

PROJECT REPORT

Team ID	PNT2022TMID16644
Project Name	VirtualEye-Lifeguard for Swimming Pools to Detect the Active Drowning

CHAPTER 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. 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.

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

Swim eye is computer vision detection system for the prevention of drowning incidents in swimming pool. Swim eye works like an “extra lifeguard” under the water of your pool. Life guard of swimming pools to **detect activedrowning**. In this project we will detect the drowning person using yolov3.

To Evaluate the currentliterature on DrowningDetection Systems, including their use in indoor pool environments along with interaction with traditional lifeguarding.

CHAPTER 2

LITERATURE SURVEY

2.1 Existing Problem

Exposure to chlorine and chloramines can cause an eye infection called chemical conjunctivitis (pink eye caused by chemical irritants). Chloramines are chemicals that are formed when chlorine binds to the bodily fluids people bring into the pool. 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.

Currently, there is little qualitative or quantitative research analysing the experiences of lifeguards themselves relating to this concept

2.2 References

[1] <https://www.linkedin.com/company/aqua-teik>

[2] Journal of Computational Information Systems 9: 21

(2013) 8619{8627 Available at <http://www.Jofcis.com>

[3] International Journal of Innovative Research in

Computer and Communication Engineering (An ISO

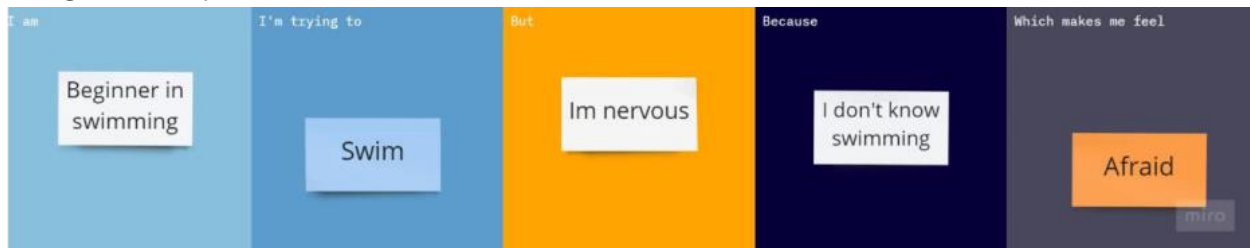
3297: 2007 Certified Organization) Vol. 3, Special Issue

2, March 2015

2.3 Problem Statement Definitions :

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.

PROBLEM 1:



PROBLEM 2:



PROBLEM 3:

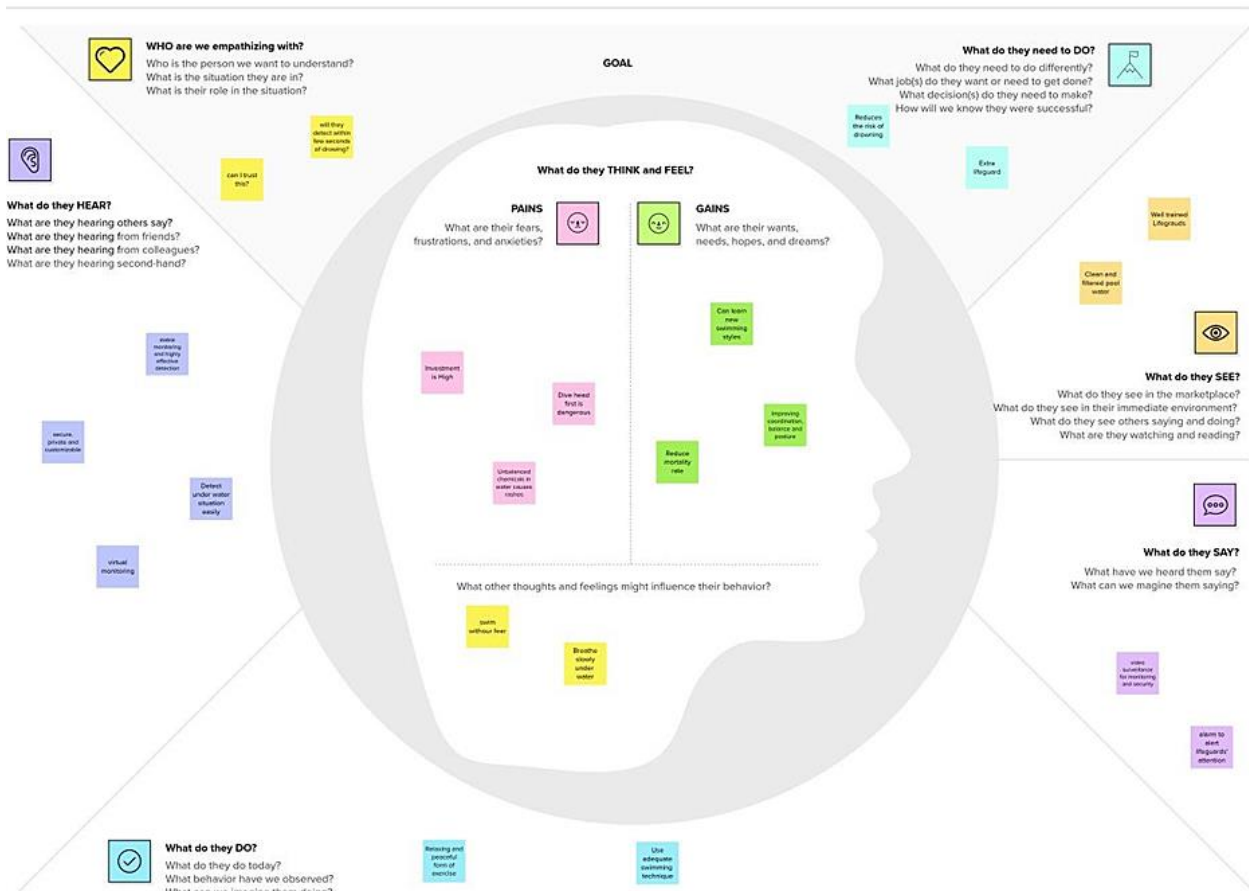


CHAPTER 3

IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

In this activity you are expected to prepare the empathy map canvas to capture the user Pains & Gains, Prepare list of problem statements.



3.2 Ideation and Brainstorming

In this activity you are expected to list the ideas (at least 4 per each team member) by organizing the brainstorming session and prioritize the top 3 ideas based on the feasibility & importance.

3.3 Proposed Solution

In this activity you are expected to prepare the proposed solution document, which includes the novelty, feasibility of idea, business model, social impact, scalability of solution, etc.

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Swimming is one of the best exercises that help people to reduce stress in this urban lifestyle. Even with a lifeguard on duty, swimmers may still have trouble in underwater or in part of the pool beyond the lifeguard's field of view.
2.	Idea / Solution description	We make use of Artificial Intelligence in this project. We install the camera underwater to detect the drowning cases. Using deep learning, images can be recognized. When the images are detected, it triggers the alarm to alert the Life Guard who rescues the drowning people.

3.	Novelty / Uniqueness	Our project Novelty is our system software, which tracks the position and the location of a drowning person with the use of the YOLO Algorithm. It has high accuracy and fast detection speed, So it helps lifeguard to save people within few seconds.
4.	Social Impact/ Customer Satisfaction	Higher death rates occur due to drowning and are also the third leading cause of unexpected deaths, especially among children under the age of five. Our major aim is to overcome this conflict of drowning. This will have a greater social impact.
5.	Business Model (Revenue Model)	Software-based approach is used for gaining more profit. It is extremely useful to lifeguards, swimmers and business operators. The inbuilt feature helps to increase the revenue of the organisation.
6.	Scalability of the Solution	The software system created can be used by the company, who manages the swimming pools. IBM cloud server is used to store and maintain the data to ensure the safety of the swimmers.

3.4 Proposed Solution Fit

In this activity you are expected to prepare problem - solution fit document and submit for review.

1. USER SEGMENT(S)

CS

Person who swim in the pool are meant to be constantly kept an eye over them by visual based monitoring system.

2. JOBS-TO-BE-DONE / PROBLEMS

J&P

- Many deaths account for the third cause of unplanned death globally about 1.2M cases per Year.
- Existing visual based monitoring systems are too economical and these are needed to environment.

Focus on J&P, tap into BE, understand RC

6. USER CONSTRAINTS

CC

- Cost for initial investment and maintenance.
- Constant network connection.
- Camera misunderstanding normal swimming actions to be abnormal.

5. AVAILABLE SOLUTIONS

AS

- Detects and prevents active drowning.
- Setting up of camera and monitoring each and every person swimming in the pool setting an alarm to notify the Lifeguard.

9. PROBLEM ROOT CAUSE

RC

- Anticipation over all the other system happens when one device fails to do its service.
- People think that the camera that is set up to monitor the persons who are swimming are of no proper and accurate use.

7. BEHAVIOUR

BE

- The customer will exhibit his behavior until an authenticated - application serves its purpose rightly.
- The customer believes more in a manual monitoring system rather than a visual monitoring system.

Focus on J&P, tap into BE, understand RC

3. TRIGGERS

TR

- Economical installation cost also plays a pivotal role.
- The customer is triggered by their surrounding talking about this approach of detecting and preventing active drowning.
-

4. EMOTIONS: BEFORE / AFTER

EM

BEFORE : Fear of unprotected swimming
AFTER : Fearless and satisfactory swimming experiences

10. YOUR SOLUTION

SL

- The proposed system makes a unique attempt to evaluate swimmers condition by analyzing their motion and shape features through visual based monitoring device and an alarm to alert, and provides solution in detecting drowning incidents.
- While challenging in many aspects, a successful system will bring inestimable value in saving human lives.

