bbox0[2])/2,(bbox0[1]+bbox0[3])/2 ]
#make vertical and horizontal movement variables
hmov = abs(centre[0]-centre0[0])
vmov = abs(centre[1]-centre0[1])
```

GitHub & Project Demo Link

https://github.com/IBM-EPBL/IBM-Project-35674-1660287492

Project Demo Link:

https://youtu.be/sP4QKmR9kwk