

VIRTUAL EYE LIFE GUARD FOR SWIMMING POOLS TO DETECT DROWNING



A PROJECT REPORT

Submitted by

GOKULA KRISHNAN.S (621718104010)

JIJESH. E (621718104015)

SABARI.M (621718104037)

VIGNESH.P (621718104053)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

MUTHAYAMMAL COLLEGE OF ENGINEERING,

RASIPURAM.

ABSTRACT

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

By studying body movement patterns and connecting cameras to artificial intelligence (AI) systems we can devise an underwater pool safety system that reduces the risk of drowning. Usually, such systems can be developed by installing more than 16 cameras underwater and ceiling and analyzing the video feeds to detect any anomalies. but AS a POC we make use of one camera that streams the video underwater and analyses the position of swimmers to assess the probability of drowning, if it is higher then an alert will be generated to attract lifeguards' attention.

CHAPTER NO	TABLE OF CONTENTS	PAGE NO
	INDEX	
1	INTRODUCTION	
	1.1 Project Overview	
	1.2 Purpose	
2	LITERATURE SURVEY	
	2.1 Existing problem	
	2.2 References	
	2.3 Problem Statement Definition	
3	IDEATION AND PROPOSED SOLUTION	
	3.1 Empathy Map Canvas	
	3.2 Ideation and Brainstorming	
	3.3 Proposed Solution	
	3.4 Problem Solution fit	
4	REQUIREMENT ANALYSIS	
	4.1 Functional Requirement	
	4.2 Non-Functional Requirements	
5	PROJECT DESIGN	
	5.1 Data Flow Diagrams	
	5.2 Solution and Technical Architecture	
	5.3 User Stories	
6	PROJECT PLANNING AND SCHEDULING	
	6.1 Sprint Planning and Estimation	
	6.2 Sprint Delivery Schedule	
	6.3 Reports from JIRA	
7	CODING AND SOLUTIONING	
	7.1 Feature 1	
	7.2 Feature 2	
	7.3 Database Schema(if Applicable)	
8	TESTING	
	8.1 Test Cases	
	8.2 User Acceptance Testing	
9	RESULTS	
	9.1 Performance Metrics	
10	ADVANTAGES & DISADVANTAGES	
11	CONCLUSION	

12	FUTURE SCOPE	
13	APPINDIX	
	Source Code	
	GitHub and Project Demo Link	

INTRODUCTION

1.1 Project overview

Virtual Eye Life Guard is a drowning detection system that detects every dangerous situation and accident. The Virtual Eye 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 Virtual Eye Life Guard represents excellence in drowning detection

Now a days, Swimming pool detector can be used a tool for monitoring and security. Observing public and private site has increasingly become very sensitive issue.Swimming pool detector systems are designed and installed in places such as pools and even dangerous environments. Image processing patterns recognition and machine learning based methods are efficient ways for real time intelligent monitoring of the objects or events of interest. Applying intelligence in Swimming pool detector allows real-time monitoring of places, people and their activities. The tracking approach can change with varying targets and can change with varying targets and change from a single camera to multiple camera configurations

1.2 Purpose

- Virtual Eye lifeguard can provide very stable monitoring and highly effective drowning incident detection.
- Virtual Eye lifeguard is secure, private and customizable to be compliant with regional data protection and privacy regulation
- Virtual Eye lifeguard is suitable for any swimming pool. New pools or existing pools (full of water), all shapes & sizes, and any construction type.
- Virtual Eye lifeguard will deliver the absolute best-quality products, software, sales-support, service and maintenance

- 9

LITERATURE SURVEY

2.1 Existing problem

As per the WHO(World health Organization), drowning is the leading cause of unintentional deaths in the world ,around 372,000 drowning deaths reported annually.

Swimming pool Drowning Deaths and kids It's unbelievable statistic: According to CDC, drowning is the number one cause of unintentional deaths for children between the age 1 to 4.To overcome this problem the surveillance IoT model will helps to avoid maximum cases of death rates.

2.2 References

- <https://onlinelibrary.wiley.com>
- <https://www.angeleye.tech/us/us-lifeguard/>
- <https://swimeye.com/>

2.3 Problem Statement Definition

The person who is swimming in a pool needs to be rescued as soon as possible if he/she is drowning so that he/she does not die and swim without the fear of drowning.

- 5W's

1. Who does the problem affect?

The problem affects a lot of people than we think it does. It affects,

- The person who drowns loses his life.
- The person's kin and kith become traumatized by the loss their loved one.

- The fellow swimmers who used to practice along with the person who drowned get their confidence and passion towards swimming lowered.

2.What is the issue?

Though Swimming is a healthy exercise and popular sport there is always a risk of people drowning. More than the fear of losing a swimming competition the fear of drowning affects a lot of people making them refrain from practicing.

3. When does the issue occur?

The issue may occur during the following scenarios:

- When a person learns swimming.
- When a person goes unconscious in a swimming pool.
- When a person gets exhausted in a swimming pool.

4.Where is the issue occurring?

The issue usually occurs in a swimming pool.

5.Why is it important that we fix the problem?

According to the U.S. Consumer Product Safety Commission, 390 deaths a year on average are attributed to drowning in a swimming pool. If we can fix this problem then it directly saves around 400 lives a year, this is why it is important.