8. CHANNELS of BEHAVIOUR

8.1 ONLINE

- Develop an application and provide all sort of assistance to the users regarding the virtual eye.

8.2 OFFLINE

- Provide quality safety wares while swimming.

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 Functional Requirement

In this activity you are expected to prepare the functional requirement document.

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 without Creating any disturbance to the people in the swimming pool.
FR-2	User registration	Register via Email/Phone number and get verified for further use
FR-3	Deduction	Either not moving or in unconscious state
FR-4	Support	Take swim tubes or take the help of rescuer.
FR-5	Alert	Set alarm and send message through the application to life guard.
FR-6	Output	Vision based monitor Image, position and movement detection Drowning is detected Rescue drowning people by LifeGuard

4.2 Non Functional Requirement

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.

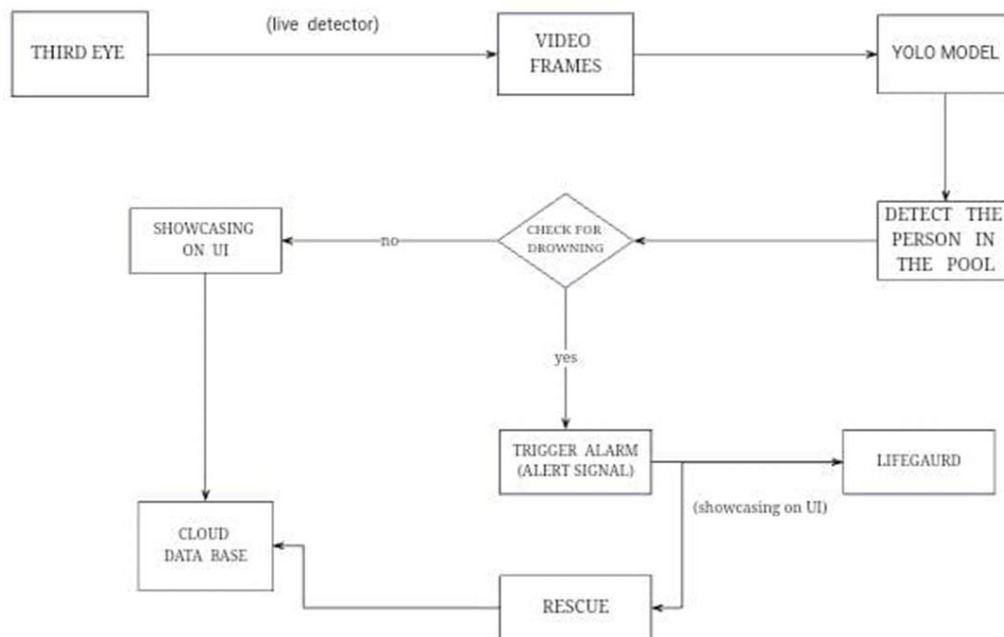
NFR-2	Security	Lifeguards should be aware of the alert message to save the life of the swimmer.
NFR-3	Reliability	Virtual eye lifeguard triggers an immediate alarm if a swimmer is in peril, helping to avoid panic even in critical situations.
NFR-4	Performance	The alarm is triggered when the swimmer is detected as drowning
NFR-5	Availability	Equipment and accessories include lifesaver rings, inflatable vests, a Shepherd's Crook, life hooks, spineboards, rescue tubes, and a first aid kit. Remember to keep them accessible to quickly pull someone from the water safely.
NFR-6	Scalability	Virtual eye lifeguard detects potential drownings and promptly notifies you. It features the latest artificial intelligence technology and adapts to the needs of the user.

CHAPTER 5

PROJECT DESIGN

5.1 Data Flow Diagram

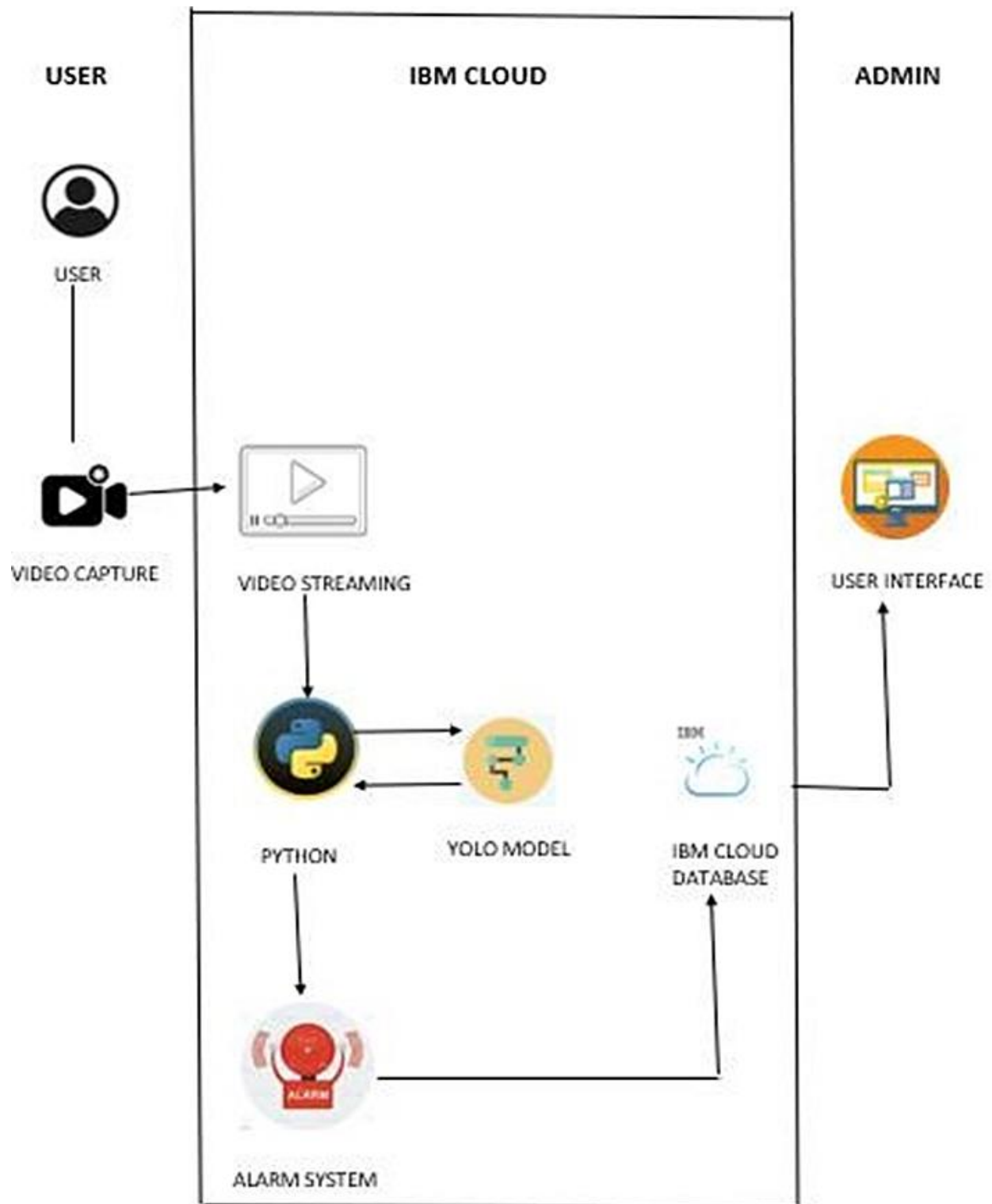
In this activity you are expected to prepare the data flow diagrams and submit for review. A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2 Solution & Technical architecture

In this activity you are expected to prepare solution architecture document and submit for review.

In this activity you are expected to draw the technology architecture diagram.



5.3 User Stories

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

User Type	Functional Requirement(Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Pool owner)	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 findthe drowning persons by using the drowning detection system	I wouldreceive an alertif a person is drowning	High	Sprint-1
	Notify the lifeguard	USN-3	As a user, I can notifythe lifeguard whenthe system detects a drowning person	I can set up an alarm that would notifythe lifeguard	High	Sprint-2
Customer (Lifeguard)	Rescue people	USN-4	As a user, I can rescuethedrowning persons from the pool	I can save the drowning person	High	Sprint-2
Customer (Swimmers)	Safety	USN-5	As a user, I can swimwithout the fearof drowning	I can swim safelywith 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	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

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
Administrator	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

CHAPTER 6

PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

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	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

Project Tracker, Velocity & Burndown Chart:

Velocity:

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

For Sprint-1 the Average Velocity (AV) is: $AV = \text{Sprint Duration} / \text{velocity} = 8 / 6 = 1.3V$

For Sprint-2 the Average Velocity (AV) is: $AV = \text{Sprint Duration} / \text{velocity} = 14 / 6 = 2.3V$

For Sprint-3 the Average Velocity (AV) is: $AV = \text{Sprint Duration} / \text{velocity} = 16 / 6 = 2.6V$

For Sprint-4 the Average Velocity (AV) is: $AV = \text{Sprint Duration} / \text{velocity} = 12 / 6 = 2.0V$

TOTAL TEAM AVERAGE VELOCITY = 2.08

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

6.2 Sprint Delivery Schedule

In this activity you are expected to prepare the sprint delivery plan.

