Project Report

VirtualEye - Life Guard for Swimming Pools to Detect Active Drowning

Team ID : PNT2022TMID50339
Team Leader : MUDUNRI SAI KRISHNAM RAJU (190701117)
Team Member : PRASANNA VENKA A (190701142)
ANAND PRINCE PURTY (190701501)
MOHAN SAI P K (190701115)
College Name : RAJALAKSHMI ENGINEERING COLLEGE
Faculty Mentor : Vijay K
Industrial Mentor : Swathi

S.NO

Table of Content

Page No.

1. INTRODUCTION 3

1.1. Project Overview 3

1.2. Purpose 4

2. LITERATURE SURVEY

2.1. Existing problem 4

2.2. References 4

2.3. Problem Statement Definition 4

3. IDEATION & PROPOSED SOLUTION

3.1. Empathy Map Canvas 5

3.2. Ideation & Brainstorming 5

3.3. Proposed Solution 5

3.4. Problem Solution fit 6

4. REQUIREMENT ANALYSIS

4.1. Functional requirements 6

4.2. Non-Functional requirements 6

5. PROJECT DESIGN

5.1. Data Flow Diagrams 7

5.2. Solution & Technical Architecture 7

5.3. User Stories 8

6. PROJECT PLANNING & SCHEDULING

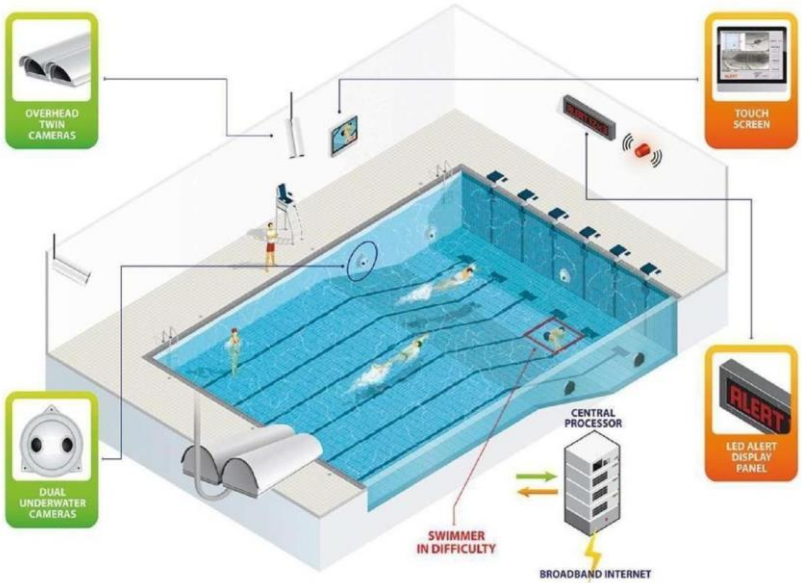
6.1. Sprint Planning & Estimation 8
6.2. Sprint Delivery Schedule 8
6.3. Reports from JIRA 9
7. CODING & SOLUTIONING
(Explain the features added in the project along with code)
7.1.Feature 1 10
7.2. Feature 2 10
8. TESTING 10
8.1. Test Cases 10
8.2. User Acceptance Testing 11
9. ADVANTAGES & DISADVANTAGES 11
10. CONCLUSION 12
11. APPENDIX
Source Code 12
GitHub & Project Demo Link 18

1.INTRODUCTION

Recently, there has been growing interest around the topic of drowning detection systems (DDS) in the sport and leisure industry both across the UK and globally. Advancements in technology, coupled with the importance of pool safety, has led to its growing prominence, with mention of DDS now in documents such as HSG179 - the latest UK standards document for health and safety in swimming pools (Health and Safety Executive, 2018). However, the topic is a debated area for various reasons explored in this review.Whilst there are plenty of academic articles dedicated to the technology and design behind these products in the fields of biometrics, computer science and electronic engineering, there is limited academic research investigating their application to real-world scenarios. Furthermore, there is uncertainty around their use alongside traditional lifeguarding; whether international testing standards (ISO standards) are robust enough; and general risks affecting the effectiveness of these products. This includes factors such as water clarity, high pool occupancy, lighting, glare and attractions such as water slides and wave machines. These concerns alongside the lack of research and high installation costs have resulted in a reluctance by some operators to incorporate DDS into their pools. This signifies the importance of independent research into DDS. intends to support the move towards the shared goal of improved pool safety.

1.1. Project Overview

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 underwaterwhich causes breathing trouble which in turn causes a drowning accident. 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.



1.2. Purpose

It helps the lifeguard to detect the underwater situation where they can't easily observe.

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

2.LITERATURE SURVEY

2.1. Existing problem

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

2.2. References

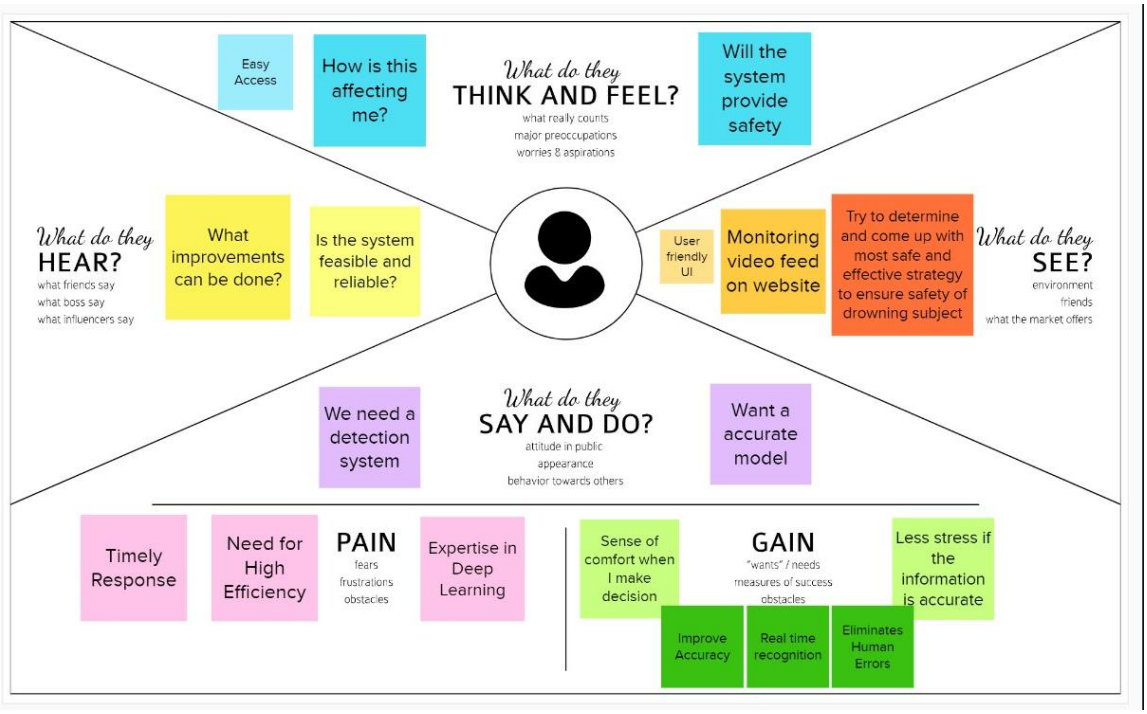
https://www.angeleye.tech/us/us-lifeguard/
https://swimeye.com/
https://www.thewirh.com/blog/dds-how-do-they-work

