**1.INTRODUCTION:**

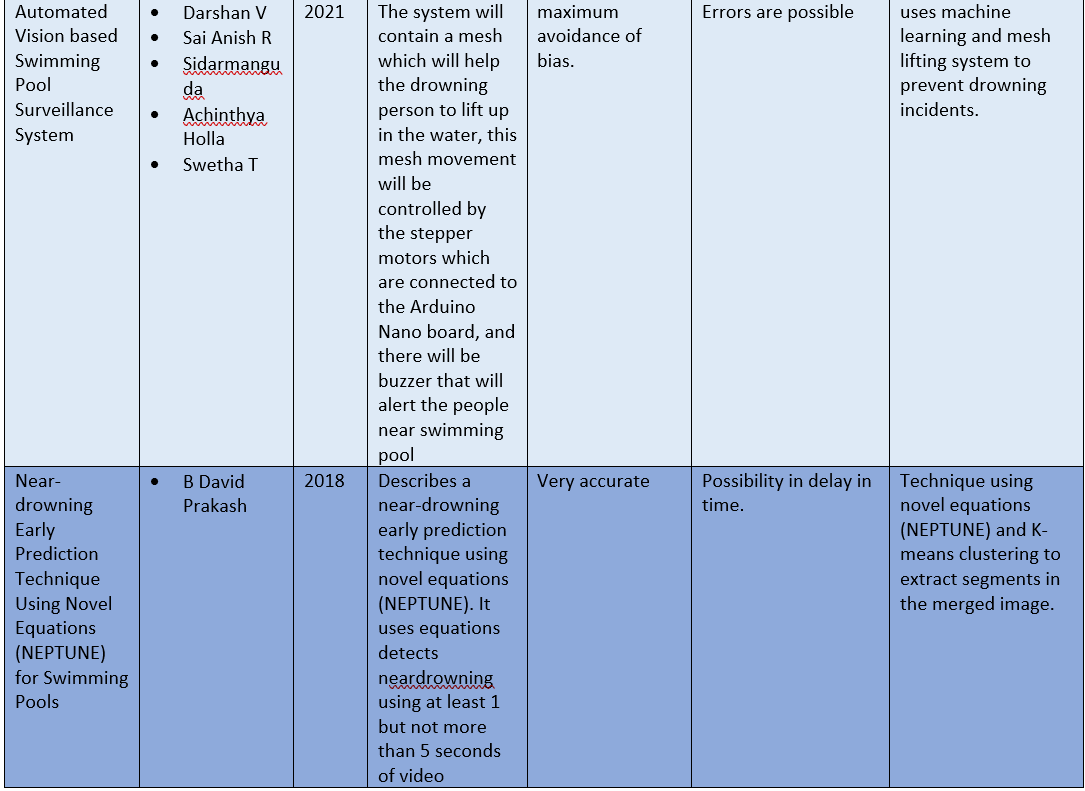
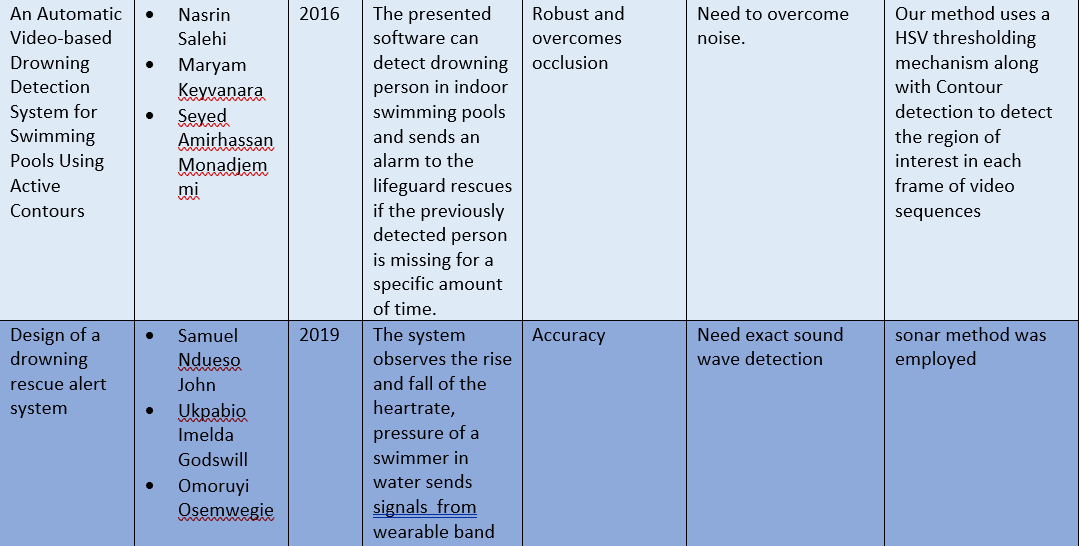
**1.1 PROJECT OVERVIEW:**

This research suggests one effective behaviour identification method using underwater video sequences to swiftly assist lifeguards in determining whether people are drowning in the swimming pool. The methodologies for data labelling and swimmer detection are first established by examining the spatial distribution of swimming pools during regular swimming. Second, this study proposes a framework for the behaviour recognition of swimmers based on the YOLOv4 algorithm (BR-YOLOv4). To further ascertain the swimmer's swimming or drowning behaviour, the spatial relationship between the location information of the target and the swimming/drowning area of the pool is studied. This study evaluates the detection accuracy of various detection algorithms and examines how changing pool angles and swimmer density affect detection. The average accuracy rate for drowning is 94.62%, the average false alarm rate is 1.43%, and the average missing target rate is 3.57%, according to test data. Swimming's average accuracy rate is 97.86%, while its average false rate is 7.93%, average missing rate is 5.93%, and average frame rate is 33f/s. All of the findings indicate that the method put out in this study satisfies the specifications for real-time detection, performs well in identifying swimmer behaviour, and offers technological assistance for lowering drowning incidents in public swimming pools.