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	SNEHA AS
Sprint-1	Registration	VLGFSP -2	As a user,I will receive confirmation email onceI have registered for the application	1	High	NIKHITA M
Sprint-1	Registration	VLGFSP -3	As a user, I can register for the application through Facebook	2	Low	KIRTHIK A S
Sprint-1	Registration	VLGFSP -4	As a user, I canregister for theapplication through Gmail	2	Medium	BHARAT HI S
Sprint-1	Login	VLGFSP -6	As a user,I can log into the application by entering email& password	1	High	SNEHA AS
Sprint-2	Dataset Collect	VLGFSP -11	Collect number of datasets and get accuracy	2	Medium	NIKHITA M
Sprint-2	Pre-processing	VLGFSP -12	The dataset is extracted	2	High	KIRTHIK A S
Sprint-2	Train the model	VLGFSP -13	Train the model.	4	High	BHARAT HI S

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	SNEHA AS
Sprint-3	Detection	VLGFSP-15	Load the trained model.	3	High	NIKHITA M
Sprint-3	Detection	VLGFSP-16	Identify the person by collecting real-time data through a webcam.	5	Medium	KIRTHIK A S
Sprint-3	Detection	VLGFSP-16	classify it by using a trained model to predict the output	8	High	BHARATHI S
Sprint-4	Detection	VLGFSP-17	If person is drowning, the system will ring an alarm to give signal	7	High	SNEHA AS
Sprint-4	Detection	VLGFSP-18	As a User, I can detect the drowning person.	3	Medium	NIKHITA M
Sprint-4	Logout	VLGFSP-19	As a User, I can logout the application.	2	Low	KIRTHIK A S

6.3 Reports from JIRA

Backlog (scrum)

The screenshot shows the Jira Software Backlog view for the project 'VirtualEye - Life Guard for Swimming Pools to Detect Active Drowning'. The interface includes a left sidebar with navigation options: PLANNING (Roadmap, Backlog, Board), DEVELOPMENT (Code), and Project pages (Project pages, Add shortcut, Project settings). The main area displays the Backlog with three sprints:

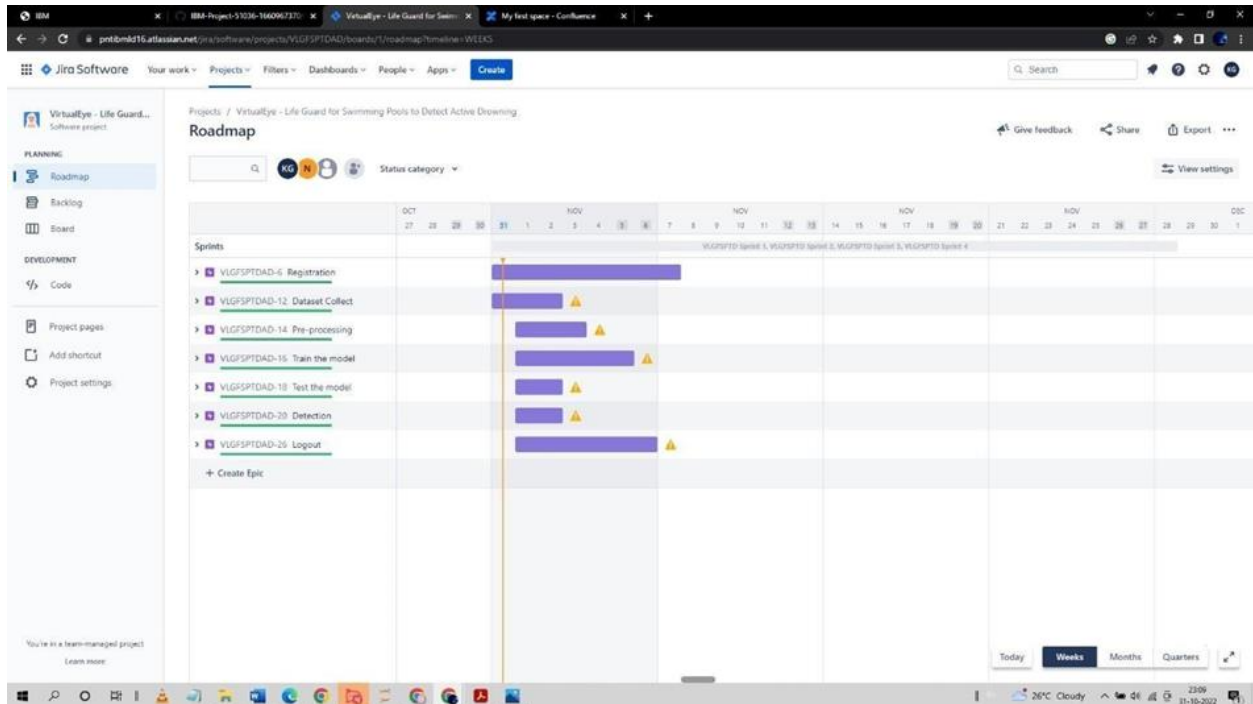
- VLGFSPD Sprint 2** (7 Nov - 14 Nov, 4 issues):
 - VLGFSPDAD-13: Collect number of datasets and get accuracy (DATASET COLLECT)
 - VLGFSPDAD-15: The dataset is extracted (PRE-PROCESSING)
 - VLGFSPDAD-17: Train the model (TRAIN THE MODEL)
 - VLGFSPDAD-19: Test the model (TEST THE MODEL)
- VLGFSPD Sprint 3** (14 Nov - 21 Nov, 3 issues):
 - VLGFSPDAD-21: Load the trained model (DETECTION)
 - VLGFSPDAD-22: Identify the person by collecting real-time data through a webcam (DETECTION)
 - VLGFSPDAD-23: classify it by using a trained model to predict the output (DETECTION)
- VLGFSPD Sprint 4** (21 Nov - 28 Nov, 3 issues):
 - VLGFSPDAD-24: If person is drowning, the system will ring an alarm to give signal (DETECTION)
 - VLGFSPDAD-25: As a User.I can detect the drowning person (DETECTION)
 - VLGFSPDAD-30: As a User.I can logout the applications. LOGOUT

The bottom of the screen shows a Windows taskbar with various application icons and a system tray displaying the date and time (22:48, 31-10-2022).

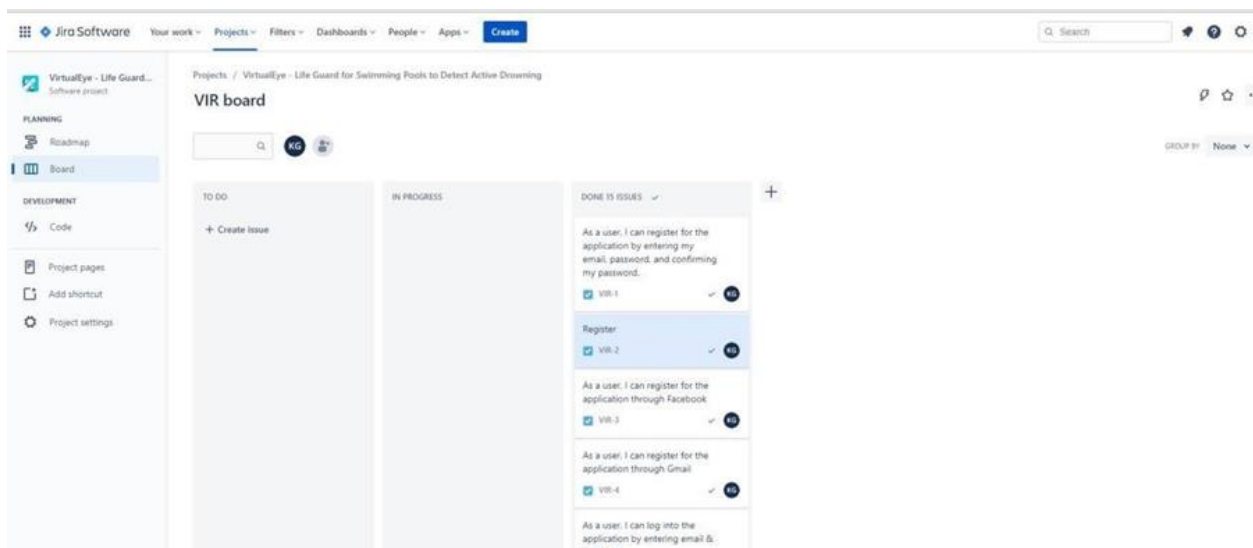
Roadmap

The screenshot shows the Jira Software Roadmap view for the project 'VirtualEye - Life Guard for Swimming Pools to Detect Active Drowning'. The interface includes a left sidebar with navigation options: PLANNING (Roadmap, Backlog, Board), DEVELOPMENT (Code), and Project pages (Project pages, Add shortcut, Project settings). The main area displays the Roadmap with a timeline view showing sprints across months (OCT, NOV, DEC, JAN '21, FEB '21). The roadmap shows a single sprint, VLGFSPDAD-6: Registration, scheduled for November. The bottom of the screen shows a Windows taskbar with various application icons and a system tray displaying the date and time (22:09, 31-10-2022).