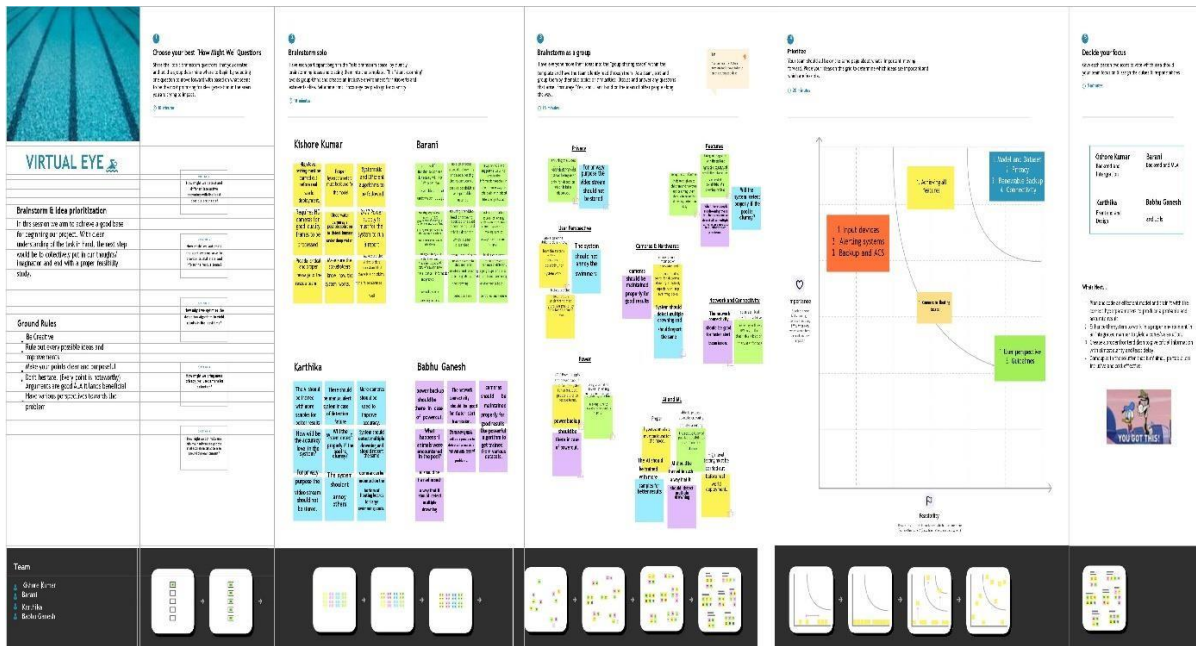
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

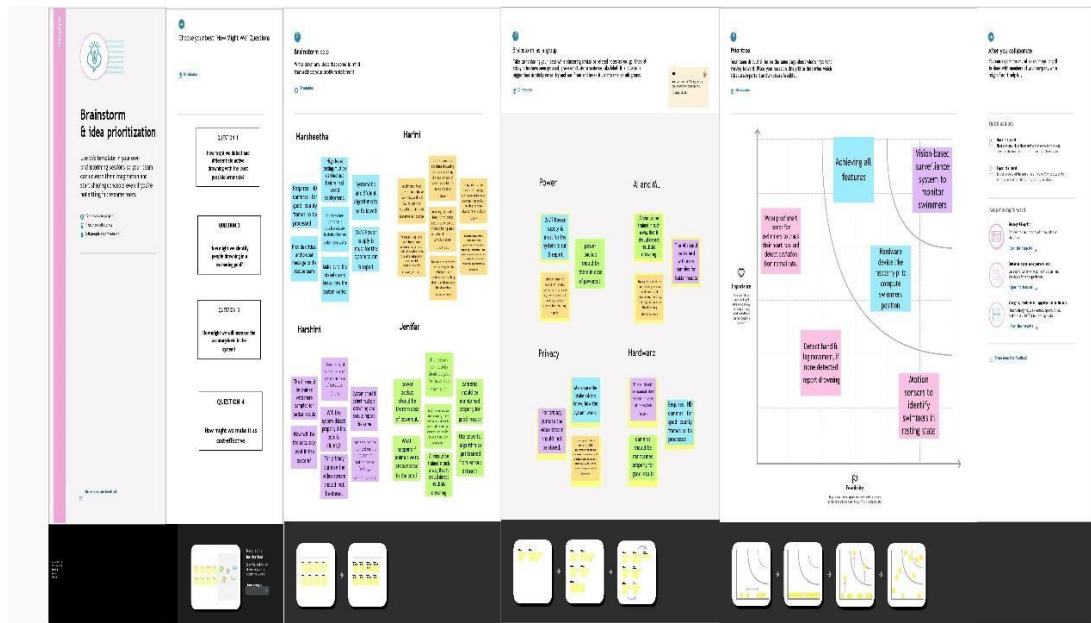


3.2 Ideation & Brainstorming

Ideation:



Brainstorming:



3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Swimming pools are generally places of fun and healthy exercise, but they can be deadly as well. Even with a lifeguard observer on duty, swimmers may still have trouble in underwater or in parts of the pool beyond the lifeguard's field of view.
2.	Idea / Solution description	In this project, we use Artificial Intelligence. We install the cameras in underwater to detect the drowning people. Using deep learning, image can be recognized. If the image is detected, it triggers the alarm to alert the Life Guard who rescue the drowning peoples.
3.	Novelty / Uniqueness	The uniqueness of our system software to track the position and the location of a drowning person. We use YOLO Algorithm. Because of its high accuracy and fast detection speed. So it helps lifeguard to save people within seconds.
4.	Social Impact / Customer Satisfaction	Drowning globally has a higher death rate and is also the third leading cause of unexpected deaths worldwide, especially among children under the age of six. To overcome this conflict our drowning detection system will have an impact on society.
5.	Business Model (Revenue Model)	We can introduce the software-based approach for making a good income. It is extremely useful to lifeguards, swimmers and business operators. The number of features makes it attractive for end users to use our software system.
6.	Scalability of the Solution	Our software system can be used by the company driver who manages the pools. We use the IBM cloud server to collect and maintain the data. We will ensure the safety of the swimmers.