**1.2 PURPOSE**

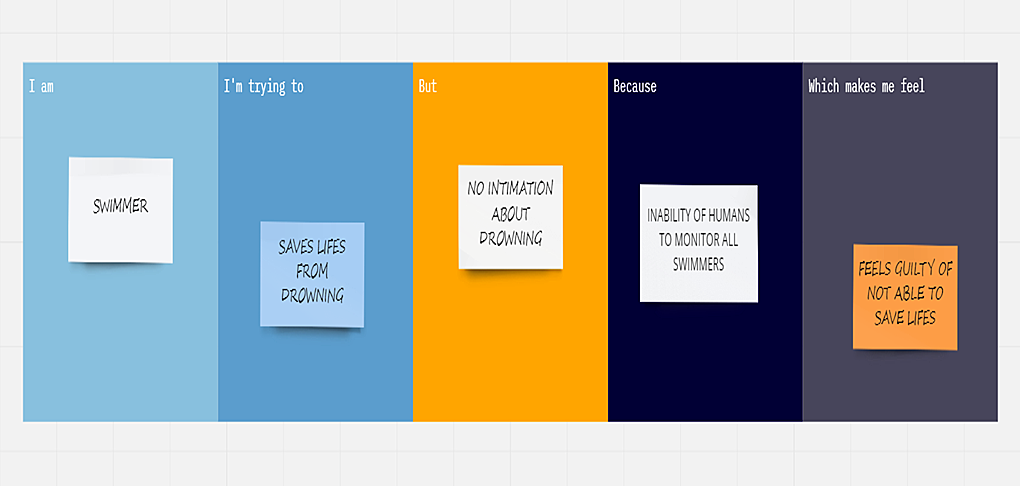
**2.LITERATURE SURVEY**

2.1 EXISTING PROBLEM



**2.3 PROBLEM STATEMENT**

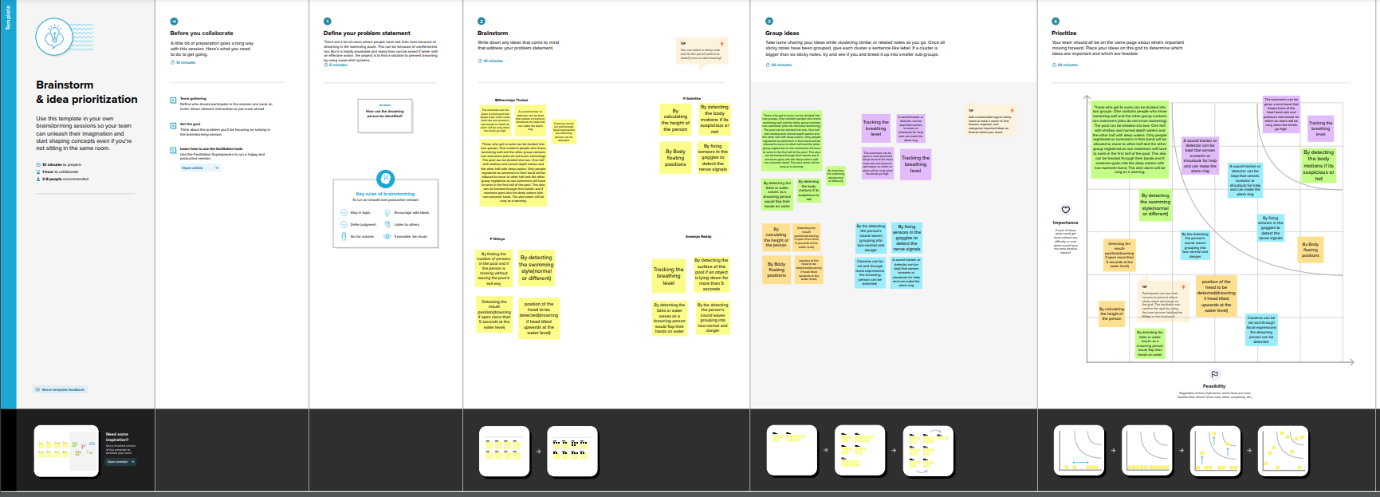
Several death happens in several nations due to drowning swimming pools especially with kids. Due to lack of monitoring and not intimated by drowning many lives are lost.so we have come with solution which detects drowning and rings an alarm and the trainer gets intimated and saves the person who is drowning.



**3 IDEATION AND PROPOSED SOLUTION:**

**3.1 EMPATHY MAP CANVAS:**

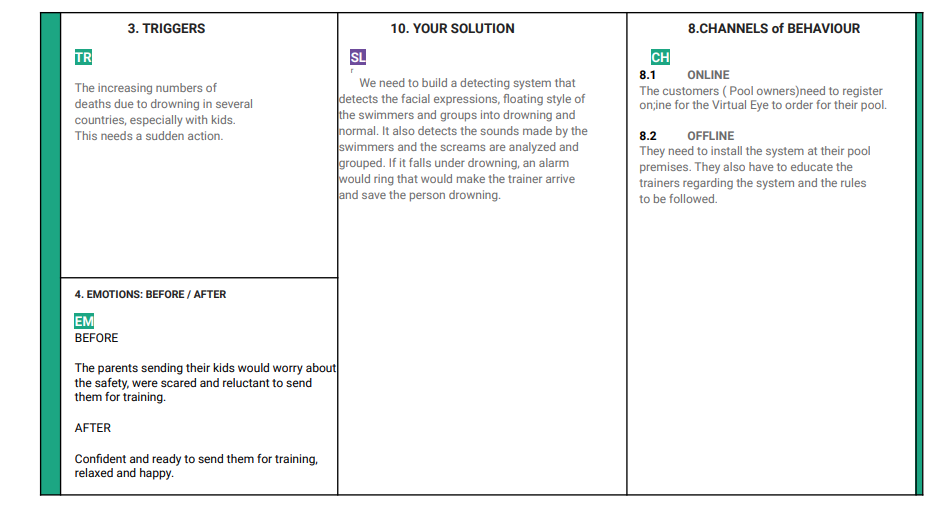
**3.2 IDEATION AND BRAINSTROMING**



**3.3 PROPOSED SOLUTION**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Parameter** | **Description** |
| 1. | Problem Statement (Problem to be solved) | In day to day life we hear about several water accidents.Especially children drowning in swimming pools. To reduce this kind of accidents, a virtual eye Lifeguard is designed. |
| 2. | Idea / Solution description | This Lifeguard is designed to send signals when a swimmer is drowned or facing any critical situation and about to drown in water. The network of video cameras mounted  above and below the water covers the  entire pool surface to identify and pinpoint  any dangers. |
| 3. | Novelty / Uniqueness | Artificial intelligence is a vital aid that    alerts lifeguards to the telltale signs of a    swimmer in distress, helping to prevent    drownings. |
| 4. | Social Impact / Customer Satisfaction | Virtual eye lifeguard triggers an immediatealarm if a swimmer is in peril, helping toavoid panic even in critical situations.  Because, keeping everyone safe is more than a job; it’s a mission. |
| 5. | Business Model (Revenue Model) | The combination of cameras and panted  LED system, the water just got a whole lot  safer. |
| 6. | Scalability of the Solution | Designed for those who had to guarantee everyday the safety in public and intensive-use pools, Virtual eye lifeguard detects potential drownings and promptly notifies you. It features the latest artificial intelligence technology and adapts to theneeds of the user. It’s the ultimate  drowning detection system for those who demand the ultimate in safety. |

**3.4 PROBLEM SOLUTION FIT**



**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 |

**4.2 NON - FUNCTIONAL REQUIREMENTS**

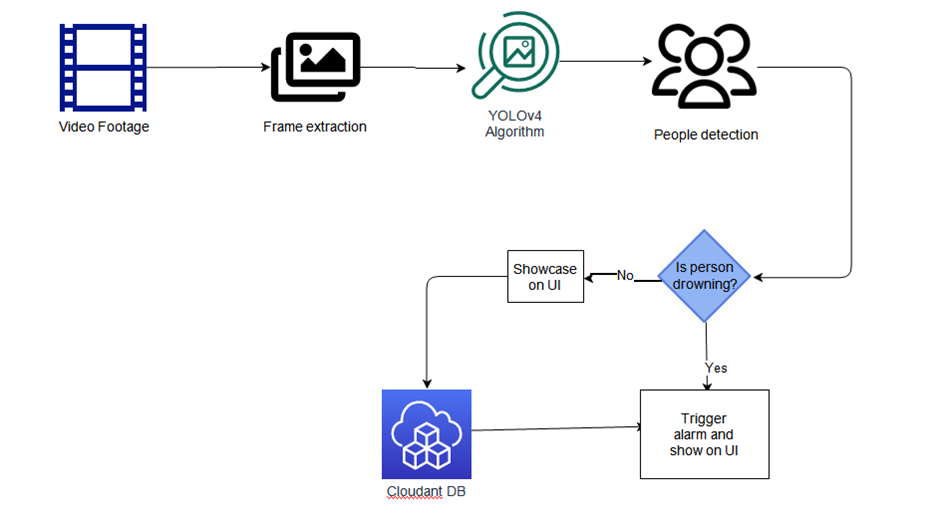
Following are the non-functional requirements of the proposed solution.