2.3. Problem Statement Definition

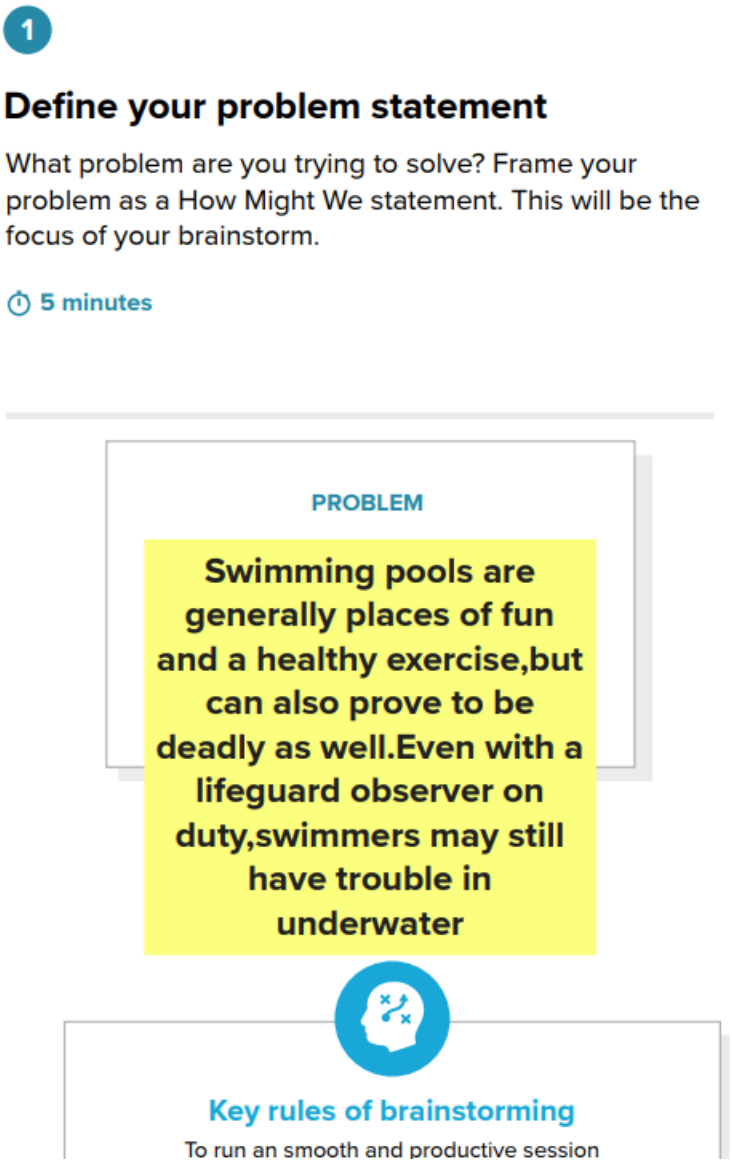
Problem Statements (PS)	I am	I'm trying to	but	Because	Which makes me feel
PS-1	Pool owner	Give high Security	I cann't ensure safety	More likely to drown	Pressure
PS-2	Parents	Get my kids into swimming	I cann't leave him alone to swim	Drowing is more possible	Fear
PS-3	Beginner in swimming	Swim on the pool	It hesitates me a little	I don't know Swimming	Panic
PS-4	Lifeguard	Save the people	I cann't save those people without prior intimation	There is no detection system	Helpless
PS-5	Depressed people	Relax my mind by swimming	I cann't swim on my own	If I accidently drown	Afraid

3.IDEATION & PROPOSED SOLUTION

3.1. Empathy Map Canvas



3.2. Ideation & Brainstorming



2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

MOHAN SAI PK

- Detect victims
- Vision-based surveillance system to monitor swimmers

MUDUNURI SAI KRISHNAM RAJU

- Using YOLO object detection to detect whether a person is drowning or not
- Alarm to notify lifeguard

ANAND PRINCE PURTY

- Real-Time image processing to track swimmers in swimming pools
- Check medical condition before swimming

PRASANNA VENKAT A

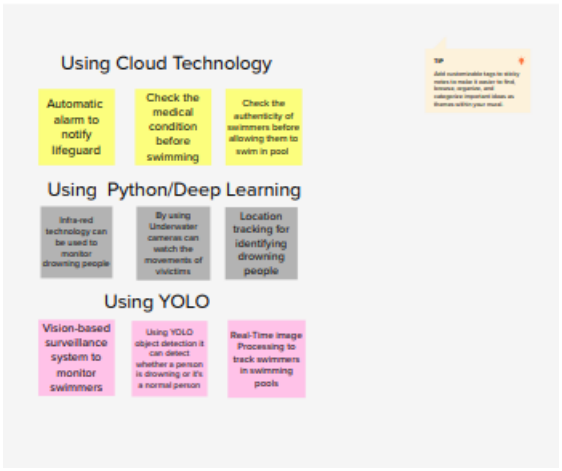
- Infra-red technology can be used to monitor drowning people
- Resuce people by sending lifeguard

3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes



Using Cloud Technology

Automatic alarm to notify lifeguard

Check the medical condition before swimming

Check the authenticity of swimmers before allowing them to swim in pool

Using Python/Deep Learning

Infra-red technology can be used to monitor drowning people

By using Underwater cameras can watch the movements of vivictims

Location tracking for identifying drowning people

Using YOLO

Vision-based surveillance system to monitor swimmers

Using YOLO object detection it can detect whether a person is drowning or it's a normal person

Real-Time image Processing to track swimmers in swimming pools

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 swimmers, who are inexperienced may be more prone to unexpected mishaps such as drowning even when a life-guard is on-duty.
2.	Idea / Solution description	In this project we use AI that works based on YOLO v5 Algorithm. It helps detect potential drowning subjects at individual frame level from a video feed being generated off of a camera that's planted over the swimming pool. Upon a positive detection the life-guard would be alerted through the web application.
3.	Novelty / Uniqueness	The proposed system detects the drowning subjects using an AI that's based off of a YOLO v5 model which yields high accuracy and fast detection speeds.
4.	Social Impact / Customer Satisfaction	With the device planted, the subject would feel safer as it would alert life-guards in case of an active drowning.
5.	Business Model (Revenue Model)	Software based approach can be done for individual clients & adding more features and integrations in future updates would make it profitable for business prospects.
6.	Scalability of the Solution	The system uses IBM Cloud to collect and maintain data, which is also scalable-friendly.