3.4 Problem Solution fit:

Define CS, fit into CC	1. CUSTOMER SEGMENT(S) Children under six	6. CUSTOMER CONSTRAINTS spending power, budget, no cash, network connection, available devices.	5. AVAILABLE SOLUTIONS Fire fighters and trained swimmers	Explore AS, different
	2. JOBS-TO-BE-DONE / PROBLEMS we make use of one camera that streams the video underwater and analyses the position of swimmers to assess the probability of drowning	9. PROBLEM ROOT CAUSE customers have to do it because of the change in luxurious activities have drastically increased and polls have become common everywhere.	7. BEHAVIOUR Install drowning detectors, or call for emergency help	
Focus on J&P, tap into BE, understand RC	3. TRIGGERS Seeing others install virtual eye on their swimming pools	10. YOUR SOLUTION we make use of one camera that streams the video underwater and analyses the position of swimmersto assess the probability of drowning	8. CHANNELS of BEHAVIOUR 8.1 ONLINE Ordering of drowning detectors, or pool lifeguards 8.2 OFFLINE	
	4. EMOTIONS: BEFORE / AFTER Lost and insecure/confident and in control		Implementing them to wear them without fail	

4. REQUIREMENT ANALYSIS

4.1 Functional requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Installation	Needed to be fixed under the water without creating any disturbance to the people in the swimming pool.
FR-2	Deduction	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	Pulse rate sensor	Detect the pulse rate of a swimmer
FR-6	Prior Alert	Send alert message to the lifeguard

4.2 Non-Functional requirements:

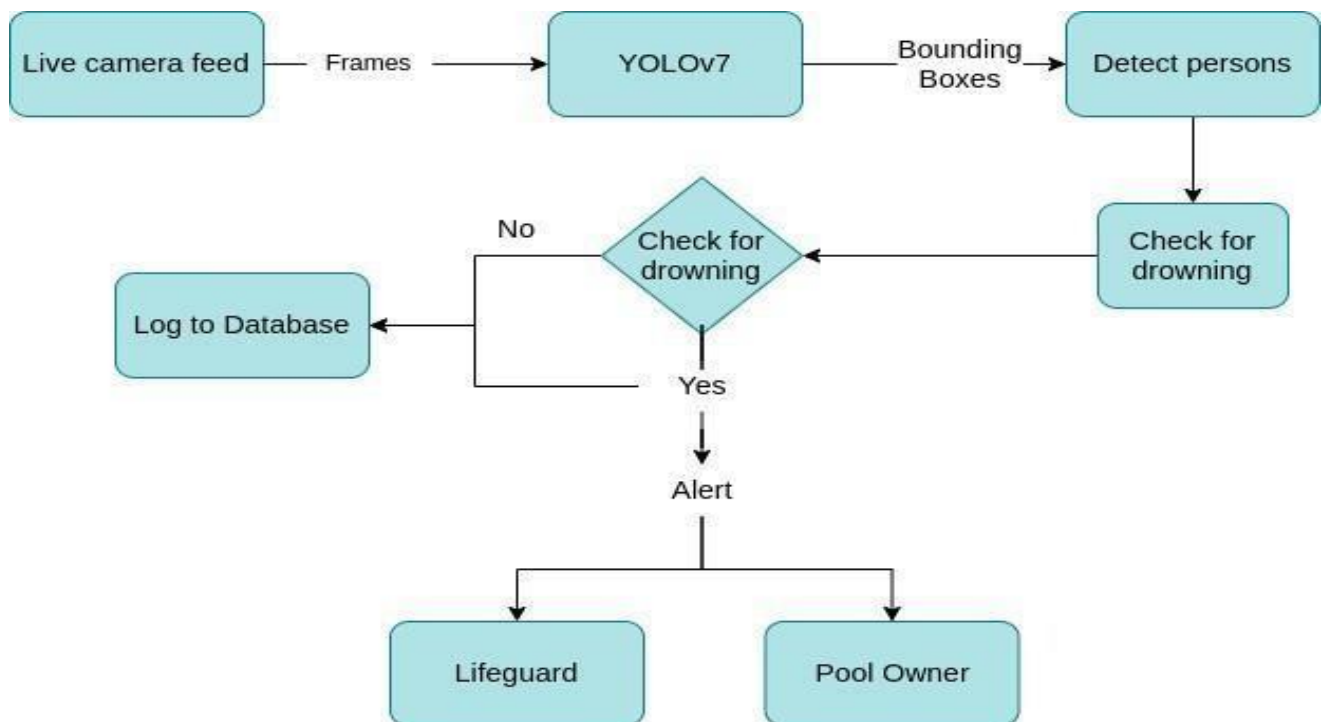
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	To ensure the safety of each and every person present in the pool. A Lifeguard should be present all the time in the pool.
NFR-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 alarm is triggered when the swimmer's pulse rate is decreasing
NFR-5	Availability	Equipment and accessories include lifesaver rings, inflatable vests, a Shepherd's Crook, life hooks, spine boards, 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.