|  |  |  |
| --- | --- | --- |
| 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 can detect motion at the pool gate or in the water itself. Some water breach sensors float on the water unobtrusively, whereas others mount to the side of the pool. |
| 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 there is a change in water pressure and when people are about to drown. |
| 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. |

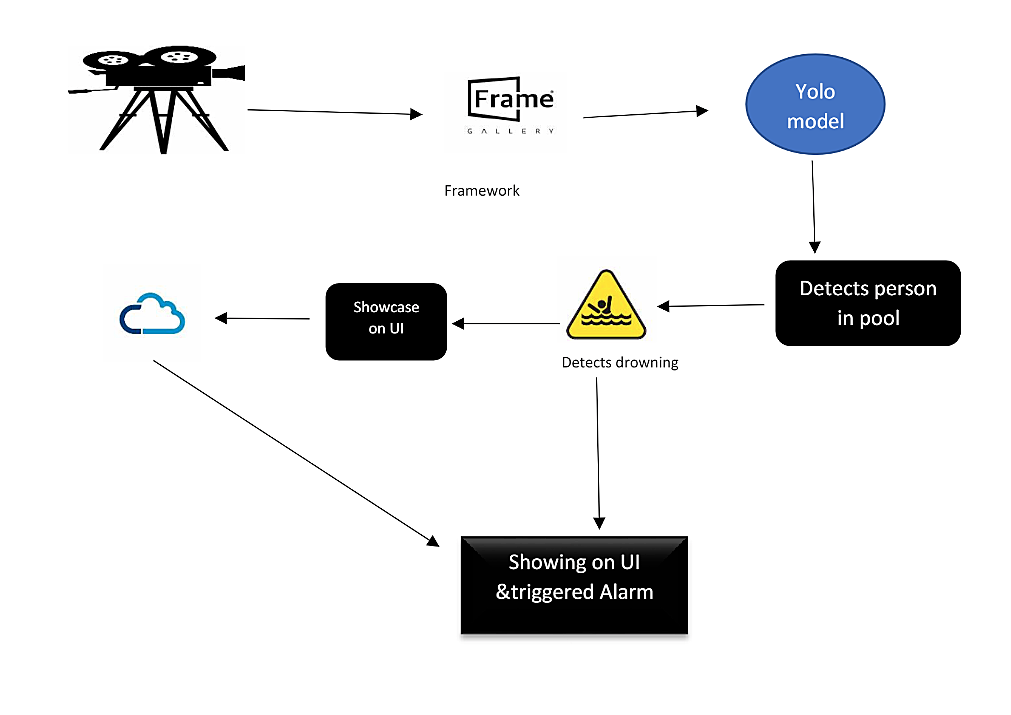
**5 PROJECT DESIGN**

**5.1 DATA FLOW DIAGRAM:**

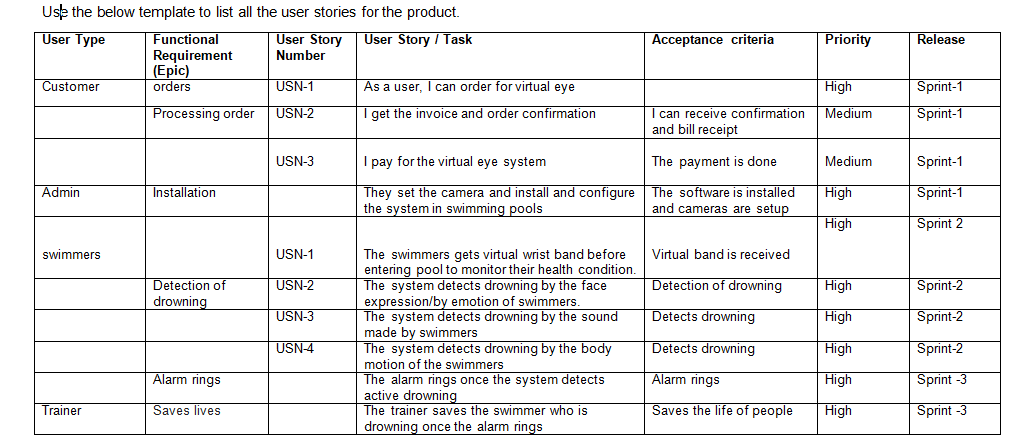
**5.2 SOLUTION ARCHITECTURE**



**5.2 TECHNICAL ARCHITECTURE**



**5.3 USER STORIES:**



**6  PROJECT PLANNING AND SCHEDULING:**

**6.1 SPRINT PLANNING AND ESTIMATION**

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

Use the below template to create product backlog and sprint schedule

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional Requirement (Epic)** | **User Story Number** | **User Story/ Task** | **Story Points** | **Priority** | **Team Members** |
| Sprint-1 | Registration | USN-1 | The user needs to first register on the official website of Virtual Eye using Email, Password. | 2 | High |  |
| Sprint-1 | Login | USN-2 | On registering, the user can directly login to order the product. | 1 | High |  |
| Sprint-1 | Dashboard | USN-3 | Develop HTMLpage for creating login forms andwebsite. | 2 | Medium |  |
| Sprint-2 | Model code | USN-4 | Develop python code for detection using yolo model | 2 | Medium |  |
| Sprint-3 | Development | USN-5 | Connect python code with HTMLpages using YOLO. | 1 | High |  |
| Sprint-4 | Designing the alarm | USN-6 | Develop a system that alarms if drowning is detected. | 1 | High |  |

**6.2 SPRINT DELIVERY SCHEDULE**

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Total Story Points** | **Duration** | **Sprint StartDate** | **Sprint  End Date      (Planned)** | **Story Points**  **Completed (as on PlannedEnd Date)** | **Sprint Release Date(Actual)** |
| Sprint-1 | 20 | 6 Days | 19 Oct 2022 | 24 Oct 2022 |  |  |
| Sprint-2 | 20 | 6 Days | 26 Oct 2022 | 31 Nov 2022 |  |  |
| Sprint-3 | 20 | 6 Days | 02 Nov 2022 | 08 Nov 2022 |  |  |
| Sprint-4 | 20 | 6 Days | 10 Nov 2022 | 15 Nov 2022 |  |  |

**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 = Sprint Duration / velocity = 8 / 6 = 1.3V

For Sprint-2 the AverageVelocity (AV) is: AV = Sprint Duration / velocity = 14 / 6 = 2.3V