3.4. Problem Solution fit

Project Name: VirtualEye - Life Guard for Swimming Pools to Detect Active Drowning

Project Design – Phase 1: Solution Fit

Team ID: PNT2022TMID02193

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) — CS

Who is your customer?
i.e. working parents of 0-5 y.o. kids

The main customers for our project are:

- Private Swimming Pool Owners
- Home Owners who own a Swimming Pool
- Life-Guards hired at the Swimming Pool

6. CUSTOMER CONSTRAINTS — CC

What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.

- Customers could be skeptical about the accuracy of the detection.
- They can harbor security concerns.

5. AVAILABLE SOLUTIONS — AS

Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking

Prediction process takes place only after drowning but proposed solution uses Deep Learning Algorithm for detection so that there is a chance for detecting drowning accident at earlier stage (i.e., model could also detect partially drowned subjects).

Pros: Detect before the subject has completely drowned.
Cons: If the video feed is broken or obstructed it does not give a result.

Explore AS, differentiate

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

2. JOBS-TO-BE-DONE / PROBLEMS — J&P

Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.

- Detect potential drowning subjects in the Swimming Pool.
- Alert life-guards when a subject is drowning.

9. PROBLEM ROOT CAUSE — RC

What is the real reason that this problem exists?
What is the back story behind the need to do this job?
i.e. customers have to do it because of the change in regulations.

- Life-guard is alerted only when a person has partially/completely drowned.
- Cannot save the person until they have partially drowned.

7. BEHAVIOUR — BE

What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

- Saving people's life.
- Taking effective action in case of an emergency.
- Being attentive and quick in responding to emergencies.

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

Identify strong TR & EM

3. TRIGGERS — TR

Potential subject drowning match in the video frame based on the sample images the model is trained on

4. EMOTIONS: BEFORE / AFTER — EM

Before: Subject being anxious about their safety in swimming pool.

After: With the device planted, the subject would feel safer as it would alert life-guards in case of an active drowning.

10. YOUR SOLUTION — SL

If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.
If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.

The model uses advanced YOLO v5 Algorithm to detect potential drowning subjects which yields higher accuracy and performance compared to existing solutions.

Upon a positive detection an alert would be sent to the Web Application.

8. CHANNELS of BEHAVIOUR — CH

8.1 ONLINE
What kind of actions do customers take online? Extract online channels from #7

Monitoring active swimmers via Web Application.

8.2 OFFLINE
What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.

Be on the look for potential drowning and responding to emergencies.

Extract online & offline CH of BE

4.REQUIREMENT ANALYSIS

4.1. Functional requirement

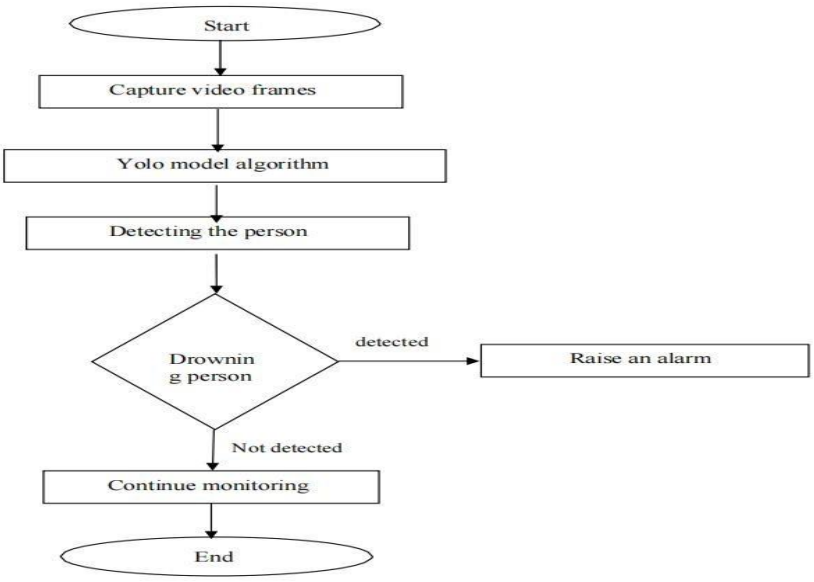
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration Via Email Registration Via phone number
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP Create and store the data
FR-3	Alarm system	Monitor and detect the drowning person Alert the lifeguard by trigger the alarm
FR-4	Output	Visual representation Image detection Report generation

4.2. Non-Functional requirements

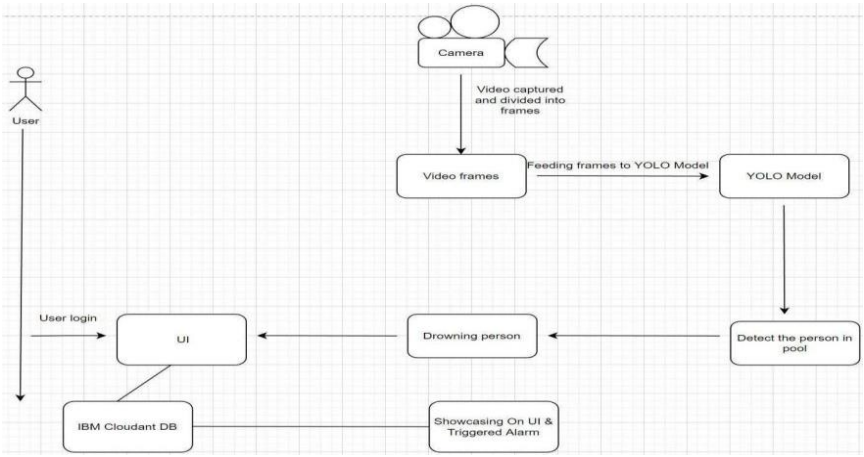
NFR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	To ensure the safety of each and every person present in the pool. A Lifeguard should be present all the time in the pool.