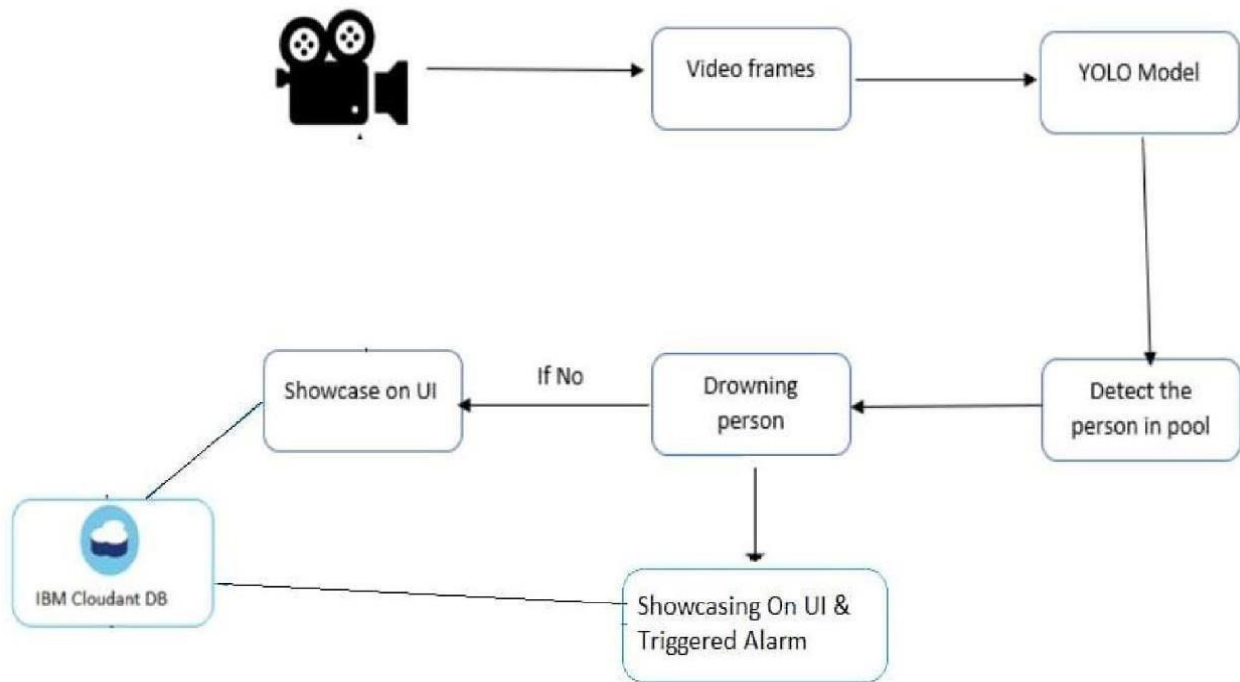
5. PROJECT DESIGN

5.1 Data Flow Diagrams :

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



5.2 Solution & Technical Architecture:



5.3 User Stories:

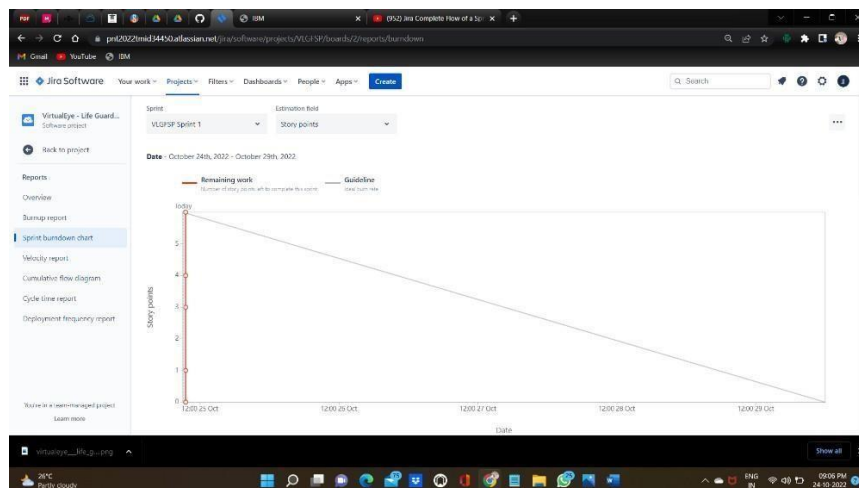
User Stories

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

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation:

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 velocity = $8 / 6 = 1.3V$ For Sprint-2 the

Average Velocity (AV) is: $AV = \text{Sprint Duration} /$

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

Burndown Chart: A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software

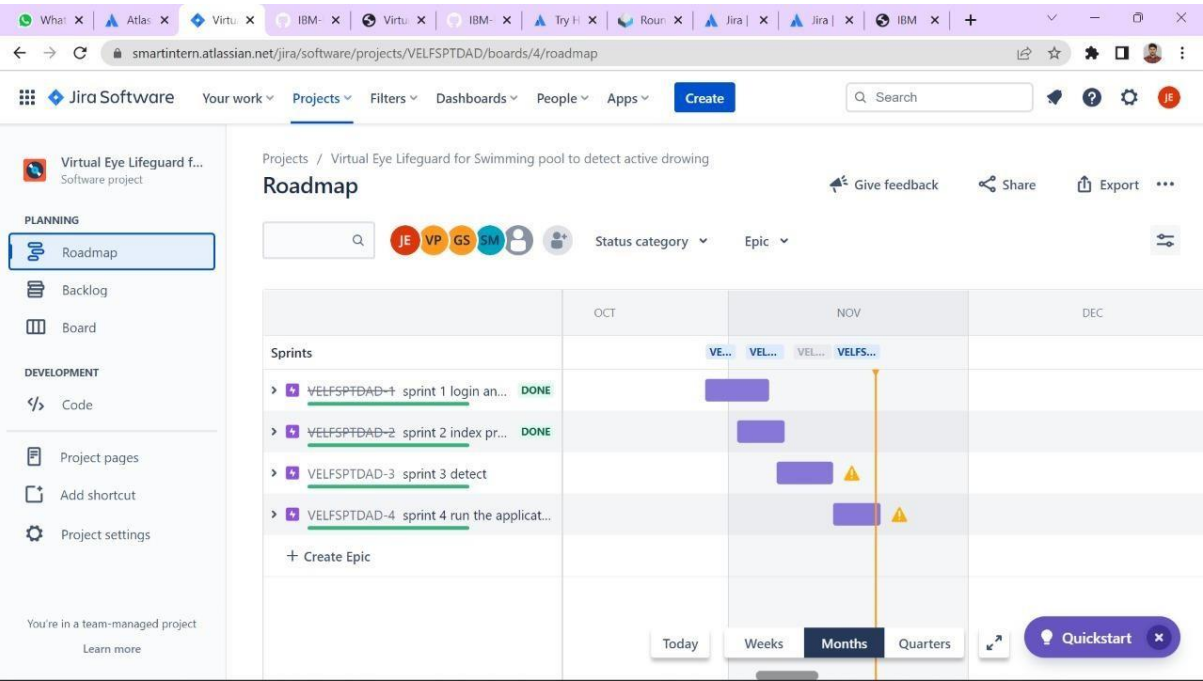
development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