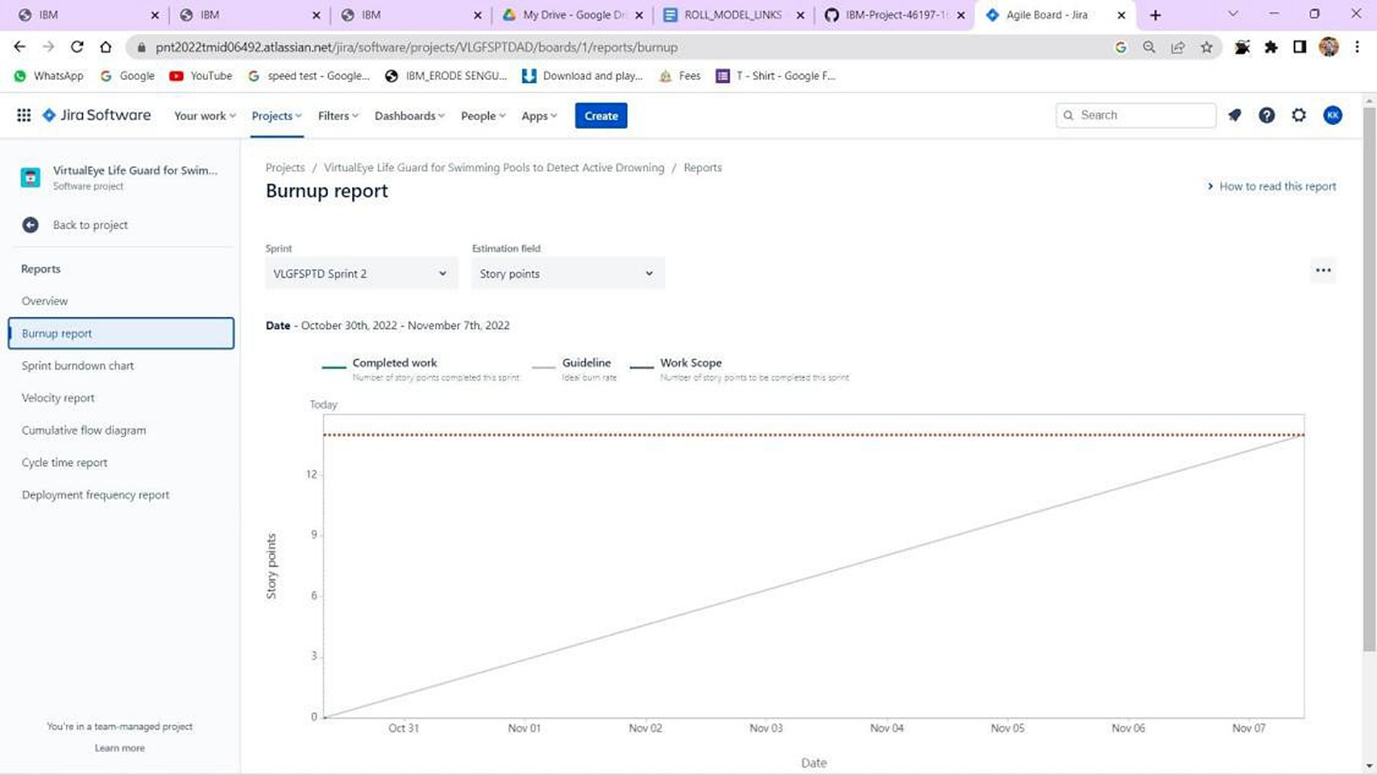
For Sprint-3 the Average Velocity(AV) is: AV = Sprint Duration / velocity = 16 / 6 = 2.6V

For Sprint-4 the Average Velocity (AV) is: AV = Sprint Duration / velocity = 12/ 6 = 2.0V TOTAL TEAM AVERAGE VELOCITY = 2.08

**6.3 REPORTS FROM JIRA:**

**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](https://www.visual-paradigm.com/scrum/what-is-agile-software-development/) methodologies such as [Scrum](https://www.visual-paradigm.com/scrum/scrum-in-3-minutes/). However, burn down charts can be applied to any project containing measurable progress over time.



**7 CODING AND SOLUTIONING:**

7.1 FEATURE 1

Drowning\_V2\_testing\_Serial\_17\_5\_22.py

import tensorflow as tf

from keras.preprocessing.image import ImageDataGenerator

from keras.preprocessing import image

import numpy as np

import easygui

from keras.models import load\_model

import os

import serial

import time

print(tf.\_\_version\_\_)

model1 = load\_model('model/Class1/model\_Class1.h5')

model2 = load\_model('model/Class2/model\_Class2.h5')

# Testing

image11 = easygui.fileopenbox()

test\_image2 = image.load\_img(image11, target\_size = (64, 64))

test\_image2 = image.img\_to\_array(test\_image2)

test\_image2 = np.expand\_dims(test\_image2, axis = 0)

# cnn prediction on the test image

result2 = model1.predict(test\_image2)

print(result2)

if result2[0][0] == 1:

   result3 = model2.predict(test\_image2)

   if result3[0][0] == 1:

      prediction2 = 'Human Drowning Detected'

      SerialObj = serial.Serial('COM7')

      SerialObj.baudrate = 9600

      SerialObj.bytesize = 8

      SerialObj.parity   ='N'

      SerialObj.stopbits = 1

      time.sleep(3)

      SerialObj.write(b'a')

      SerialObj.close()

      #ser = serial.Serial("COM7", 9600)

      #data = "X61"  # a  -> Human

      #ser.write(data)

      #s = ser.read(9)

   else:

      prediction2 = 'AnimalDrowning Detected'

      SerialObj = serial.Serial('COM7')

      SerialObj.baudrate = 9600

      SerialObj.bytesize = 8

      SerialObj.parity   ='N'

      SerialObj.stopbits = 1

      time.sleep(3)

      SerialObj.write(b'a')

      SerialObj.close()

##      ser = serial.Serial("COM7", 9600)

##      data = "X62"  # b -> Animal

##      ser.write(data)

##      s = ser.read(9)

else:

   prediction2 = 'Normal '

   SerialObj = serial.Serial('COM6')

   SerialObj.baudrate = 9600

   SerialObj.bytesize = 8

   SerialObj.parity   ='N'

   SerialObj.stopbits = 1

   time.sleep(3)

   SerialObj.write(b'a')

   SerialObj.close()

##   ser = serial.Serial("COM7", 9600)

##   data = "X63"  # c -> Empty

##   ser.write(data)

##   s = ser.read(9)

print(prediction2)

**7.2  FEATURE 2:**

Drowning\_V2\_Class1\_Training\_17\_5\_22.py

import tensorflow as tf

from keras.preprocessing.image import ImageDataGenerator

from keras.preprocessing import image

import numpy as np

import easygui

import os

import serial

print(tf.\_\_version\_\_)

#feature training

train\_datagen = ImageDataGenerator(

        # reducing/normalizing the pixels

        rescale=1./255,

        shear\_range=0.2,

        zoom\_range=0.2,

        horizontal\_flip=True)

#connecting the image augmentation tool to our dataset

train\_set = train\_datagen.flow\_from\_directory(

        'Class1/training\_set',

        #final size of the images that will be fed into the ann

        target\_size=(64, 64),

        # number of images that we want to have in each batch

        batch\_size=32,

        # we have binary classification --> binary class mode

        class\_mode='binary')

#only rescaling but no transformations

test\_datagen = ImageDataGenerator(rescale=1./255)

#connecting to the test data

test\_set = test\_datagen.flow\_from\_directory(

        'Class1/test\_set',

        target\_size=(64, 64),

        batch\_size=32,

        class\_mode='binary')

print(test\_set)

#--------------------- Building CNN --------------------#

# initializing CNN as sequential layers

cnn = tf.keras.models.Sequential()

# Step 1: Convolution to get the Feature Map

cnn.add(tf.keras.layers.Conv2D(filters = 32, kernel\_size = 3, activation = 'relu', input\_shape=[64,64,3]))

# Step 2: Max Pooling

cnn.add(tf.keras.layers.MaxPool2D(pool\_size=2 ,strides=2))

#adding a second convolutional layer

cnn.add(tf.keras.layers.Conv2D(filters = 32, kernel\_size = 3, activation = 'relu'))

cnn.add(tf.keras.layers.MaxPool2D(pool\_size=2 ,strides=2))

# Step 3: Flattening

cnn.add(tf.keras.layers.Flatten())

# Step 4: Full Connection

cnn.add(tf.keras.layers.Dense(units = 128, activation = 'relu'))

# Step 5: Output Layer

cnn.add(tf.keras.layers.Dense(units = 1, activation = 'sigmoid'))

#--------------------- Training the CNN --------------------#

#compiling the CNN

cnn.compile(optimizer = 'adam', loss = 'binary\_crossentropy', metrics = ['accuracy'])