5.PROJECT DESIGN

5.1. Data Flow Diagrams



5.2. Solution & Technical Architecture



5.3. User Stories

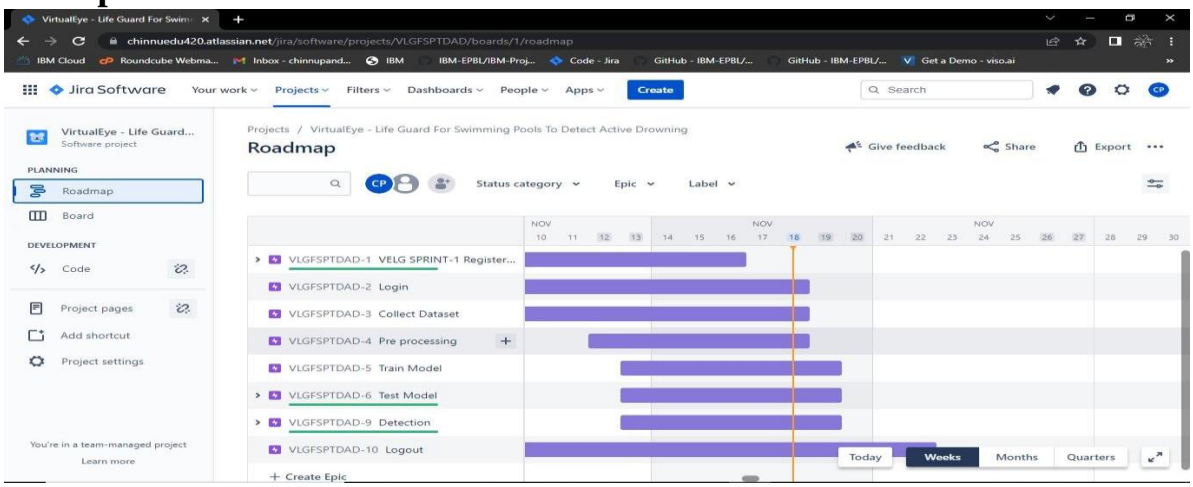
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Prasanna Venkat A Mohan Sai Sai Krishnam Raju Anand Prince
Sprint-1	Registration	USN-2	As a user, I will receive confirmation email once I have registered for the application	1	High	Prasanna Venkat A Mohan Sai Anand Prince
Sprint-1	Registration	USN-3	As a user, I can register for the application through Facebook	2	Low	Prasanna Venkat A Mohan Sai Anand Prince
Sprint-1	Registration	USN-4	As a user, I can register for the application through Gmail	2	Medium	Prasanna Venkat A Mohan Sai Anand Prince
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	1	High	Prasanna Venkat A Mohan Sai Sai Krishnam Raju Anand Prince