Chart



Board (Kanban)



CHAPTER 7

7.1 CODING & SOLUTIONING

```
[net]
# Testing#
batch=1
# subdivisions=1#
Training batch=64
subdivisions=16
width=608 height=608
channels=3
momentum=0.9
decay=0.0005 angle=0
saturation = 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=1
filters=32 size=3
stride=1
pad=1
activation=leaky

# Downsample

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

[convolutional] batch_normalize=1
filters=32 size=1
stride=1
pad=1
activation=leaky

[convolutional]
batch_normalize=1
```

filters=64 size=3
stride=1
pad=1
activation=leaky

[shortcut]from=- 3
activation=linear#

Downsample

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

[convolutional]
batch_normalize=1
filters=64 size=1 stride=1
pad=1
activation=leaky

[convolutional]
batch_normalize=1
filters=128 size=3
stride=1
pad=1
activation=leaky

[shortcut]from=- 3
activation=linear

[convolutional]
batch_normalize=1
filters=64 size=1
stride=1
pad=1
activation=leaky

[convolutional]
batch_normalize=1
filters=128 size=3
stride=1
pad=1
activation=leaky

[shortcut]from=- 3
activation=linear # Downsample
[convolutional]
batch_normalize=1
filters=256size=3
stride=2 pad=1
activation=leaky

[convolutional] batch_normalize=1 filters=128 size=1 stride=1
pad=1 activation=leaky

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

[shortcut]from=- 3
activation=linear

[convolutional]
batch_normalize=
filters=128 size=1 stride=1
pad=1
activation=leaky

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

[shortcut]from=- 3
activation=linear

[convolutional]
batch_normalize=1
filters=128 size=1
stride=1
pad=1
activation=leaky

[convolutional]

batch_normalize=1
filters=256 size=3

stride=1
pad=1
activation=leaky

[shortcut]from=- 3
activation=linear

[convolutional]
batch_normalize=1
filters=128 size=1
stride=1
pad=1
activation=leaky

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

[shortcut]from=- 3
activation=linear

[convolutional]
batch_normalize=1
filters=128 size=1
stride=1
pad=1
activation=leaky

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

[shortcut]from=- 3
activation=linear

[convolutional]
batch_normalize=1

filters=128 size=1
stride=1
pad=1
activation=leaky

[convolutional]

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

[shortcut]from=- 3
activation=linear

[convolutional]
batch_normalize=1
filters=128 size=1
stride=1
pad=1
activation=leaky

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

[shortcut]from=- 3
activation=linear

[convolutional]
batch_normalize=1
filters=128 size=1 s
tride=1
pad=1
activation=leaky

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

[shortcut]from=- 3
activation=linear#

Downsample

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

pad=1
activation=leaky

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

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

[shortcut]from=- 3
activation=linear

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

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

[shortcut]from=- 3
activation=linear

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

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

pad=1
activation=leaky

[shortcut]from=- 3
activation=linear

[convolutional]
batch_normalize=1
filters=256 size=1 s
tride=1
pad=1
activation=leaky

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

[shortcut]from=- 3
activation=linear

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

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

stride=1
pad=1
activation=leaky

[shortcut]from=- 3
activation=linear

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

[convolutional]

batch_normalize=1
filters=512 size=3
stride=1
pad=1
activation=leaky

[shortcut]from=- 3
activation=linear

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

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

[shortcut]from=- 3
activation=linear

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

pad=1
activation=leaky

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

[shortcut]from=- 3
activation=linear#

Downsample

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

stride=2
pad=1
activation=leaky

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

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

[shortcut]from=- 3
activation=linear

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

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

```
[shortcut]from=- 3
activation=linear
```

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

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

```
activation=leaky
```

```
[shortcut]from=- 3
activation=linear
```

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

```
[convolutional]
  batch_normalize=1 f
  ilters=1024 size=3
  stride=1
  pad=1
  activation=leaky
```

```
[shortcut]from=- 3
activation=linear #####
[convolutional]
```

batch_normalize=1
filters=512 size=1
stride=1
pad=1
activation=leaky

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

pad=1
filters=1024
activation=leaky

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

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

activation=leaky

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

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

[convolutional] size=1
stride=1
pad=1

filters=255
activation=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 = 1 random=1

[route] layers = -4

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

[upsample] stride=2

[route]
layers = -1, 61

[convolutional]

batch_normalize=1
filters=256 size=1
stride=1
pad=1
activation=leaky

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

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

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

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

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

```
[convolutional]size=1
stride=1
pad=1
filters=255
activation=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 = 1 random=1
```

```
[route] layers = -4
```

```
[convolutional]
batch_normalize=1
filters=128 size=1
stride=1
pad=1
activation=leaky
```

```
[upsample] stride=2
```


[route]
layers = -1, 36

[convolutional]
batch_normalize=1
filters=128 size=1
stride=1
pad=1
activation=leaky

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

[convolutional]
batch_normalize=1
filters=128 size=1
stride=1
pad=1
activation=leaky

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

pad=1
filters=256
activation=leaky

[convolutional]
batch_normalize=1
filters=128 size=1
stride=1
pad=1
activation=leaky

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

```
filters=256
activation=leaky
```

```
[convolutional]size=1
stride=1
pad=1
filters=255
activation=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 = 1 random=1
```

7.2 Feature 2

```
#import necessary packagesimport cv2
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 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 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 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 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  
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

```

CHAPTER 8

TESTING

8.1 TEST CASES

Test case ID	Feature Type		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	1.Enter URL and click go 2.Click on My Account dropdown button 3.Verify login/Signup popup displayed or not	Login.html	Login/Signup popup should display	Working as
LoginPage_TC_002		Home Page	Verify the UI elements in Login/Signup popup	1.Enter URL and click go 2.Click on My Account dropdown 3.Verify login/Signup 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 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_003	Functional	Home page	Verify user is able to log into application with Valid credentials	1.Enter URL and click go 2.Click on My Account dropdown 3.Enter Valid username/email in Email text 4.Enter valid password in password text box 5. Click On In button	Username:lax@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 credentials	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 In button	Username:lax password:lax26	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 credentials	1-Enter URL and click go 2.Click On My Account dropdown 3.Enter Valid username/email in Email text box 4.Enter Invalid password in password text box 5.Click on In button	username:lax26@mail password:lax26	Application should show 'incorrect email or password' validation message.	working as
LoginPage_TC_005	Functional	Login page	Verify user is able to into application with Invalid credentials	1.Enter URL and click go 2.Click on My Account dropdown 3.Enter Invalid username/email in Email text box 4. Enter Invalid password in password text box 5. Click on In button	username:lax26@mail password:1803	Application should show 'incorrect email or password' validation message.	working as
Predictionpage_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

8.2 USER ACCEPTANCE TESTING

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	7	3	1	2	13
Duplicate	1	0	2	0	3
External	2	3	0	1	6
Fixed	10	2	4	10	26
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	77

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	7	0	0	7
Client Application	1	0	0	41
Security	42	0	0	42
Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

CHAPTER 9

RESULTS

9.1 PERFORMANCE METRICS

BASE.HTML

```
<!--DOCTYPE html-->
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>High Quality Facial Recognition</title>
  <link href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css" rel="stylesheet">
  <script src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
  <script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
  <script src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
  <link href="{ { url_for('static', filename='css/main.css') } }" rel="stylesheet">
  <style>
    .bg-dark {
      background-color: #42678c!important;
    }
    #result {
      color: #0a1c4ed1;
    }
  </style>
</head>

<body style="background-color:black";>
  <header id="head" class="header">
    <section id="navbar">
      <h1 class="nav-heading"></i>Virtual Eye</h1>
      <div class="nav--items">
        <ul>
          <li><a href="{ { url_for('index') } }">Home</a></li>
          <li><a href="{ { url_for('logout') } }">Logout</a></li>
          <!-- <li><a href="#about">About</a></li>
          <li><a href="#services">Services</a></li> -->
        </ul>
      </div>
    </section>
  </header>
```



```

<div class="container">
  <div id="content" style="margin-top:2em">
    <div class="container">
      <div class="row">
        <div class="col-sm-6 bd" >
          <h2><em style="color:white;">High Quality Facial
Recognition</em></h2>
          <br>
          <p><h5><i style="color:white;">Emotion Detection Through Facial
Feature Recognition</i></h5></p>
          
          </div>
          <div class="col-sm-6">
            <div>
              <h4 style="color:white;">Upload Image Here</h4>
              <form action = "http://localhost:5000/" id="upload-file" method="post"
enctype="multipart/form-data">
                <label for="imageUpload" class="upload-label">
                  Choose Image
                </label>
                <input type="file" name="image" id="imageUpload"
accept=".png, .jpg, .jpeg,.pdf">
                </form>

                <div class="image-section" style="display:none;">
                  <div class="img-preview">
                    <div id="imagePreview">
                    </div>
                  </div>
                  <div>
                    <button type="button" class="btn btn-info btn-lg " id="btn-
predict">Analyse</button>
                  </div>
                </div>

                <div class="loader" style="display:none;"></div>

                <h3>
                  <span id="result"> </span>
                </h3>

            </div>
          </div>
        </div>
      </div>
    </div>
  </div>

```

```

        </div>
    </div>
</div>
</body>

<footer>
    <script src="{ { url_for('static', filename='js/main.js') } }" type="text/javascript"></script>
</footer>

</html>

```

INDEX.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">

    <!--Bootstrap -->
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">
    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-
KJ3o2DKtIkVYIK3UENzmM7KCKRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.12.9/dist/umd/popper.min.js"
integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
crossorigin="anonymous"></script>

    <script src="https://kit.fontawesome.com/8b9cdc2059.js" crossorigin="anonymous"></script>
    <link
href="https://fonts.googleapis.com/css2?family=Akronim&family=Roboto&display=swap"
rel="stylesheet">
    <link rel="stylesheet" href="../static/style.css">
    <!-- <script defer src="../static/js/main.js"></script> -->

```