#training the CNN on the training set and evaluating it on the test set

cnn.fit(x = train\_set, validation\_data = test\_set, epochs = 25)

cnn.save('model/save1',overwrite=True,

    include\_optimizer=True,

    save\_format=None,

    signatures=None,

    options=None,

    save\_traces=True,)

cnn.save('model/Class1/model\_Class1.h5')

# Testing

image11 = easygui.fileopenbox()

test\_image2 = image.load\_img(image11, target\_size = (64, 64))

test\_image2 = image.img\_to\_array(test\_image2)

test\_image2 = np.expand\_dims(test\_image2, axis = 0)

# cnn prediction on the test image

result2 = cnn.predict(test\_image2)

print(result2)

if result2[0][0] == 1:

   prediction2 = 'Present'

else:

   prediction2 = 'Empty'

print(prediction2)

**8 TESTING:**

**8.1 TEST CASES**

**Acceptance Testing**

**UAT Execution & Report Submission**

|  |  |
| --- | --- |
| Date | 03 November 2022 |
| Team ID | PNT2022TMID000595 |
| Project Name | Project – Virtual Eye for drowning detection |
| Maximum Marks | 4 Marks |

1. **Purpose of Document**

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

1. **Defect Analysis**

This report showsthe 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 | 9 | 3 | 2 | 2 | 16 |
| Duplicate | 1 | 1 | 2 | 0 | 4 |
| External | 3 | 3 | 0 | 2 | 8 |
| Fixed | 12 | 2 | 3 | 15 | 32 |
| Not Reproduced | 0 | 0 | 1 | 0 | 1 |
| Skipped | 1 | 0 | 1 | 1 | 3 |
| Won't Fix | 0 | 3 | 2 | 1 | 6 |
| Totals | 26 | 12 | 11 | 21 | 45 |

1. **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 | 6 | 0 | 1 | 5 |
| Client Application | 45 | 0 | 0 | 45 |
| Security | 3 | 0 | 0 | 3 |
| Outsource Shipping | 5 | 1 | 0 | 4 |
| Exception Reporting | 9 | 0 | 0 | 9 |
| Final Report Output | 4 | 0 | 0 | 4 |
| Version Control | 2 | 0 | 0 | 2 |