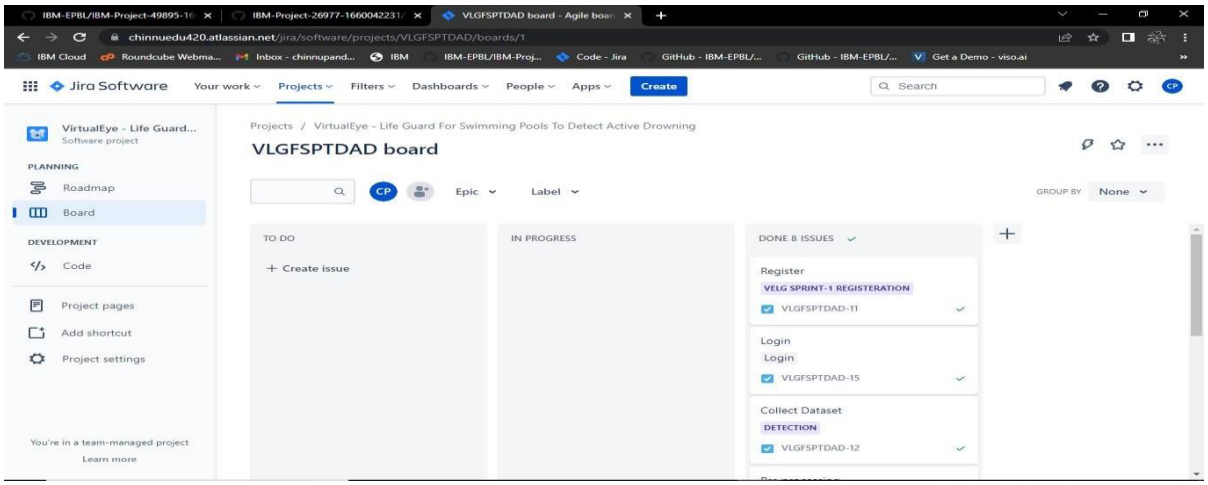
6.PROJECT PLANNING & SCHEDULING

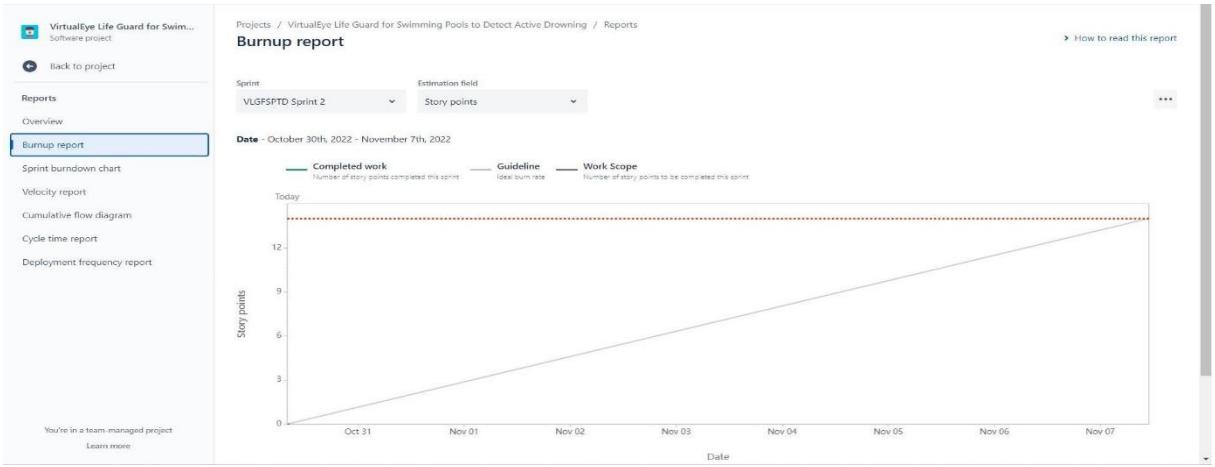
6.1. Sprint Planning & Estimation

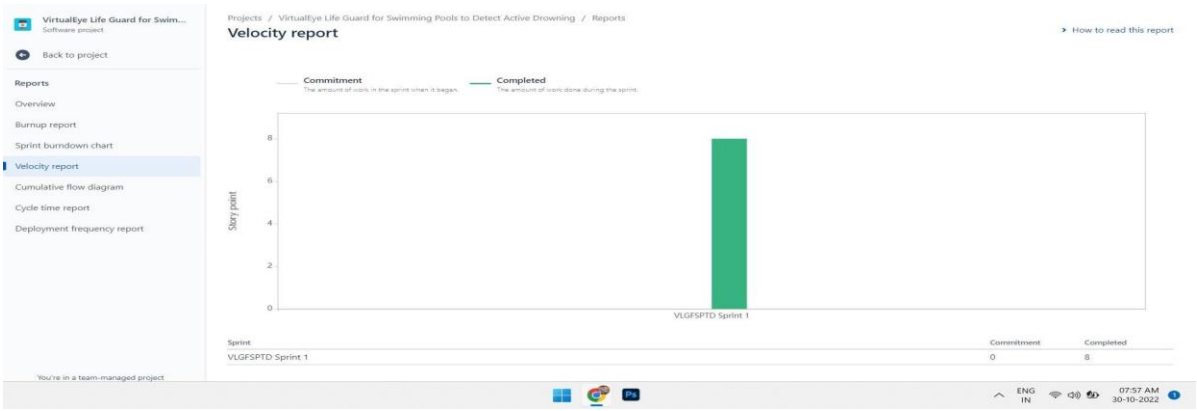
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	8	6 Days	01 Nov 2022	06 Nov 2022	6	06 Nov 2022
Sprint-2	14	4 Days	06 Nov 2022	10 Nov 2022	12	10 Nov 2022
Sprint-3	16	4 Days	10 Nov 2022	14 Nov 2022	11	14 Nov 2022
Sprint-4	12	6 Days	14 Nov 2022	19 Nov 2022	12	19 Nov 2022

6.2. Reports from JIRA









7. CODING & SOLUTIONING

7.1. Feature 1

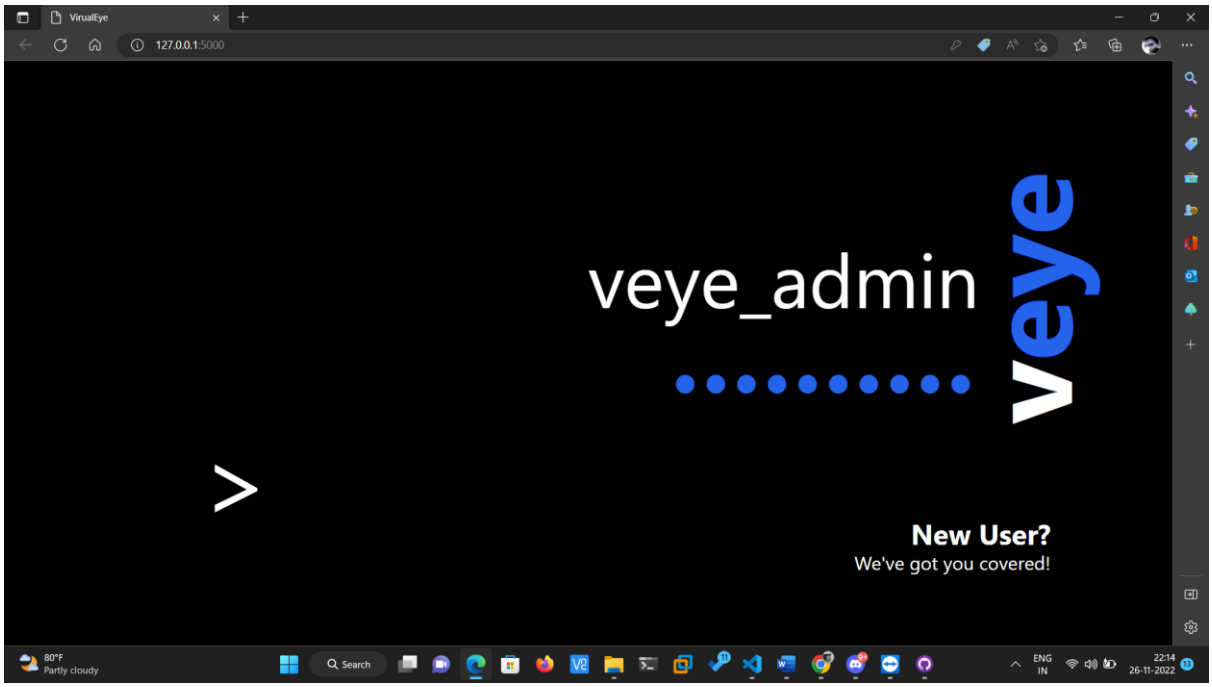
Humans have always had the innate ability to recognize and distinguish between faces. Now computers are able to do the same. This opens up tons of applications. Face detection and recognition is a heavily researched topic and there are tons of resources online. We have tried multiple open source to find the ones that are simplest to implement while being accurate. We have also created a pipeline for detection, recognition and emotion understanding on any input image with just 8 lines of code after the images have been loaded!