6.2 Sprint Delivery Schedule:

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

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	8	6 Days	24 Oct 2022	29 Oct 2022	6	29 Oct 2022
Sprint-2	14	6 Days	31 Oct 2022	05 Nov 2022	12	05 Nov 2022
Sprint-3	16	6 Days	07 Nov 2022	12 Nov 2022	11	12 Nov 2022
Sprint-4	12	6 Days	14 Nov 2022	19 Nov 2022	12	19 Nov 2022

6.3 Reports from JIRA:



Virtual Eye Lifeguard f...
Software project

PLANNING

- Roadmap
- Backlog
- Board

DEVELOPMENT

- Code
- Project pages
- Add shortcut
- Project settings

You're in a team-managed project
Learn more

Projects / Virtual Eye Lifeguard for Swimming pool to detect active drowning

All sprints

0 days remaining **Complete sprint**

Search

JE SM GS VP Epic Sprint

GROUP BY None Insights

TO DO

IN PROGRESS

DONE 3 ISSUES ✓

- Create login and registration page
SPRINT 1 LOGIN AND REGISTER P...
VELFSPTDAD-5 ✓ SM
- Create index and prediction page
SPRINT 2 INDEX PREDICTION
VELFSPTDAD-9 ✓ GS
- Run the application
VELFSPTDAD-12 ✓ VP

Quickstart

Virtual Eye Lifeguard f...
Software project

PLANNING

- Roadmap
- Backlog
- Board

DEVELOPMENT

- Code
- Project pages
- Add shortcut
- Project settings

You're in a team-managed project
Learn more

Projects / Virtual Eye Lifeguard for Swimming pool to detect active drowning

Roadmap

Give feedback Share Export

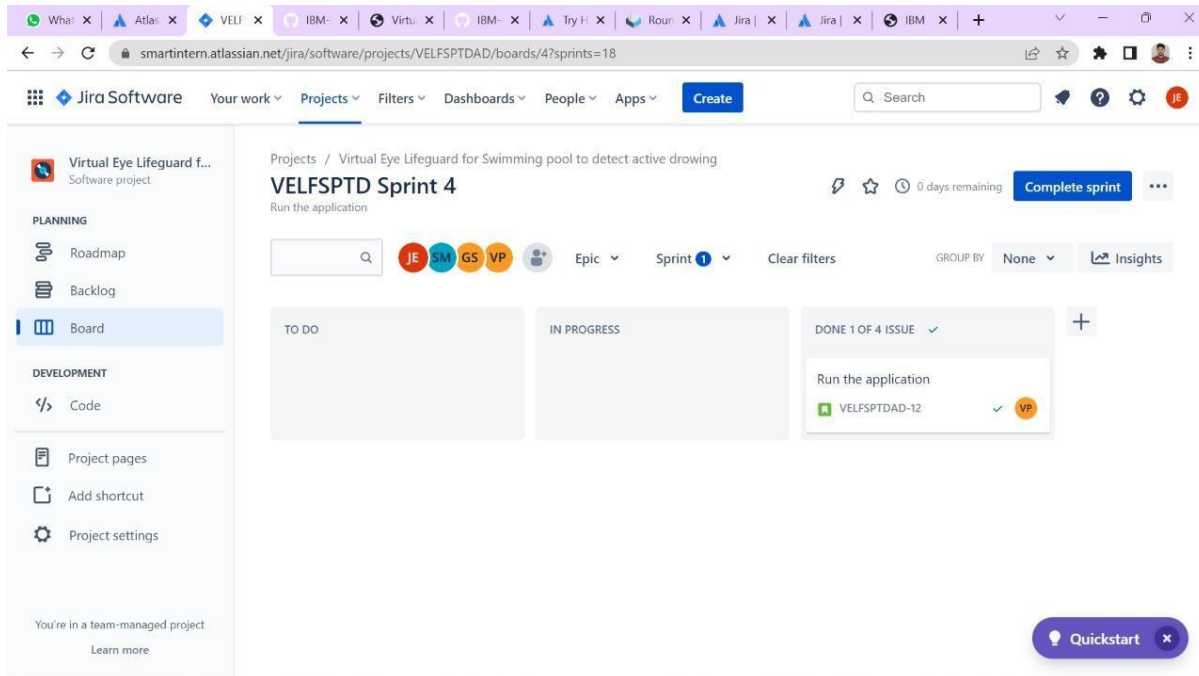
Search

JE VP GS SM Status category Epic

	T	NOV	DEC
Sprints		VE... VEL... VEL... VELFS...	
> VELFSPTDAD-1 sprint 1 login an... DONE			
> VELFSPTDAD-2 sprint 2 index pr... DONE			
> VELFSPTDAD-3 sprint 3 detect			
> VELFSPTDAD-4 sprint 4 run the applicat...			
+ Create Epic			

Today Weeks Months Quarters

Quickstart



7. CODING & SOLUTIONING:

(Explain the features added in the project along with code)

7.1 Feature 1

```
from google.colab import drive
drive.mount('/content/drive')
```

Mounted at /content/drive

```
!unzip '/content/drive/MyDrive/Drowning Classification.v1i.folder.zip'
```

```

import numpy as np
import tensorflow as tf
from tensorflow.keras import layers
from tensorflow.keras.models import Sequential
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
import matplotlib.pyplot as plt
batch_size = 32
img_height = 180
img_width = 180
data_dir = "/content/drive/MyDrive/Drowning Classification.v1i.folder.zip"

# Data augmentation on training variable
train_datagen = ImageDataGenerator(rescale=1./255,
zoom_range=0.2,
horizontal_flip=True)

# Data augmentation on testing variable
test_datagen = ImageDataGenerator(rescale=1./255)
xtrain = train_datagen.flow_from_directory('/content/train',
target_size=(64,64),
class_mode='categorical',
batch_size=100)

    Found 678 images belonging to 2 classes.

xtest = test_datagen.flow_from_directory('/content/test',
target_size=(64,64),
class_mode='categorical',
batch_size=100)

    Found 28 images belonging to 2 classes.

model=Sequential()

from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
model=Sequential()

model.add(Convolution2D(32, (3,3), activation = 'relu', input_shape = (64,64,3) ))