```

<title>Virtual Eye</title>
</head>
<body>
  <header id="head" class="header">
    <section id="navbar">
      <h1 class="nav-heading"><i>Virtual Eye</i></h1>
      <div class="nav--items">
        <ul>
          <li><a href="{{ url_for('index')}}">Home</a></li>
          <li><a href="{{ url_for('login')}}">Login</a></li>
          <li><a href="{{ url_for('register')}}">Register</a></li>
          <li><a href="{{ url_for('login')}}">Demo</a></li>
        </ul>
      </div>
    </section>
    <section id="slider">
      <div id="carouselExampleIndicators" class="carousel" data-ride="carousel">
        <ol class="carousel-indicators">
          <li data-target="#carouselExampleIndicators" data-slide-to="0" class="active"></li>
          <li data-target="#carouselExampleIndicators" data-slide-to="1"></li>
          <li data-target="#carouselExampleIndicators" data-slide-to="2"></li>
        </ol>
        <div class="carousel-inner">
          <div class="carousel-item active">
            
          </div>
          <div class="carousel-item">
            
          </div>
          <div class="carousel-item">
            
          </div>
        </div>
        <a class="carousel-control-prev" href="#carouselExampleIndicators" role="button" data-
slide="prev">
          <span class="carousel-control-prev-icon" aria-hidden="true"></span>
          <span class="sr-only">Previous</span>
        </a>
        <a class="carousel-control-next" href="#carouselExampleIndicators" role="button" data-
slide="next">
          <span class="carousel-control-next-icon" aria-hidden="true"></span>
          <span class="sr-only">Next</span>
        </a>
      </div>
    </section>
  </body>
</html>

```

```
</section>
</header>
<section id="about">
  <div class="top">
    <h3 class="title text-muted">
      ABOUT PROJECT
    </h3>
    <div class="line"></div>
  </div>
```

```
<div class="body">
```

```
<div class="left">
```

```
<h2>Problem:</h2>
```

```
<p>
```

Swimming is one of the best exercises that helps people to reduce stress in this urban lifestyle. Swimming pools are found larger in number in the hotels, weekend tourist spots and barely people have in their house backyard. Beginners, especially often feel it difficult to breathe under water and causes breathing trouble which in turn cause a drowning accident. Worldwide, drowning produces a higher rate of mortality without causing injury to children. Children under six of their age are found to be suffering the highest drowning mortality rates worldwide..Such kinds of deaths account for the third cause of unplanned death globally, with about 1.2 million cases yearly.

```
</p>
```

```
</div>
```

```
<div class="left">
```

```
<h2>Solution:</h2>
```

```
<p>
```

To overcome the conflict, a meticulous system is to be implemented along the swimming pools to save the human life. By studying body movement patterns and connecting cameras to an artificial intelligence (AI) system 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 analysing 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 than an alert will be generated to attract lifeguards attention.

```
</p>
```

```
</div>
```

```
</div>
```

```
<div class="bottom">
```

```
<p><b>
```

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

```
</b></p>
```



```

        box-shadow: 0px 8px 4px grey;
        overflow: hidden;
        padding-left: 20px;
        font-family: 'Josefin Sans';
        font-size: 2vw;
        width: 100%;
        height: 8%;
        text-align: center;
    }
}

.topnav {
    overflow: hidden;
    background-color: #333;
}

.topnav-right a {
    float: left;
    color: #f2f2f2;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none;
    font-size: 18px;
}

.topnav-right a:hover {
    background-color: #ddd;
    color: black;
}

.topnav-right a.active {
    background-color: #565961;
    color: white;
}

.topnav-right {
    float: right;
    padding-right: 100px;
}

.login {
    margin-top: -70px;
}

body {
    background-color: #ffffff;
    background-repeat: no-repeat;
    background-size: cover;

```

```
    background-position: 0px 0px;
  }
.login{
    margin-top:100px;
}
form {border: 3px solid #f1f1f1; margin-left:400px;margin-right:400px;}

input[type=text], input[type=email],input[type=number],input[type=password] {
    width: 100%;
    padding: 12px 20px;
    display: inline-block;
    margin-bottom:18px;
    border: 1px solid #ccc;
    box-sizing: border-box;
}

button {
    background-color: #28272c;
    color: white;
    padding: 14px 20px;
    margin-bottom:8px;
    border: none;
    cursor: pointer;
    width: 100%;
    font-weight:bold;
}

button:hover {
    opacity: 0.8;
}

.cancelbtn {
    width: auto;
    padding: 10px 18px;
    background-color: #f44336;
}

.imgcontainer {
    text-align: center;
    margin: 24px 0 12px 0;
}

img.avatar {
    width: 30%;
    border-radius: 50%;
}
```

```
.container {  
    padding: 16px;  
}
```

```
span.psw {  
    float: right;  
    padding-top: 16px;  
}
```

```
/* Change styles for span and cancel button on extra small screens */  
@media screen and (max-width: 300px) {  
    span.psw {  
        display: block;  
        float: none;  
    }  
    .cancelbtn {  
        width: 100%;  
    }  
}
```

```
</style>  
</head>
```

```
<body style="font-family:Montserrat;">
```

```
<div class="header">
```

```
<div style="width:50%;float:left;font-size:2vw;text-align:left;color:white; padding-  
top:1%">Virtual Eye</div>
```

```
<div class="topnav-right" style="padding-top:0.5%;">
```

```
<a href="{{ url_for('index')}}">Home</a>
```

```
<a class="active" href="{{ url_for('login')}}">Login</a>
```

```
<a href="{{ url_for('register')}}">Register</a>
```

```
</div>
```

```
</div>
```

```
<div id="login" class="login">
```

```
<form action="{{ url_for('afterlogin')}}" method="post">
```

```
<div class="imgcontainer">
```

```

```

```
</div>
```



```

        <div class="container">
            <input type="email" placeholder="Enter registered email ID" name="_id"
required><br>

            <input type="password" placeholder="Enter Password" name="psw"
required>

            <button type="submit">Login</button><br>

        {{pred}}
    </div>
</form>

</div>

</body>
</html>

```

LOGOUT.HTML

```

<!DOCTYPE html>
<html >

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Virtual Eye</title>
    <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet'
type='text/css'>

    <link href='https://fonts.googleapis.com/css?family=Merriweather' rel='stylesheet'>
    <link href='https://fonts.googleapis.com/css?family=Josefin Sans' rel='stylesheet'>
    <link href='https://fonts.googleapis.com/css?family=Montserrat' rel='stylesheet'>

<style>
.header {

        top:0;
        margin:0px;
        left: 0px;
        right: 0px;
        position: fixed;
        background-color: #28272c;

```

```

        color: white;
        box-shadow: 0px 8px 4px grey;
        overflow: hidden;
        padding-left: 20px;
        font-family: 'Josefin Sans';
        font-size: 2vw;
        width: 100%;
        height: 8%;
        text-align: center;
    }
    .topnav {
        overflow: hidden;
        background-color: #333;
    }

    .topnav-right a {
        float: left;
        color: #f2f2f2;
        text-align: center;
        padding: 14px 16px;
        text-decoration: none;
        font-size: 18px;
    }

    .topnav-right a:hover {
        background-color: #ddd;
        color: black;
    }

    .topnav-right a.active {
        background-color: #565961;
        color: white;
    }

    .topnav-right {
        float: right;
        padding-right: 100px;
    }

    .login {
        margin-top: -70px;
    }
    body {

        background-color: #ffffff;
        background-repeat: no-repeat;
    }

```

```
background-size:cover;
background-position: 0px 0px;
}
.main{
margin-top:100px;
text-align:center;
}
form { margin-left:400px;margin-right:400px;}

input[type=text], input[type=email],input[type=number],input[type=password] {
width: 100%;
padding: 12px 20px;
display: inline-block;
margin-bottom:18px;
border: 1px solid #ccc;
box-sizing: border-box;
}

button {
background-color: #28272c;
color: white;
padding: 14px 20px;
margin-bottom:8px;
border: none;
cursor: pointer;
width: 20%;
}

button:hover {
opacity: 0.8;
}

.cancelbtn {
width: auto;
padding: 10px 18px;
background-color: #f44336;
}

.imgcontainer {
text-align: center;
margin: 24px 0 12px 0;
}

img.avatar {
width: 30%;
border-radius: 50%;
```

```

}

.container {
  padding: 16px;
}

span.psw {
  float: right;
  padding-top: 16px;
}

/* Change styles for span and cancel button on extra small screens */
@media screen and (max-width: 300px) {
  span.psw {
    display: block;
    float: none;
  }
  .cancelbtn {
    width: 100%;
  }
}

</style>
</head>

<body style="font-family:Montserrat;">

<div class="header">
  <div style="width:50%;float:left;font-size:2vw;text-align:left;color:white; padding-top:1%">Virtual eye</div>
  <div class="topnav-right" style="padding-top:0.5%;">

    <a href="{{ url_for('home')}}">Home</a>
    <a href="{{ url_for('login')}}">Login</a>
    <a href="{{ url_for('register')}}">Register</a>
  </div>
</div>
<div class="main">
<h1>Successfully Logged Out!</h1>
<h3 style="color:#4CAF50">Login for more information</h3>

    <a href="{{ url_for('login') }}"><button type="submit">Login</button></a>
  </form>
</div>

```