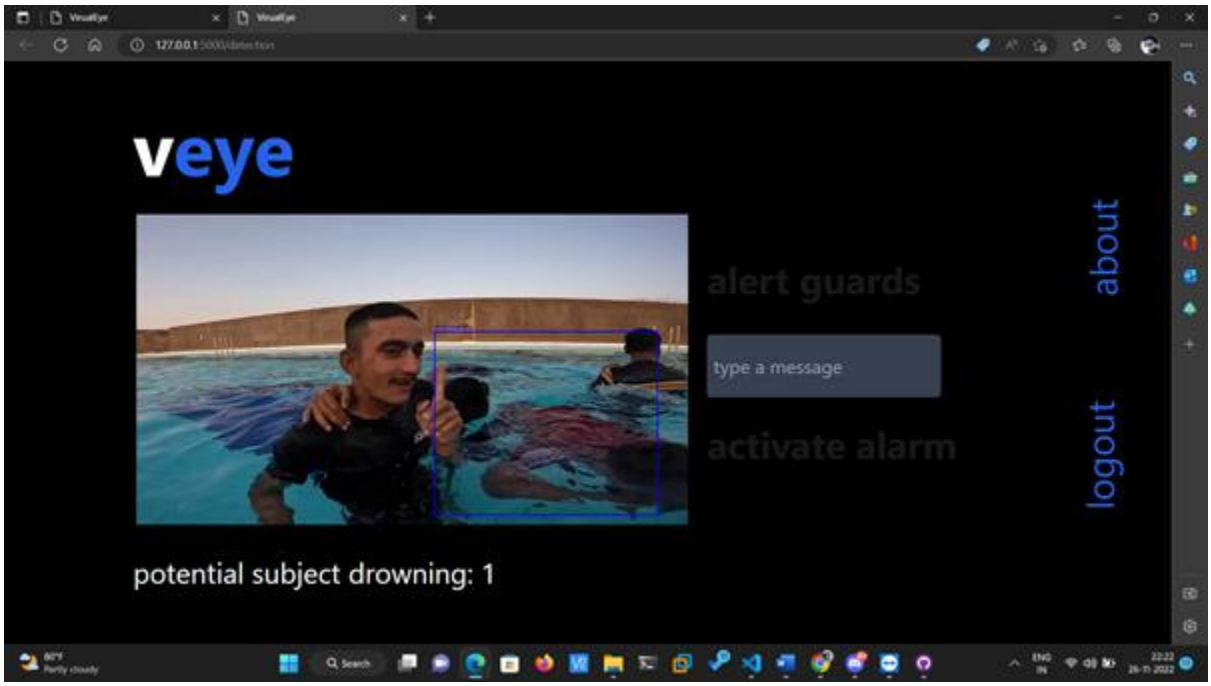
7.2. Feature 2

Most strokes involve rhythmic and coordinated movements of all major body parts — torso, arms, legs, hands, feet, and head.

8. TESTING

8.1. Test Cases





8.2. User Acceptance Testing

1. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	2	0	0	2
Client Application	2	0	0	2
Security	1	0	0	1
Outsource Shipping	1	0	0	1
Exception Reporting	2	0	0	2
Final Report Output	1	0	0	1

2. Test Case Analysis

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	4	2	3	20
Duplicate	1	0	3	0	4
External	2	3	0	1	6
Fixed	11	2	4	20	37
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

This report shows the number of test cases that have passed, failed, and untested

Version Control	1	0	0	1

9. ADVANTAGES & DISADVANTAGES

- ✓ The Approach detected human drifting and drowning up to a range of 5m in water bodies. The final result achieved an average of 82.10% accuracy.
- ✓ Identifies drowning victims in a minimum amount of time and dispatches an automated drone to save them
- × Too much air bubbles generated by the drowning swimmer in the water will also occur. There is a chance that the action cannot be captured by the computer

10. CONCLUSION

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".

11. APPENDIX

Source Code:

// base.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">
    <link rel="stylesheet" href="styles.css">
    <script src="https://cdn.tailwindcss.com"></script>
    <script
      src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scri
      pt>
    <!-- external js -->
    <title>VirualEye</title>
    {% block head %}{% endblock %}
</head>
<body>
```

```
        {% block body %}{% endblock %}
    </body>
</html>
```

// **counter.html**

```
<span id="object_counter">{{ dyn_var }}</span>
```

// **detection.html**

```
    {% extends 'base.html' %}


    {% block head %}
    <link rel="stylesheet" href="styles.css">
    <script
        src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scri
        pt>
    <style>


      #detection-about {
         rotate: 270deg;
      }


      #detection-logout {
         rotate: 270deg;
      }


      #detection-about:hover {
         color: black;
```

```
    background:rgb(67, 67, 245);
    padding: 1vh;
}

#detection-logout:hover {
    color: black;
    background:rgb(67, 67, 245);
    padding: 1vh;
}

#alert-btn:hover {
    background: white;
    color: black;
    border-width: 0 5px 5px 0;
    border-color: black rgb(67, 67, 245) rgb(67, 67, 245) black;
    opacity: 100%;
}

#activateAlarm-btn:hover {
    background: white;
    color: black;
    border-width: 0 5px 5px 0;
    border-color: black rgb(67, 67, 245) rgb(67, 67, 245) black;
    opacity: 100%;
```

```
    }
</style>
<script>

    // Potential Counter

    const delay = ms => new Promise(res => setTimeout(res, ms));

    const update_counter = () => {
        $.ajax({
        url: "/counter",
        type: "POST",
        dataType: "json",
        success: async function(data) {
            console.log("potential subject drowning: " + parseInt(data[1].slice(26,-
      7)))
            $(object_counter).replaceWith(data)
            if (parseInt(data[1].slice(26,-7)) > 0) {
                await delay(2000)
            }
            update_counter()
        }
     })
    }
```

```
    update_counter()


</script>
{% endblock %}


{% block body %}
<body style="overflow: hidden;" class="bg-black">
    <div class="container mx-auto mt-5">
        <div class="container bg-black px-5 grid">
            <h1 class="px-10 py-10 text-8xl text-white font-bold">v<span class="text-blue-600">eye</span></h1>
            <div style="display: inline-block; position: absolute; top: 25%; right: 0">
                <button id="detection-about" class="bg-black text-blue-600 text-5xl m-5 pr-2 pb-2">about</button>
            </div>
            <div style="display: inline-block; position: absolute; top: 60%; right: 0">
                <button id="detection-logout" class="bg-black text-blue-600 text-5xl m-5 pt-2 pr-2 pb-2">logout</button>
            </div>
        </div>
        <div>
            <img style="display: inline-block; position: relative; left: 5%" class="" src="{{url_for('video')}}" width="55%"/>
            <div style="display: inline-block; position: relative; top: -50px; left: 5%">
```