```

```
model.add(MaxPooling2D(pool_size = (2,2)))

model.add(Flatten())

model.add(Dense(300, activation = "relu"))
model.add(Dense(150, activation = "relu"))

model.add(Dense(5, activation = "softmax"))

model.summary()

model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])

print(xtrain.class_indices)

{'Drowning': 0, 'Not Drowning': 1}

model.fit(xtrain, epochs = 0, steps_per_epoch = len(xtrain))

<keras.callbacks.History at 0x7fd28aa70310>

print(xtest.class_indices)
```



```
{'Drowning': 0, 'Not Drowning': 1}
```

```
model.save('Drowning Classification.h5')
```

```
from tensorflow.keras.models import load_model
from keras.preprocessing import image
model=load_model("Drowning Classification.h5")
```

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
```

```
drown_img = image.load_img('/content/train/Drowning/100_png.rf.11dd1e96267c3d3925cc078cf41')
x = image.img_to_array(drown_img)
x = np.expand_dims(x,axis=0)
predicted_class=model.predict(x)
```

```
1/1 [=====] - 0s 107ms/step
```

```
drown_img
```



```
notdrown_img = image.load_img('/content/train/Not Drowning/100_png.rf.7278a89f7b93062d2daa')
x = image.img_to_array(notdrown_img)
x = np.expand_dims(x,axis=0)
predicted_class=model.predict(x)
```

```
1/1 [=====] - 0s 26ms/step
```

```
notdrown_img
```



7.2 Feature 2:

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.mi
n.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://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper
r.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-
JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmY1"
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&d
isplay=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</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>
```

```
</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 >

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.

</p>

</div>

</section>

<section id="footer">

```

    <!-- <div class="social">
      <a href="#" target="_blank"><i class="fab fa-2x fa-twitter-
square"></i></a>
      <a href="#" target="_blank">
        <i class="fab fa-2x fa-linkedin"></i></a>
        <a href="#">
          <i class="#"></i>
        </a>
      </div>-->
</section>
</body>
</html>

```

Login.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: #0889f3;
  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>

```



```

    </div>
  </form>

```

```

</div>

```

```

</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: #0990f7;
  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">
      <input type="text" placeholder="Enter Name" name="name" required><br>
      <input type="email" placeholder="Enter Email ID" name="_id" required><br>
      <input type="password" placeholder="Enter Password" name="psw" required>

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

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

```

```
</div>
</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.mi
n.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://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper
r.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-
JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmY1"
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&d
isplay=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">

</form>

</div>

<h5 style="text-color:yellow">

<b style="text-color:yellow">

</h5>

</div>

</div>

</section>

</br></br>

<section id="footer">

</section>

</body>

</html>

Python :

App.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

```

```
#
```

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 = popul
    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
ate_class_labels()

```

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)

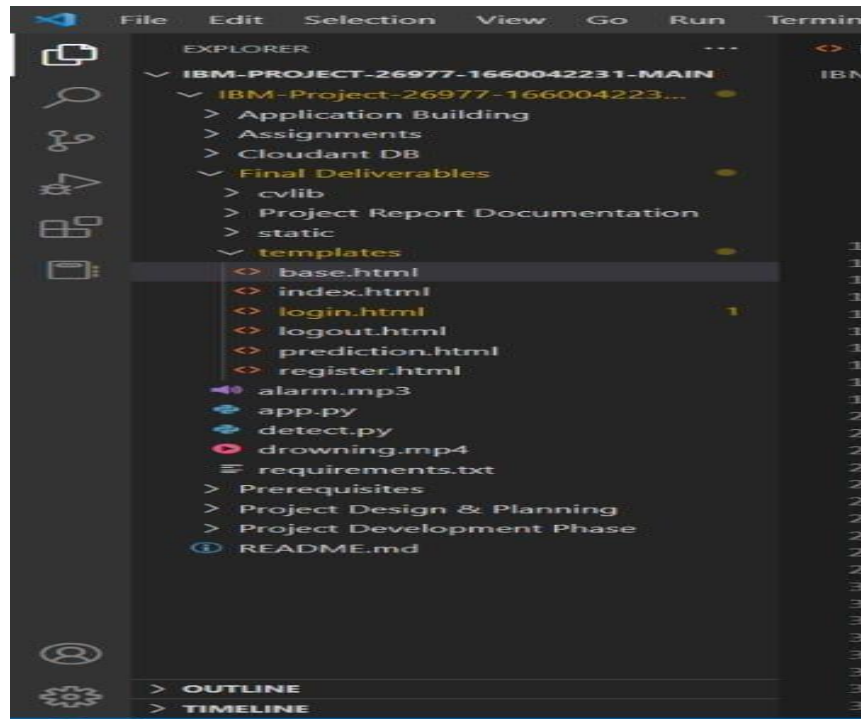
    # display output
    cv2.imshow("Real-time object detection", out)
    if(isDrowning == True):
        playsound(r'C:\Users\HP\Downloads\alarm.mp3')

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

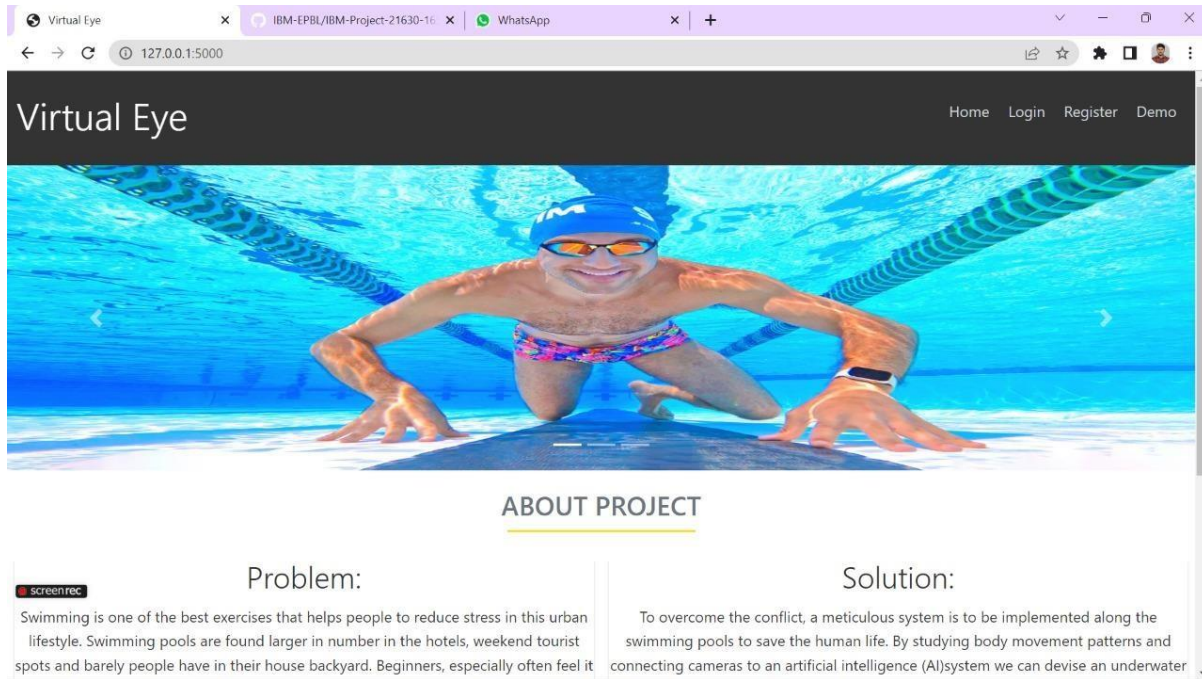
# release resources
webcam.release()
cv2.destroyAllWindows()
```