**Project Development Phase**

**Model Performance Test**

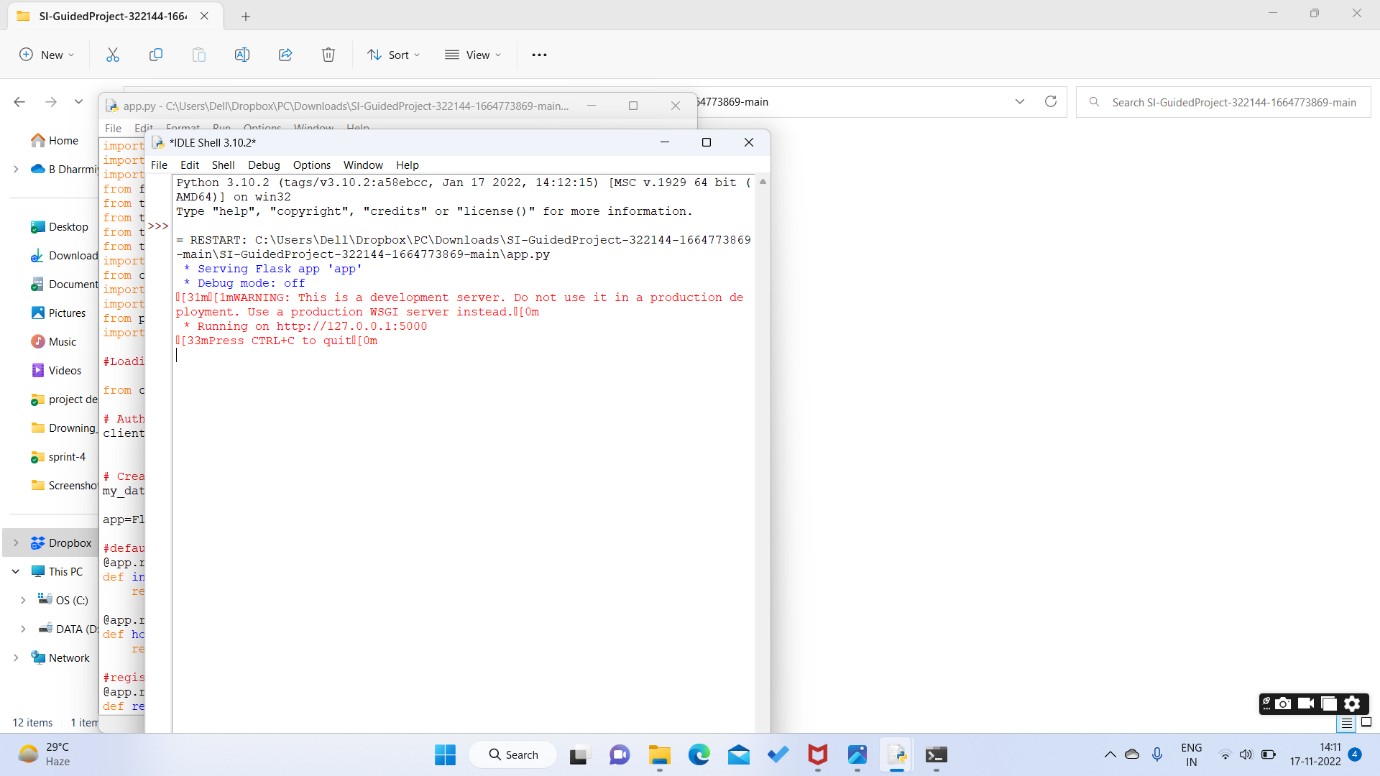
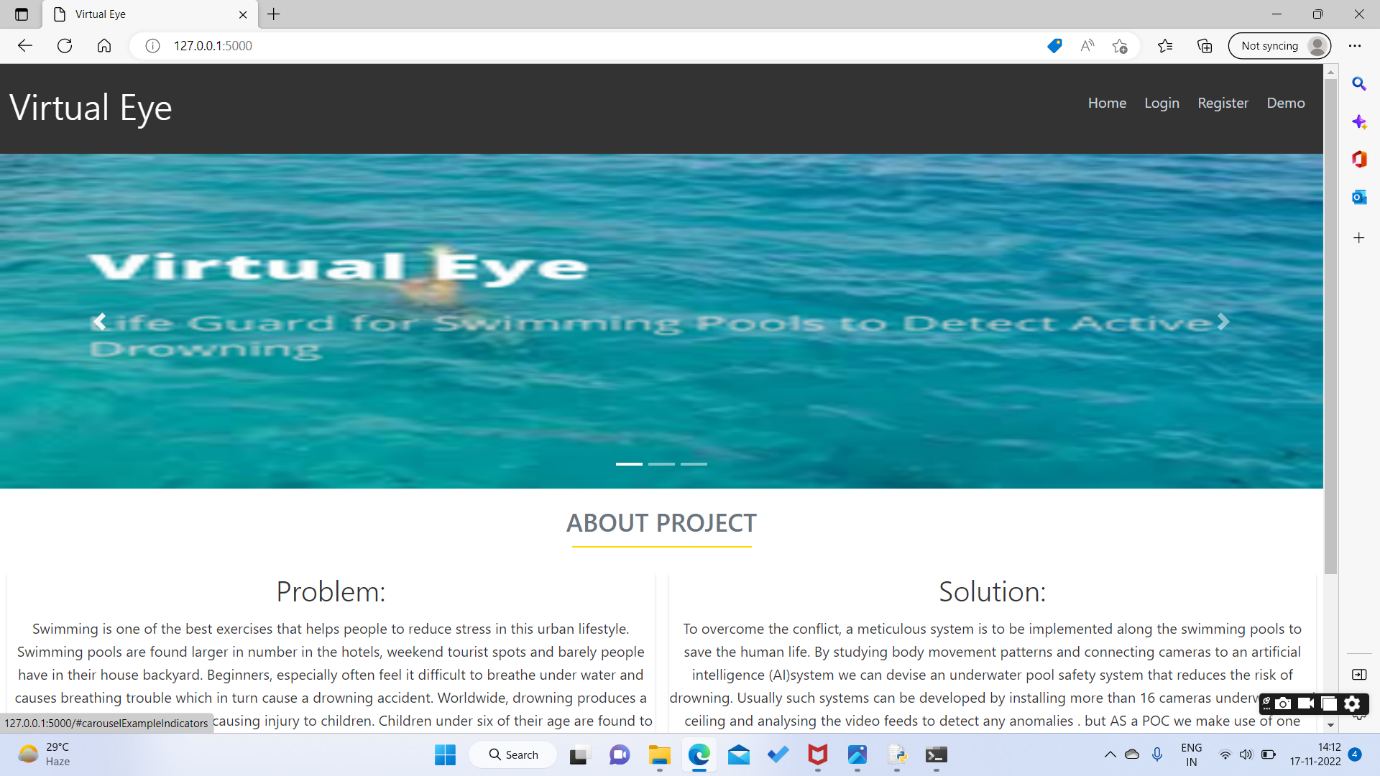
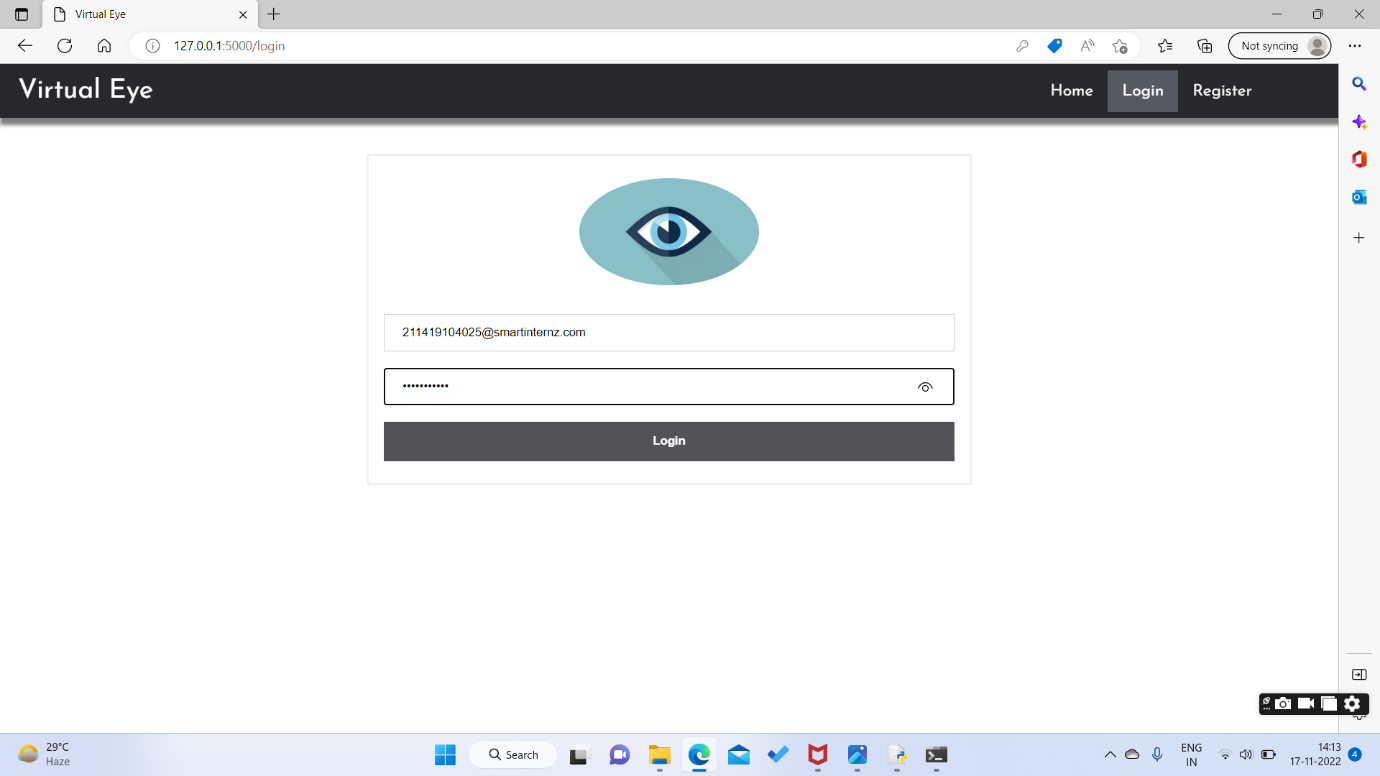
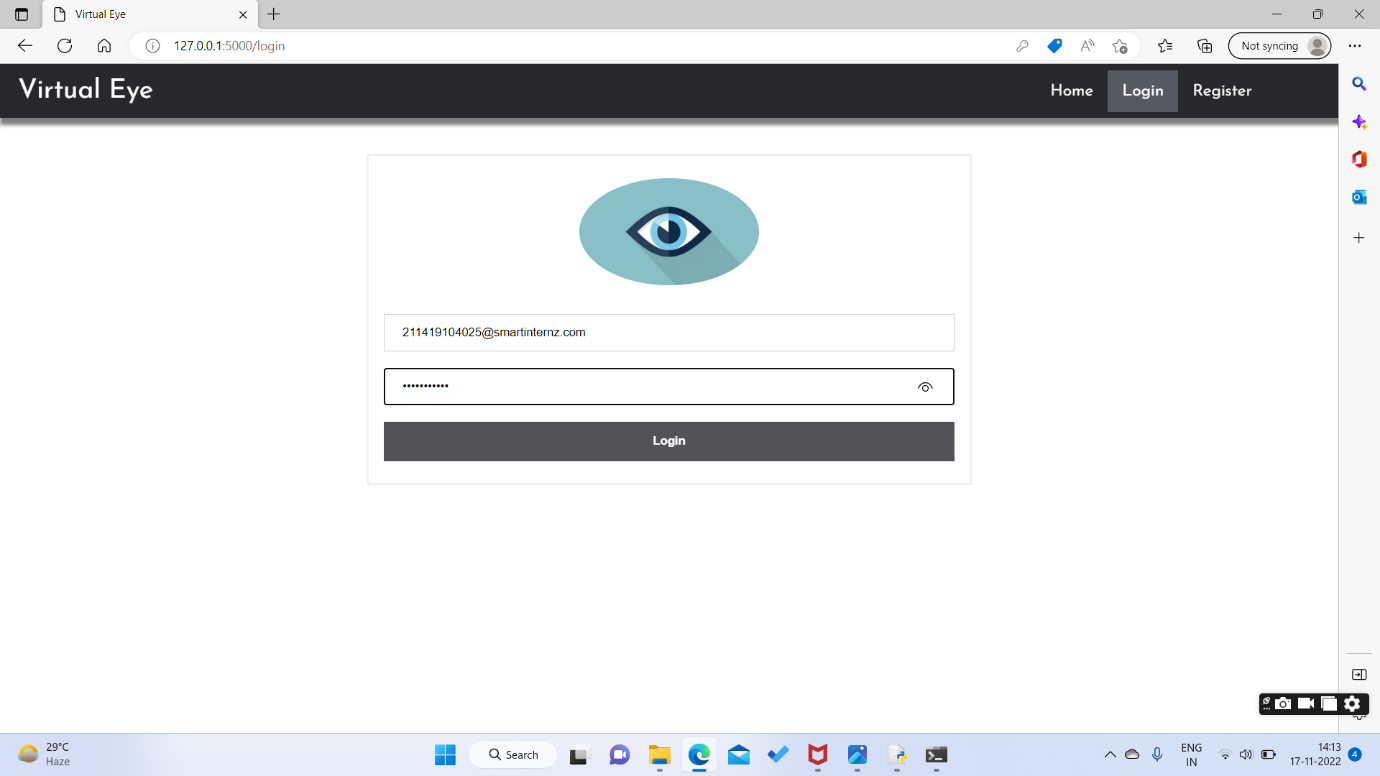
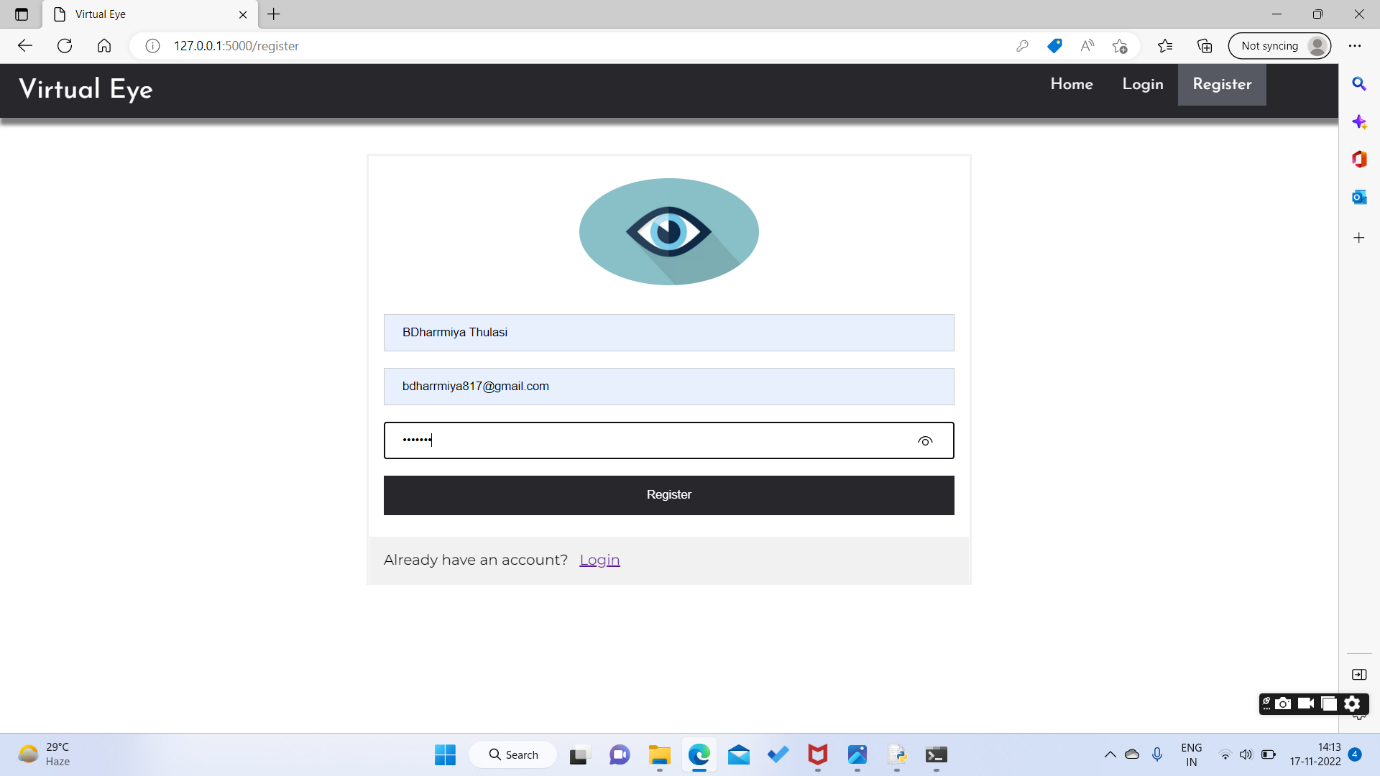
|  |  |
| --- | --- |
| Date | 10 November 2022 |
| Team ID | PNT2022TMID000595 |
| Project Name | Project – Virtual Eye for drowning detection |
| Maximum Marks | 10 Marks |