```
</body>
</html>
```

PREDICTION.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">
  <!--Bootstrap -->
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">
  <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-
KJ3o2DKtIkVYIK3UENzmM7KChRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
crossorigin="anonymous"></script>

  <script src="https://kit.fontawesome.com/8b9cdc2059.js" crossorigin="anonymous"></script>
  <link
href="https://fonts.googleapis.com/css2?family=Akronim&family=Roboto&display=swap"
rel="stylesheet">
  <link rel="stylesheet" href="../static/style.css">

  <script defer src="../static/js/JScript.js"></script>
  <title>Prediction</title>
</head>
<body>
  <header id="head" class="header">
    <section id="navbar">
      <h1 class="nav-heading">Virtual Eye</h1>
      <div class="nav--items">
        <ul>
```

```

        <li><a href="{{ url_for('index')}}">Home</a></li>
        <li><a href="{{ url_for('logout')}}">Logout</a></li>
    <!-- <li><a href="#about">About</a></li>
    <li><a href="#services">Services</a></li> -->

</ul>
</div>
</section>
</header>
<!-- dataset/Training/metal/metal326.jpg -->
</br>
<section id="prediction">
<h2 class="title text-muted">Virtual Eye- Life Guard for Swimming Pools to Detect Active
Drowning</h1>
<div class="line" style="width: 900px;"></div>
    </section>
    </br>
    <section id="about">

<div class="body">
<div class="left">
    <p>
        Swimming is one of the best exercises that helps people to reduce stress in this urban
        lifestyle. Swimming pools are found larger in number in the hotels, weekend tourist spots and
        barely people have in their house backyard. Beginners, especially often feel it difficult to breathe
        under water and causes breathing trouble which in turn cause a drowning accident. Worldwide,
        drowning produces a higher rate of mortality without causing injury to children. Children under
        six of their age are found to be suffering the highest drowning mortality rates worldwide..Such
        kinds of deaths account for the third cause of unplanned death globally, with about 1.2 million
        cases yearly.

    </p>
</div>
<div class="left">

    <div class="prediction-input">
        
        </br>
        <form id="form" action="/result" method="post" enctype="multipart/form-data">

            <input type="submit" class="submitbtn" value="Click Me! For a Demo"
onclick="drowning.mp4">
        </form>
    </div>
    <h5 style="text-color:Red">
    <b style="text-color:Red">{{ prediction }}<b>

```

```
        </h5>
    </div>
</div>

</section>

    </br></br>

<section id="footer">
    <p>Copyright © 2021. All Rights Reserved</p>

</section>
</body>
</html>
```

REGISTER.HTML

```
<!DOCTYPE html>
<html >

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Virtual Eye</title>
    <link href='https://fonts.googleapis.com/css?family=Pacifico' rel='stylesheet' type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Arimo' rel='stylesheet' type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Hind:300' rel='stylesheet' type='text/css'>
    <link href='https://fonts.googleapis.com/css?family=Open+Sans+Condensed:300' rel='stylesheet'
    type='text/css'>
    <link rel="stylesheet" href="{ { url_for('static', filename='css/style.css') } }">

    <link href='https://fonts.googleapis.com/css?family=Merriweather' rel='stylesheet'>
    <link href='https://fonts.googleapis.com/css?family=Josefin Sans' rel='stylesheet'>
    <link href='https://fonts.googleapis.com/css?family=Montserrat' rel='stylesheet'>

<style>
.header {

                top:0;
                margin:0px;
                left: 0px;
                right: 0px;
                position: fixed;
                background-color: #28272c;
                color: white;
```

```

        box-shadow: 0px 8px 4px grey;
        overflow: hidden;
        padding-left: 20px;
        font-family: 'Josefin Sans';
        font-size: 2vw;
        width: 100%;
        height: 8%;
        text-align: center;
    }
    .topnav {
        overflow: hidden;
        background-color: #333;
    }

    .topnav-right a {
        float: left;
        color: #f2f2f2;
        text-align: center;
        padding: 14px 16px;
        text-decoration: none;
        font-size: 18px;
    }

    .topnav-right a:hover {
        background-color: #ddd;
        color: black;
    }

    .topnav-right a.active {
        background-color: #565961;
        color: white;
    }

    .topnav-right {
        float: right;
        padding-right: 100px;
    }

    .login {
        margin-top: -70px;
    }
    body {

        background-color: #ffffff;
        background-repeat: no-repeat;
        background-size: cover;
    }

```



```
    background-position: 0px 0px;
  }
.login{
    margin-top:100px;
}
form {border: 3px solid #f1f1f1; margin-left:400px;margin-right:400px;}

input[type=text], input[type=email],input[type=number],input[type=password] {
    width: 100%;
    padding: 12px 20px;
    display: inline-block;
    margin-bottom:18px;
    border: 1px solid #ccc;
    box-sizing: border-box;
}

button {
    background-color: #28272c;
    color: white;
    padding: 14px 20px;
    margin-bottom:8px;
    border: none;
    cursor: pointer;
    width: 100%;
}

button:hover {
    opacity: 0.8;
}

.cancelbtn {
    width: auto;
    padding: 10px 18px;
    background-color: #f44336;
}

.imgcontainer {
    text-align: center;
    margin: 24px 0 12px 0;
}

img.avatar {
    width: 30%;
    border-radius: 50%;
}
```

```
.container {  
  padding: 16px;  
}
```

```
span.psw {  
  float: right;  
  padding-top: 16px;  
}
```

```
/* Change styles for span and cancel button on extra small screens */  
@media screen and (max-width: 300px) {  
  span.psw {  
    display: block;  
    float: none;  
  }  
  .cancelbtn {  
    width: 100%;  
  }  
}
```

```
</style>
```

```
</head>
```

```
<body style="font-family:Montserrat;">
```

```
<div class="header">
```

```
<div style="width:50%;float:left;font-size:2vw;text-align:left;color:white; padding-  
top:1%">Virtual Eye</div>
```

```
<div class="topnav-right" >
```

```
<a href="{{ url_for('home')}}" ">Home</a>
```

```
<a href="{{ url_for('login')}}" ">Login</a>
```

```
<a class="active" href="{{ url_for('register')}}" ">Register</a>
```

```
</div>
```

```
</div>
```

```
<div id="login" class="login">
```

```
<form action="{{ url_for('afterreg')}}" method="post">
```

```
<div class="imgcontainer">
```

```