8. TESTING

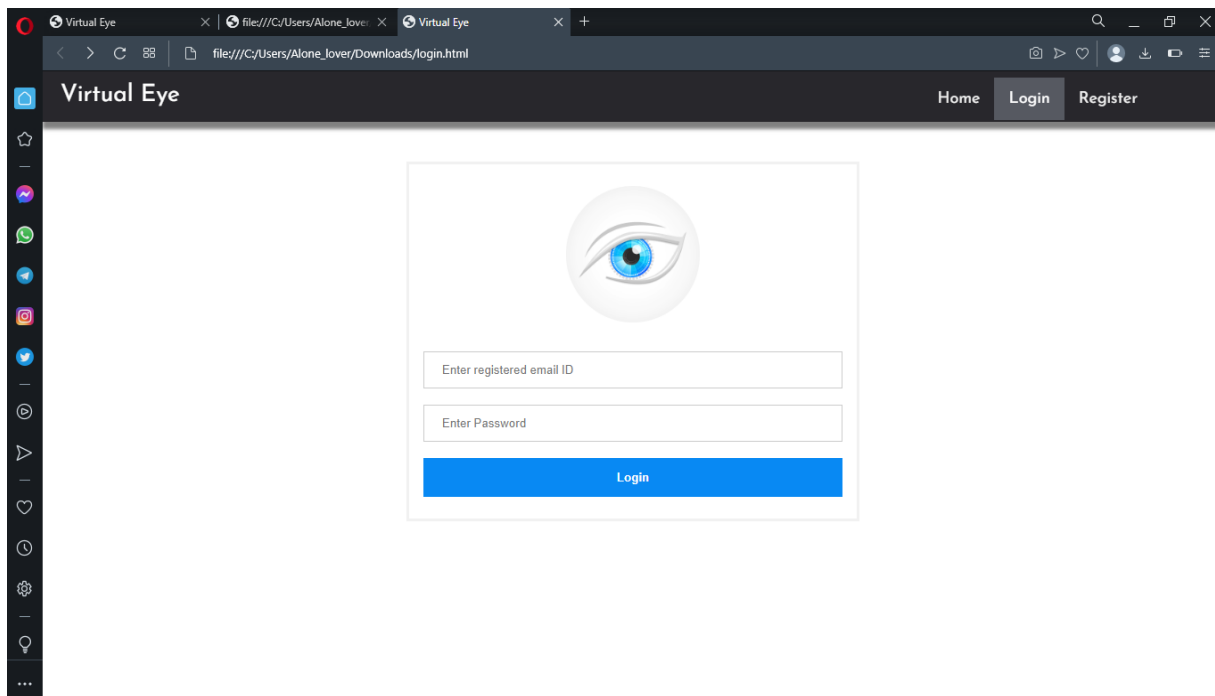
8.1. Test Cases:



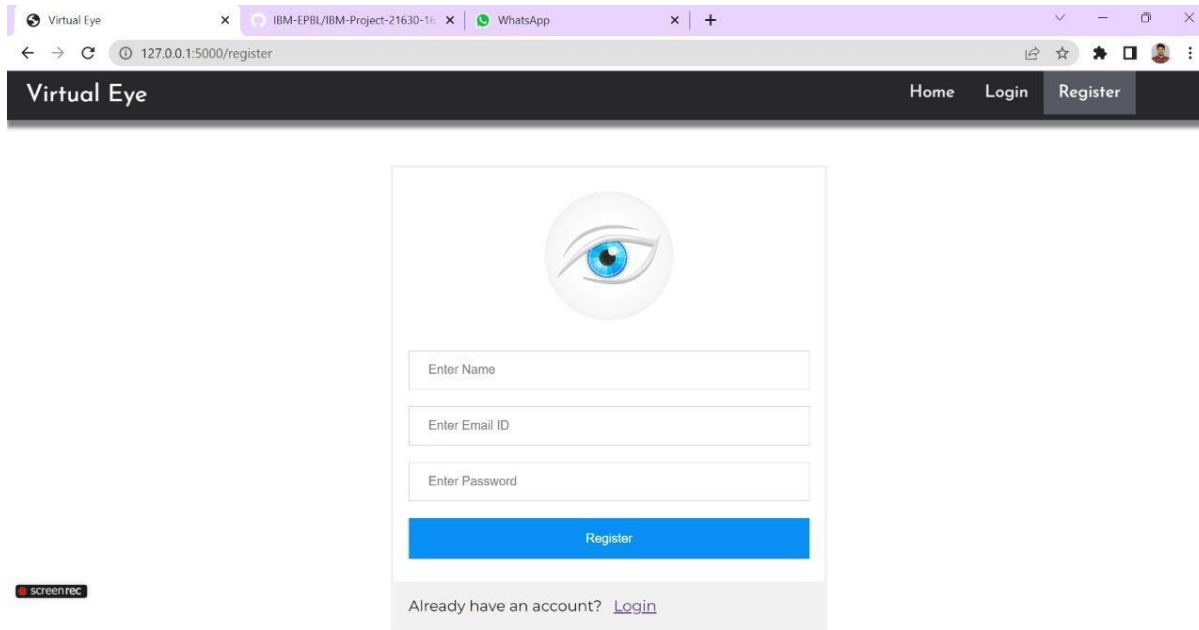
HOME PAGE:



LOGIN PAGE:



REGISTER PAGE:



Virtual Eye

Home Login Register

Enter Name

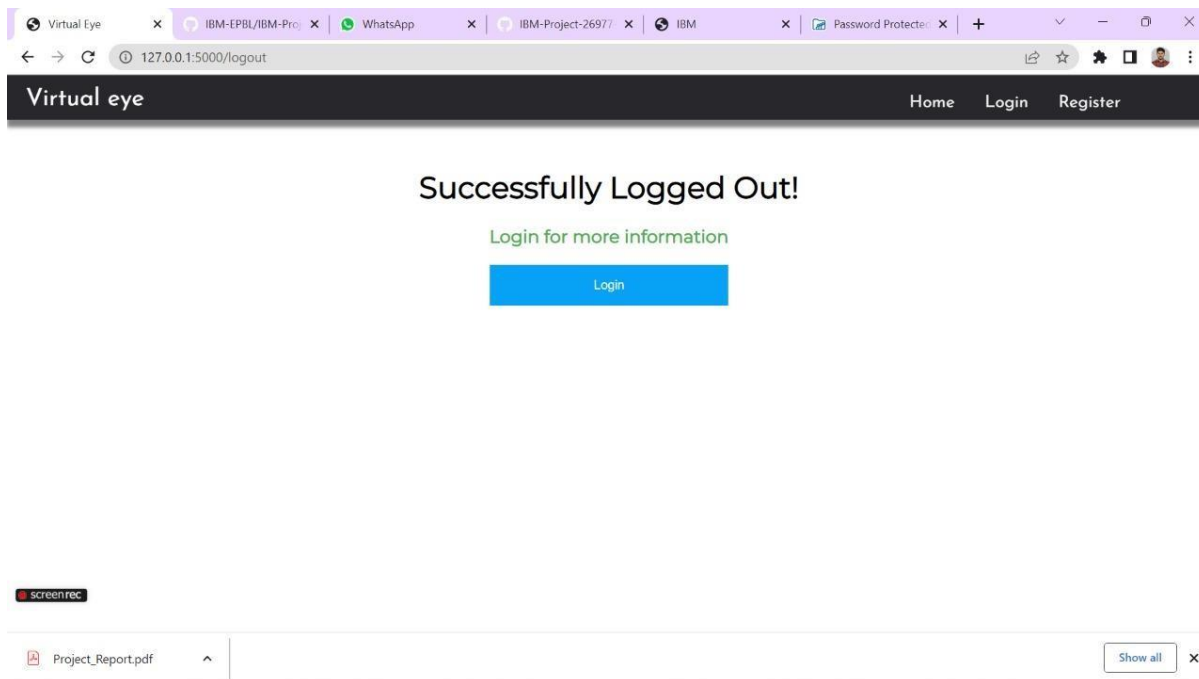
Enter Email ID

Enter Password

Register

Already have an account? [Login](#)

LOG OUT:



Virtual eye

Home Login Register

Successfully Logged Out!

[Login for more information](#)

Login

Project_Report.pdf

Show all

10. 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. After the implementation of all these essentials, this system also can be used on sea beaches for drowning detection

11. Conclusion:

Once we have the working drowning detection model we can feed live video footage of the swimming pool to it so that it can keep detecting continuously for any drowning activities. If drowning is detected it will be highlighted on the system screen as well as alarms will be raised to alert security guards so that they can initiate rescue

12. Appendix

Source Code:

<https://colab.research.google.com/drive/1kSw48COblU6C4sfftvKdvQRhhBqvUW3?usp=sharing>

<https://colab.research.google.com/drive/1NlGhcmvI2EXlILrh55nB40wB5AcDYy1?usp=sharing>

Github

<https://github.com/IBM-EPBL/IBM-Project-21630-1659786288>

Videolink

https://drive.google.com/file/d/1qpLL2z9tHTlXemO2D_3pvEzBfcS4O2cc/view?usp=sharing