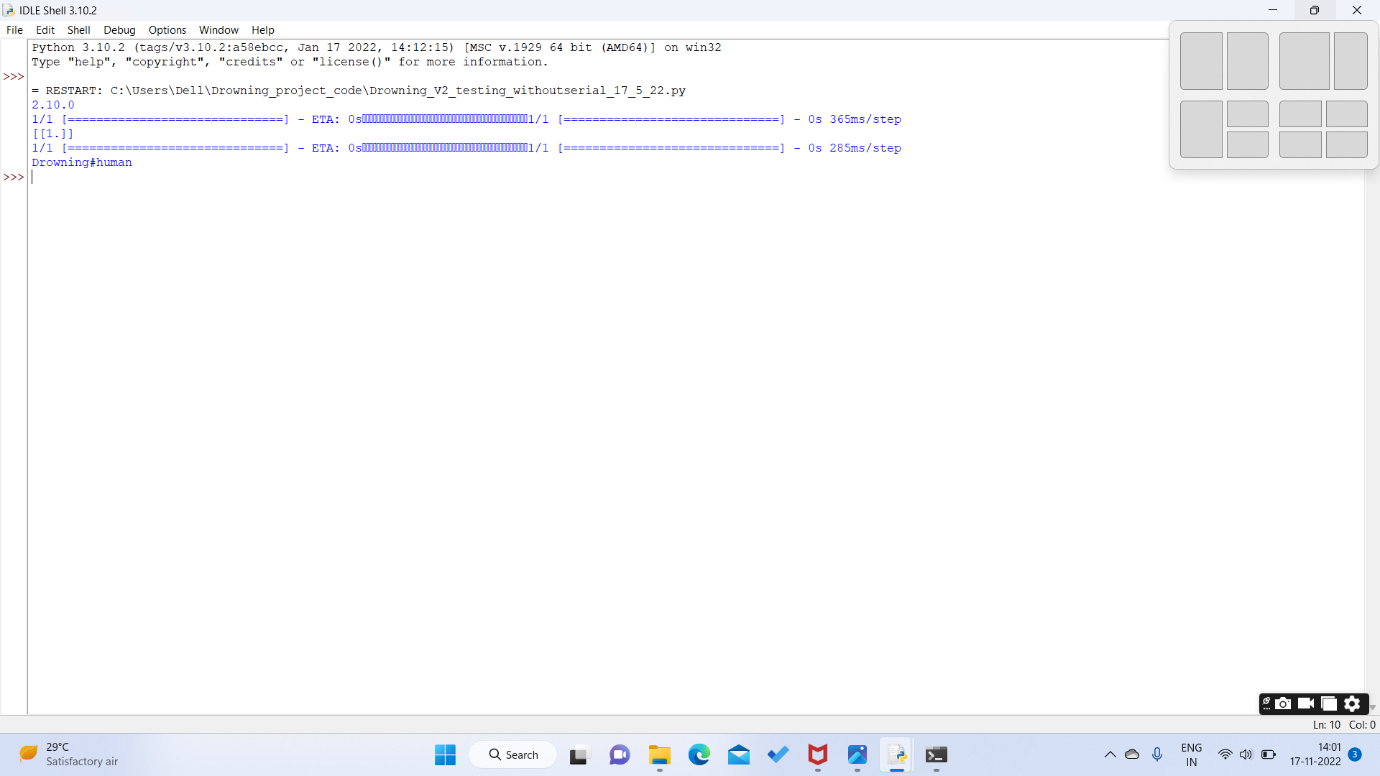
**Model Performance Testing:**

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

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Parameter** | **Values** | **Screenshot** |
|  | Model Summary | This model is trained and tested to detect drowning in the swimming pool. |  |
|  | Accuracy | Training Accuracy - 9  Validation Accuracy -11 |  |
| 3. | Confidence Score (Only Yolo Projects) | Class Detected -4  Confidence Score - 9 |  |