```

```
</div>
```

```
<div class="container">
```

```

required><br>
required><br>
required>

<input type="text" placeholder="Enter Name" name="name"

<input type="email" placeholder="Enter Email ID" name="_id"

<input type="password" placeholder="Enter Password" name="psw"

<button type="submit">Register</button><br>

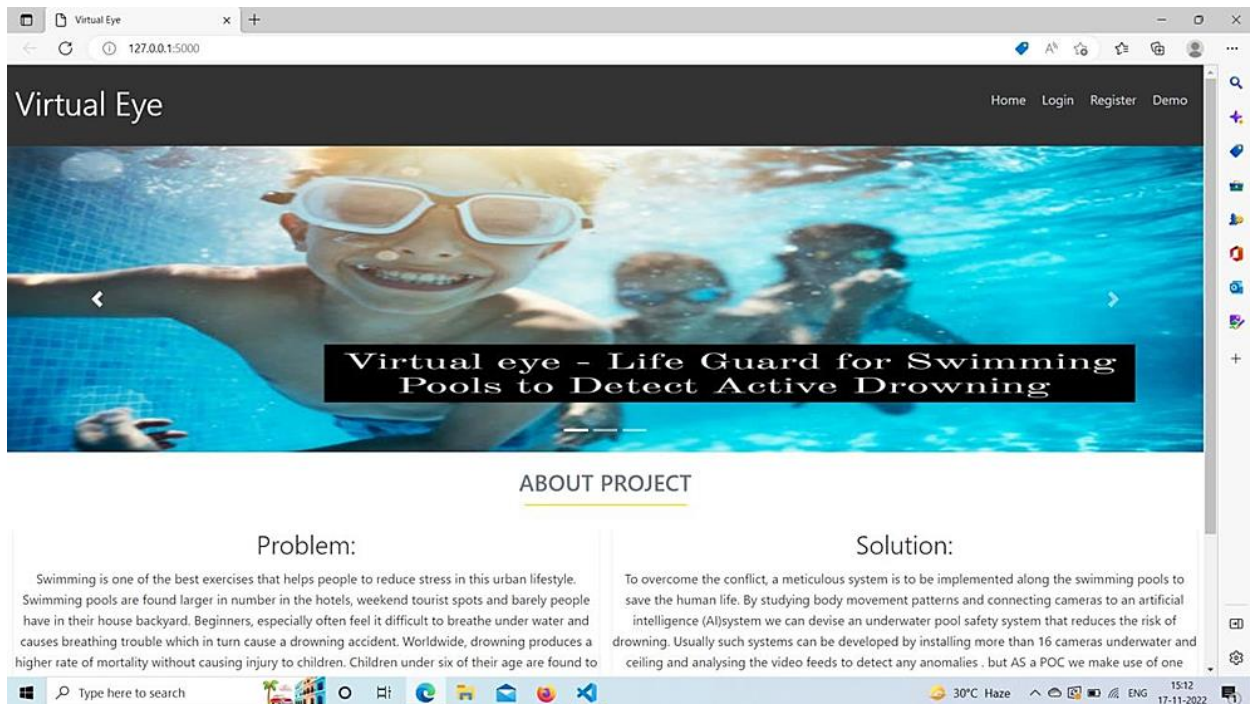
{{pred}}
</div>
<div class="container" style="background-color:#f1f1f1">
  <div class="psw">Already have an account?&nbsp;&nbsp;&nbsp;<a href="{{ url_for('login')
}}">Login</a></div>
</div>
</form>

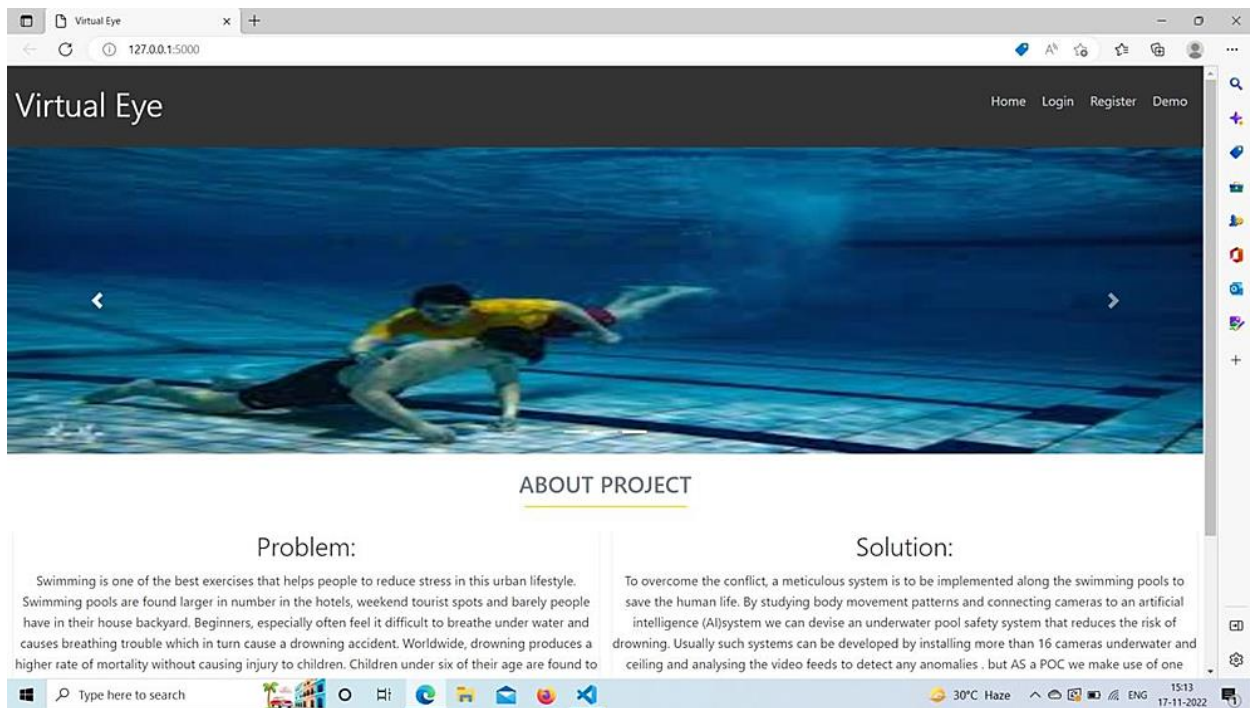
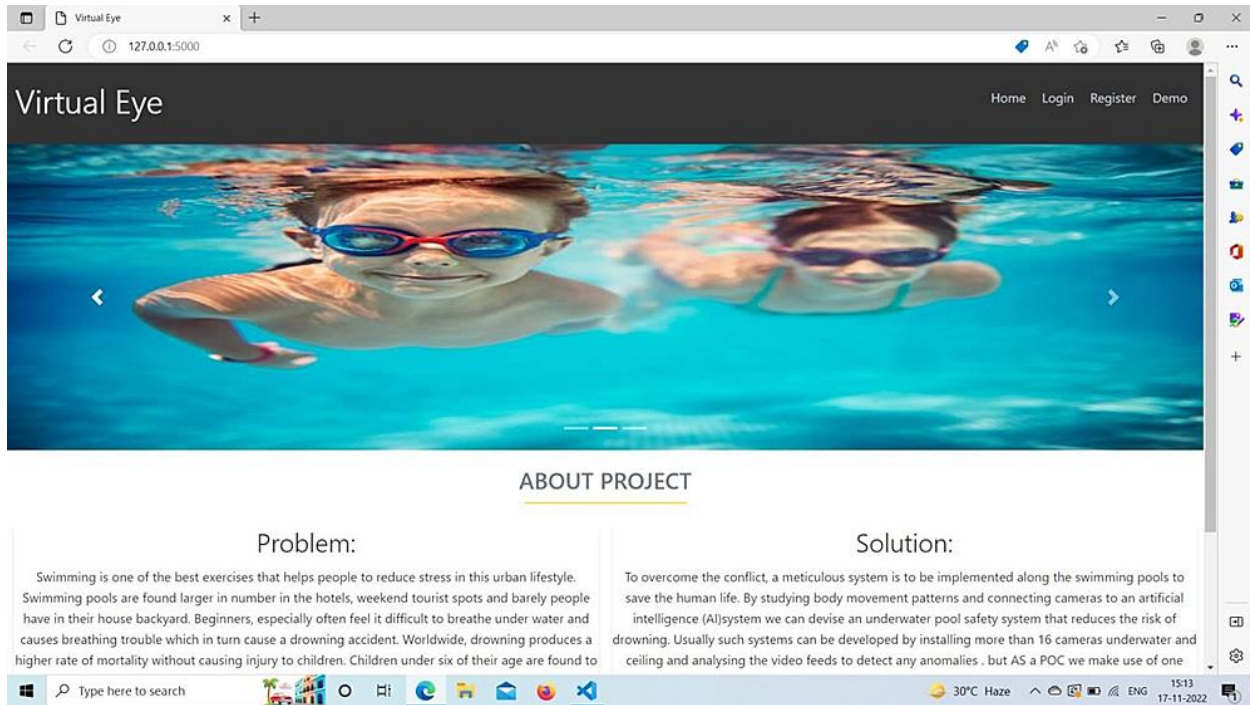
</div>

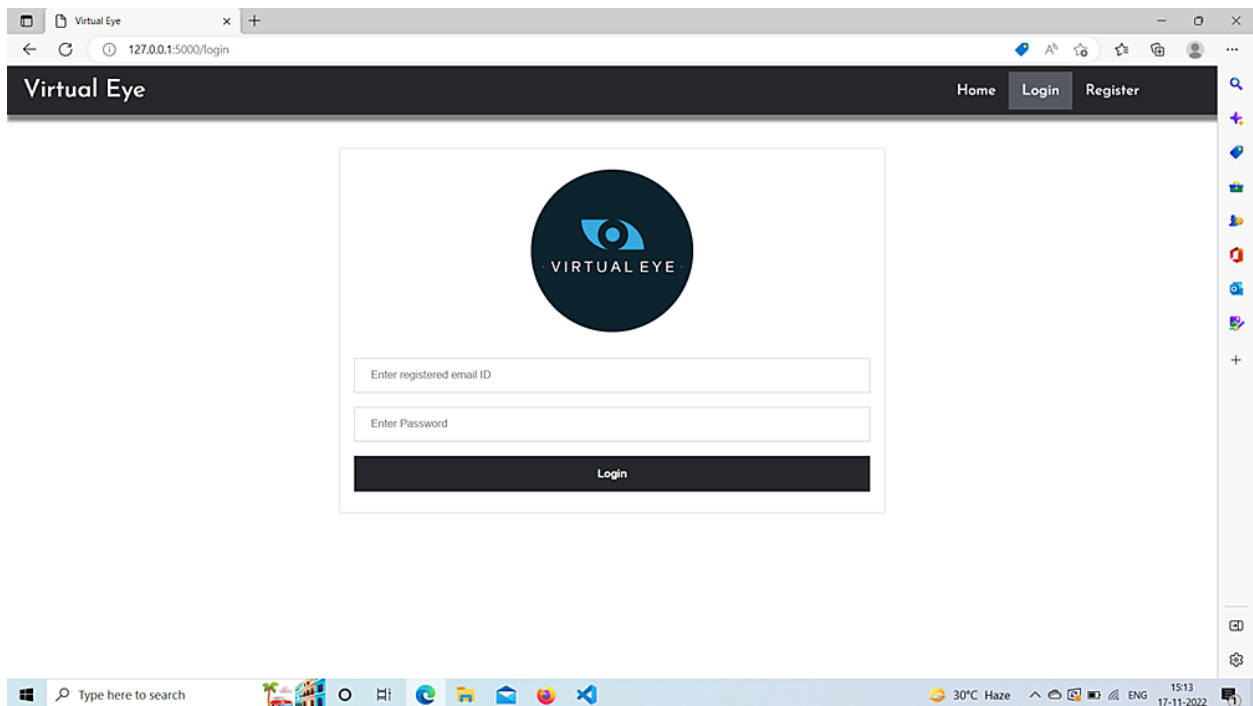
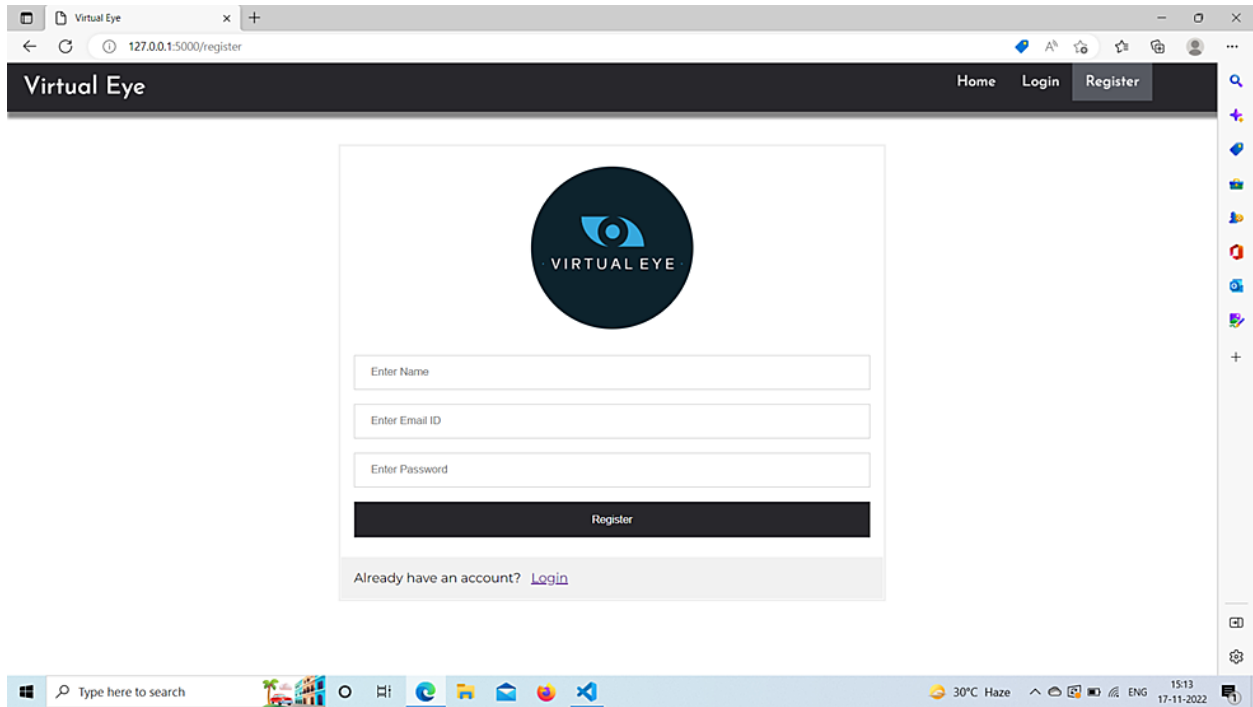
</body>
</html>