```
        <button id="alert-btn" class="bg-black opacity-10 font-bold text-white text-5xl m-5 pr-8 pb-2">alert guards</button>
        <br/>
        <input id="alert-message"
        style="position: absolute; width: 20vw; height: 80px; border-radius: 5px;"
        class="text-2xl text-white bg-gray-700 m-5 p-2"
        placeholder="type a message">
        <br/>
        <button style="position: fixed; margin-top: 110px" id="activateAlarm-btn" class="bg-black opacity-10 font-bold text-white text-5xl ml-5 mr-5 mb-5 pr-2 pb-2">activate alarm</button>
    </div>
  </div>
  <div class="container bg-black px-5">
    <h1 class="px-10 py-10 text-4xl text-white">potential subject drowning: <span class="text-red-500" id="object_counter">{{ dyn_var }}</span></h1>
  </div>
</div>
<script type="module" src="{{ url_for('static', filename="js_modules/alarm.js") }}"></script>
<script type="module" src="{{ url_for('static', filename="js_modules/sms.js") }}"></script>
<script type="text/javascript">
  var detection_logout = document.getElementById("detection-logout")
```

```
            var detection_about = document.getElementById("detection-about")
            var alert_btn = document.getElementById("alert-btn")
            var activate_alarm = document.getElementById("activateAlarm-btn")

            detection_logout.addEventListener("click", () => {
               window.open("/logout", "_blank")
               window.close()
            })

            detection_about.addEventListener("click", () => {
               window.open("/about", "_blank")
               window.close()
            })


         </script>
      </body>

{% endblock %}
```

// login.html

```
      {% extends 'base.html' %}

      {% block head %}
      <!-- Tailwind-Powered CSS -->
```

```
<link rel="stylesheet" href="styles.css">

<!-- JQuery -->

<script
    src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scri
    pt>

<!-- Bootstrap Imports -->

<link
    href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.cs
    s" rel="stylesheet" integrity="sha384-
    Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1
    WTRi" crossorigin="anonymous">

<script
    src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.
    min.js" integrity="sha384-
    oBqDVmMz9ATKxIep9tiCxS/Z9fNfEXiDAYTujMAeBAsjFuCZSmKbSS
    UnQlmh/jp3" crossorigin="anonymous"></script>

<script
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.min.js"
    integrity="sha384-
    IDwe1+LCz02ROU9k972gdyvl+AESN10+x7tBKgc9I5HFtuNz0wWnPclzo
    6p9vxnk" crossorigin="anonymous"></script>

<style>

   /* to-do: fade-in effect for all successive pages */

</style>

{% endblock %}


{% block body %}
```

```
<body style="max-width: 1440px; margin-left: 5%" class="bg-black overflow-
    hidden place-content-center">
  <h1 style="position: absolute; rotate: 270deg; font-size: 150px; right: 2%"
    class="text-white font-bold">v<span class="text-blue-
    600">eye</span></h1>
  <!-- login container -->
  <div style="margin-top: 15%;" class="container grid place-content-center">

    <form id="login" action="{{url_for("validate_login")}}" method="post">
    <!-- <form id="login" action="" method="post"> -->
      <!-- username -->

      <label for="username"></label>
      <input id="user_login_email"
      style="text-align: right;"
      class="focus:outline-none bg-black text-white text-8xl"
      type="text"
      name="user_login_email"
      placeholder="u/name"/>

      <!-- password -->

      <label for="password"></label>
      <input id="user_login_pass"
      style="text-align: right;"
```

```
    class="focus:outline-none bg-black text-blue-600 text-8xl"
    type="password"
    name="user_login_password"
    placeholder="p/word">

    <p style="text-align: right;
    position: relative;
    right: 14%;"
    class="text-red-600">{{dyn_message}}</p>

 </form>
 <button id="login_btn" form="login" type="submit" style="margin-left: 5%; position: relative; max-width: 150px" class="hover:bg-blue-600 rounded-full text-white text-9xl pl-2 pb-4">></button>

 <div style="position: relative; bottom: 50%" class="container grid place-content-end">
    <button id="register_direct">
       <h1 class="mr-10 p-4 hover:bg-blue-600 text-2xl text-right text-white">
          <span class="font-bold text-4xl">New User?</span></br>We've got you covered!
       </h1>
    </button>
 </div>
```

```
</div>



<script type="text/javascript">

    // inupt elements
    const user_login_email = document.getElementById("user_login_email")
    const user_login_password=
   document.getElementById("user_login_password")

    // buttons
    const login_user_button = document.getElementById("login_btn")
    const register_user_button = document.getElementById("register_direct")

    // button functions
    // login_user_button.addEventListener("click", () => {
    //     sessionStorage.setItem("user_login_email", user_login_email.value)
    //     sessionStorage.setItem("user_login_pass", user_login_password.value)
    //     window.open("/validate_login", "_blank")
    //     window.close()
    // })

    register_user_button.addEventListener("click", () => {
        window.open("/register_intro", "_blank")
    })
```

```
    </script>
  </body>
{% endblock %}
```

// **style.css**

```
/*
! tailwindcss v3.1.8 | MIT License | https://tailwindcss.com
*/


/*
1. Prevent padding and border from affecting element width.
    (https://github.com/mozdevs/cssremedy/issues/4)
2. Allow adding a border to an element by just adding a border-width.
    (https://github.com/tailwindcss/tailwindcss/pull/116)
*/


*,
::before,
::after {
  box-sizing: border-box;
  /* 1 */
  border-width: 0;
  /* 2 */
  border-style: solid;
  /* 2 */
  border-color: #e5e7eb;
```

```
  /* 2 */
}

::before,
::after {
  --tw-content: '';
}

/*
1. Use a consistent sensible line-height in all browsers.
2. Prevent adjustments of font size after orientation changes in iOS.
3. Use a more readable tab size.
4. Use the user's configured `sans` font-family by default.
*/

html {
  line-height: 1.5;
  /* 1 */
  -webkit-text-size-adjust: 100%;
  /* 2 */
  -moz-tab-size: 4;
  /* 3 */
  -o-tab-size: 4;
     tab-size: 4;
```

```
  /* 3 */
  font-family: ui-sans-serif, system-ui, -apple-system, BlinkMacSystemFont,
      "Segoe UI", Roboto, "Helvetica Neue", Arial, "Noto Sans", sans-serif,
      "Apple Color Emoji", "Segoe UI Emoji", "Segoe UI Symbol", "Noto Color
      Emoji";
  /* 4 */
}

/*
1. Remove the margin in all browsers.
2. Inherit line-height from `html` so users can set them as a class directly on the
     `html` element.
*/

body {
  margin: 0;
  /* 1 */
  line-height: inherit;
  /* 2 */
}

/*
1. Add the correct height in Firefox.
2. Correct the inheritance of border color in Firefox.
     (https://bugzilla.mozilla.org/show_bug.cgi?id=190655)
```