9. RESULT





**10.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.

11.APPENDICS

CODE:

Drowning\_V2\_testing\_Serial\_17\_5\_22.py

import tensorflow as tf

from keras.preprocessing.image import ImageDataGenerator

from keras.preprocessing import image

import numpy as np

import easygui

from keras.models import load\_model

import os

import serial

import time

print(tf.\_\_version\_\_)

model1 = load\_model('model/Class1/model\_Class1.h5')

model2 = load\_model('model/Class2/model\_Class2.h5')

# Testing

image11 = easygui.fileopenbox()

test\_image2 = image.load\_img(image11, target\_size = (64, 64))

test\_image2 = image.img\_to\_array(test\_image2)

test\_image2 = np.expand\_dims(test\_image2, axis = 0)

# cnn prediction on the test image

result2 = model1.predict(test\_image2)

print(result2)

if result2[0][0] == 1:

   result3 = model2.predict(test\_image2)

   if result3[0][0] == 1:

      prediction2 = 'Human Drowning Detected'

      SerialObj = serial.Serial('COM7')

      SerialObj.baudrate = 9600

      SerialObj.bytesize = 8

      SerialObj.parity   ='N'

      SerialObj.stopbits = 1

      time.sleep(3)

      SerialObj.write(b'a')

      SerialObj.close()

      #ser = serial.Serial("COM7", 9600)

      #data = "X61"  # a  -> Human

      #ser.write(data)

      #s = ser.read(9)

   else:

      prediction2 = 'AnimalDrowning Detected'

      SerialObj = serial.Serial('COM7')

      SerialObj.baudrate = 9600

      SerialObj.bytesize = 8

      SerialObj.parity   ='N'

      SerialObj.stopbits = 1

      time.sleep(3)

      SerialObj.write(b'a')

      SerialObj.close()

##      ser = serial.Serial("COM7", 9600)

##      data = "X62"  # b -> Animal

##      ser.write(data)

##      s = ser.read(9)

else:

   prediction2 = 'Normal '

   SerialObj = serial.Serial('COM6')

   SerialObj.baudrate = 9600

   SerialObj.bytesize = 8

   SerialObj.parity   ='N'

   SerialObj.stopbits = 1

   time.sleep(3)

   SerialObj.write(b'a')

   SerialObj.close()

##   ser = serial.Serial("COM7", 9600)

##   data = "X63"  # c -> Empty

##   ser.write(data)

##   s = ser.read(9)

print(prediction2)

**7.2  FEATURE 2:**

Drowning\_V2\_Class1\_Training\_17\_5\_22.py

import tensorflow as tf

from keras.preprocessing.image import ImageDataGenerator

from keras.preprocessing import image

import numpy as np

import easygui

import os

import serial

print(tf.\_\_version\_\_)

#feature training

train\_datagen = ImageDataGenerator(

        # reducing/normalizing the pixels

        rescale=1./255,

        shear\_range=0.2,

        zoom\_range=0.2,

        horizontal\_flip=True)

#connecting the image augmentation tool to our dataset

train\_set = train\_datagen.flow\_from\_directory(

        'Class1/training\_set',

        #final size of the images that will be fed into the ann

        target\_size=(64, 64),

        # number of images that we want to have in each batch

        batch\_size=32,

        # we have binary classification --> binary class mode

        class\_mode='binary')

#only rescaling but no transformations

test\_datagen = ImageDataGenerator(rescale=1./255)

#connecting to the test data

test\_set = test\_datagen.flow\_from\_directory(

        'Class1/test\_set',

        target\_size=(64, 64),

        batch\_size=32,

        class\_mode='binary')

print(test\_set)

#--------------------- Building CNN --------------------#

# initializing CNN as sequential layers

cnn = tf.keras.models.Sequential()

# Step 1: Convolution to get the Feature Map

cnn.add(tf.keras.layers.Conv2D(filters = 32, kernel\_size = 3, activation = 'relu', input\_shape=[64,64,3]))

# Step 2: Max Pooling

cnn.add(tf.keras.layers.MaxPool2D(pool\_size=2 ,strides=2))

#adding a second convolutional layer

cnn.add(tf.keras.layers.Conv2D(filters = 32, kernel\_size = 3, activation = 'relu'))

cnn.add(tf.keras.layers.MaxPool2D(pool\_size=2 ,strides=2))

# Step 3: Flattening

cnn.add(tf.keras.layers.Flatten())

# Step 4: Full Connection

cnn.add(tf.keras.layers.Dense(units = 128, activation = 'relu'))

# Step 5: Output Layer

cnn.add(tf.keras.layers.Dense(units = 1, activation = 'sigmoid'))

#--------------------- Training the CNN --------------------#

#compiling the CNN

cnn.compile(optimizer = 'adam', loss = 'binary\_crossentropy', metrics = ['accuracy'])

#training the CNN on the training set and evaluating it on the test set

cnn.fit(x = train\_set, validation\_data = test\_set, epochs = 25)

cnn.save('model/save1',overwrite=True,

    include\_optimizer=True,

    save\_format=None,

    signatures=None,

    options=None,

    save\_traces=True,)

cnn.save('model/Class1/model\_Class1.h5')

# Testing

image11 = easygui.fileopenbox()

test\_image2 = image.load\_img(image11, target\_size = (64, 64))

test\_image2 = image.img\_to\_array(test\_image2)

test\_image2 = np.expand\_dims(test\_image2, axis = 0)

# cnn prediction on the test image

result2 = cnn.predict(test\_image2)

print(result2)

if result2[0][0] == 1:

   prediction2 = 'Present'

else:

   prediction2 = 'Empty'

print(prediction2)

APP.PY

import re

import numpy as np

import os

from flask import Flask, app, request, render\_template, redirect, url\_for

from tensorflow.keras import models

from tensorflow.keras.models import load\_model

from tensorflow.keras.preprocessing import image

from tensorflow.python.ops.gen\_array\_ops import concat

import cvlib as cv

from cvlib.object\_detection import draw\_bbox

import cv2

import time

from playsound import playsound

import requests

#Loading the model

from cloudant.client import Cloudant

# Authenticate using an IAM API key

client = Cloudant.iam('57f444d5-dfbd-4fc0-b752-dea54005c3cc-bluemix','HTLp9\_GkWGDyMR9VHruMMwi\_qzZ43qaI3UVR77GOI2GX', 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()

#print(frame)

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

#print('bbox',bbox)

#print('label',label)

#print('conf',conf)

#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

#print('came here')

out = draw\_bbox(frame, bbox, label, conf,colors=None,write\_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 drowining")

#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',prediction="Emergency !!! The Person is drowining")

""" Running our application """

if \_\_name\_\_ == "\_\_main\_\_":

app.run(debug=False)

11.GITHUB LINK

https://github.com/IBM-EPBL/IBM-Project-5104-1658747774