```

OUTPUT:







Prediction


127.0.0.1:5000/prediction

HomeLogout

Virtual Eye


Virtual Eye- Life Guard for Swimming Pools to Detect Active Drowning

Swimming is one of the best exercises that helps people to reduce stress in this urban lifestyle. Swimming pools are found larger in number in the hotels, weekend tourist spots and barely people have in their house backyard. Beginners, especially often feel it difficult to breathe under water and causes breathing trouble which in turn cause a drowning accident. Worldwide, drowning produces a higher rate of mortality without causing injury to children. Children under six of their age are found to be suffering the highest drowning mortality rates worldwide. Such kinds of deaths account for the third cause of unplanned death globally, with about 1.2 million cases yearly.



[Click Me! For a Demo](#)

Type here to search



30°C Haze

15:13 17-11-2022

CHAPTER 10

ADVANTAGES AND DISADVANTAGES

➤ ADVANTAGES:

1. Drowning of people should be monitored.
2. Spending more time with family, freedom for safety guards near theSwimming pool.
3. Users feel more comfortable and secure.
4. Swimmers, resort are gain in the financial.
5. Children, adult, pet animal, old age people are used.

➤ DISADVANTAGES:

1. Electricity will be required.
2. This technology is not suitable for uneducated people.
3. Software and hardware requirement will need.

CHAPTER 11

CONCLUSION

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

This is to establish the quantity and positioning of the equipment making up the system such as cameras, central processing unit, alarm tools, and other related equipment. The technical study must also provide a technical drawing of the pool basin, showing areas of 'coverage' and 'non-coverage', as well as the minimum lighting levels required above and below the water surface for the DDS to operate within performance requirements. To carry out the study, a list of factors to consider are given, outlining the variables that make each pool unique such as the architecture, and alarm reception coverage area of mobile devices to be used with the system. With this information all in one document, the technical study can be used to help optimise performance of the system, and forms part of the contract between the supplier and the pool operator. The next area of the standard is the performance requirements. This outlines the requirements needed to pass the regular maintenance testing and performance requirements for normal operation. This section covers the alarm set off time for operational performance, which is to be 15 seconds or less and displayed on the system interface. It also states that the alarm set off time must be built-in and shall not be changeable by staff. The section also discusses the areas covered by the DDS and highlights that each trained staff member must be aware of these areas. Another coverage-related requirement is that the DDS must be able to temporarily create areas where detection is disabled, to manage specific activities such as rescue drills.

CHAPTER 12

FUTURE SCOPE

This lifeguard system consists of three main components, i.e., the drowning detection, the rescuing drone, and the hazardous activity detection. All three components combined will create a system capable of detecting drowning victims, dispatching an inflatable tube using a drone (as depicted in Fig.9) and detecting hazardous activities—eventually becoming an entity that could assist a lifeguard. The system is accessible to its primary user, presumably a pool owner or a lifeguard, in the form of an interface with a sound alarm and an android mobile service that holds the capabilities of receiving Firebase notifications. Confronted with a few of the hardware limitations, such as the use of a single camera and the Jetson Nano over drone in the presence of better-quality hardware, could affect the speed and accuracy of the overall system is becoming a state-of-threat. This limitation could be omitted with the use of multiple cameras that could be placed over drone in emesis in several ground coordinates, increasing the accuracy of the computer vision algorithms. Moreover, due to the inability to fly a drone in extreme weather conditions such as rain, strong winds or lightning, the system is limited to be used under few specifications. As swimming in extreme weather conditions is not preferred either, the system could be further improved to emit a warning signal if a person was to swim in any of the above weather conditions, bypassing the need to fly the drone. Additionally, all the processing is done on the client side of the applications on the Jetson Nano board, preventing any security and privacy issues that might arise due to the sensitive information inputted through the cameras. For future developments convenience wise, the system could benefit by having an additional set of cameras to identify and verify a drowning or a hazardous activity on the premises. Accessibility could also be improved by extending the Android service to be an application both in Android and iOS platforms that could hold the details of each premise individually, making a centralized system that watches over the decentralized pool premises. Both drown and hazardous activity detection could be improved by gathering a night time dataset that increases the accuracy of the data in low light.

CHAPTER 13

APPENDIX

SOURCE CODE

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('8780b82a-5a3b-4da0-a180-a0e1516479f9-bluemix',
'TzYs8u0Q5eoj204gDo2eOEDAuGRhj0fG_9rlZr5SsJUH',
                    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])

# there is still need to tweak the threshold
# this threshold is for checking how much the centre has moved

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('bounding box: ', bbox, 'label: ' label , 'confidence: ' conf[0], 'centre: ', centre)
# print(bbox,label ,conf, centre)
print('bbox: ', bbox, 'centre:', centre, 'centre0:', centre0)
print('Is he drowning: ', isDrowning)

centre0 = centre
# draw bounding box over detected objects

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

```

```

# print('Seconds since last epoch: ', time.time()-t0)

# display output
cv2.imshow("Real-time object detection", out)
if (isDrowning == True):
    playsound('alarm.mp3')
    webcam.release()
    cv2.destroyAllWindows()
    return render_template('prediction.html', prediction="Emergency !!! The Person is
drowning")
    # return render_template('base.html')

# press "Q" to stop
if cv2.waitKey(1) & 0xFF == ord('q'):
    break

# release resources
webcam.release()
cv2.destroyAllWindows()
# return render_template('prediction.html',)

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

```

DETECT.PY

```

import cvlib as cv
from cvlib.object_detection import draw_bbox
import cv2
import time
import numpy as np
from playsound import playsound
#for PiCamera
#from picamera Import PiCamera
#camera = PiCamera
#camera.start_preview()
# open webcam
webcam = cv2.VideoCapture(0)

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])

    #there is still need to tweek the threshold
    #this threshold is for checking how much the centre has moved

```

```

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('bounding box: ', bbox, 'label: ' label , 'confidence: ' conf[0], 'centre: ', centre)
#print(bbox,label ,conf, centre)
print('bbox: ', bbox, 'centre:', centre, 'centre0:', centre0)
print('Is he drowning: ', isDrowning)

centre0 = centre
# draw bounding box over detected objects

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

#print('Seconds since last epoch: ', time.time()-t0)

# display output
cv2.imshow("Real-time object detection", out)
if(isDrowning == True):
    playsound(r'C:\Users\HP\Downloads\alarm.mp3')

# press "Q" to stop
if cv2.waitKey(1) & 0xFF == ord('q'):
    break

# release resources
webcam.release()
cv2.destroyAllWindows()

```

__INIT__.PY

```

from .object_detection import detect_common_objects

```


OBJECT_DETECTION.PY

```
#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 - 1] for i in net.getUnconnectedOutLayers()]
```

```
    #output_layers = [layer_names[i - 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
    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  
    numBars = round(file_size / chunk_size)  
    bar = pb.ProgressBar(maxval=numBars).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
```

GITHUB LINK:

<https://github.com/IBM-EPBL/IBM-Project-27321-1660054111>

PROJECT DEMO LINK:

https://drive.google.com/drive/folders/1nDDOMKAn_4l9M17dOC0AyO1F8pbCPP07