```
3. Ensure horizontal rules are visible by default.
*/

hr {
  height: 0;
  /* 1 */
  color: inherit;
  /* 2 */
  border-top-width: 1px;
  /* 3 */
}

/*
Add the correct text decoration in Chrome, Edge, and Safari.
*/

abbr:where([title]) {
  -webkit-text-decoration: underline dotted;
          text-decoration: underline dotted;
}

/*
Remove the default font size and weight for headings.
*/
```

```
h1,
h2,
h3,
h4,
h5,
h6 {
  font-size: inherit;
  font-weight: inherit;
}

/*
Reset links to optimize for opt-in styling instead of opt-out.
*/

a {
  color: inherit;
  text-decoration: inherit;
}

/*
Add the correct font weight in Edge and Safari.
*/
```

```
b,
strong {
  font-weight: bolder;
}

/*
1. Use the user's configured `mono` font family by default.
2. Correct the odd `em` font sizing in all browsers.
*/

code,
kbd,
samp,
pre {
  font-family: ui-monospace, SFMono-Regular, Menlo, Monaco, Consolas,
    "Liberation Mono", "Courier New", monospace;
  /* 1 */
  font-size: 1em;
  /* 2 */
}

/*
Add the correct font size in all browsers.
*/
```

```
small {
  font-size: 80%;
}

/*
Prevent `sub` and `sup` elements from affecting the line height in all browsers.
*/

sub,
sup {
  font-size: 75%;
  line-height: 0;
  position: relative;
  vertical-align: baseline;
}

sub {
  bottom: -0.25em;
}

sup {
  top: -0.5em;
}
```

```
/*
1. Remove text indentation from table contents in Chrome and Safari.
    (https://bugs.chromium.org/p/chromium/issues/detail?id=999088,
    https://bugs.webkit.org/show_bug.cgi?id=201297)
2. Correct table border color inheritance in all Chrome and Safari.
    (https://bugs.chromium.org/p/chromium/issues/detail?id=935729,
    https://bugs.webkit.org/show_bug.cgi?id=195016)
3. Remove gaps between table borders by default.
*/

table {
  text-indent: 0;
  /* 1 */
  border-color: inherit;
  /* 2 */
  border-collapse: collapse;
  /* 3 */
}

/*
1. Change the font styles in all browsers.
2. Remove the margin in Firefox and Safari.
3. Remove default padding in all browsers.
*/
```

```
button,
input,
optgroup,
select,
textarea {
  font-family: inherit;
  /* 1 */
  font-size: 100%;
  /* 1 */
  font-weight: inherit;
  /* 1 */
  line-height: inherit;
  /* 1 */
  color: inherit;
  /* 1 */
  margin: 0;
  /* 2 */
  padding: 0;
  /* 3 */
}

/*
Remove the inheritance of text transform in Edge and Firefox.
*/
```

```
button,
select {
  text-transform: none;
}

/*
1. Correct the inability to style clickable types in iOS and Safari.
2. Remove default button styles.
*/

button,
[type='button'],
[type='reset'],
[type='submit'] {
  -webkit-appearance: button;
  /* 1 */
  background-color: transparent;
  /* 2 */
  background-image: none;
  /* 2 */
}

/*
```

```
Use the modern Firefox focus style for all focusable elements.
*/

:-moz-focusring {
  outline: auto;
}

/*
Remove the additional `:invalid` styles in Firefox.
    (https://github.com/mozilla/gecko-
    dev/blob/2f9eacd9d3d995c937b4251a5557d95d494c9be1/layout/style/res/fo
    rms.css#L728-L737)
*/

:-moz-ui-invalid {
  box-shadow: none;
}

/*
Add the correct vertical alignment in Chrome and Firefox.
*/

progress {
  vertical-align: baseline;
}
```

```
/*
Correct the cursor style of increment and decrement buttons in Safari.
*/

::-webkit-inner-spin-button,
::-webkit-outer-spin-button {
  height: auto;
}

/*
1. Correct the odd appearance in Chrome and Safari.
2. Correct the outline style in Safari.
*/

[type='search'] {
  -webkit-appearance: textfield;
  /* 1 */
  outline-offset: -2px;
  /* 2 */
}

/*
Remove the inner padding in Chrome and Safari on macOS.
```

```
*/

::-webkit-search-decoration {
  -webkit-appearance: none;
}

/*
1. Correct the inability to style clickable types in iOS and Safari.
2. Change font properties to `inherit` in Safari.
*/

::-webkit-file-upload-button {
  -webkit-appearance: button;
  /* 1 */
  font: inherit;
  /* 2 */
}

/*
Add the correct display in Chrome and Safari.
*/

summary {
  display: list-item;
```

```
}

/*
Removes the default spacing and border for appropriate elements.
*/

blockquote,
dl,
dd,
h1,
h2,
h3,
h4,
h5,
h6,
hr,
figure,
p,
pre {
  margin: 0;
}

fieldset {
  margin: 0;
```

```
  padding: 0;
}

legend {
  padding: 0;
}

ol,
ul,
menu {
  list-style: none;
  margin: 0;
  padding: 0;
}

/*
Prevent resizing textareas horizontally by default.
*/

textarea {
  resize: vertical;
}

/*
```

```
1. Reset the default placeholder opacity in Firefox.
    (https://github.com/tailwindlabs/tailwindcss/issues/3300)
2. Set the default placeholder color to the user's configured gray 400 color.
*/

input::-moz-placeholder, textarea::-moz-placeholder {
  opacity: 1;
  /* 1 */
  color: #9ca3af;
  /* 2 */
}

input::placeholder,
textarea::placeholder {
  opacity: 1;
  /* 1 */
  color: #9ca3af;
  /* 2 */
}

/*
Set the default cursor for buttons.
*/

button,
```

```
[role="button"] {
  cursor: pointer;
}

/*
Make sure disabled buttons don't get the pointer cursor.
*/

:disabled {
  cursor: default;
}

/*
1. Make replaced elements `display: block` by default.
    (https://github.com/mozdevs/cssremedy/issues/14)
2. Add `vertical-align: middle` to align replaced elements more sensibly by
    default.
    (https://github.com/jensimmons/cssremedy/issues/14#issuecomment-
    634934210)
   This can trigger a poorly considered lint error in some tools but is included by
    design.
*/

img,
svg,
```

```
video,
canvas,
audio,
iframe,
embed,
object {
  display: block;
  /* 1 */
  vertical-align: middle;
  /* 2 */
}

/*
Constrain images and videos to the parent width and preserve their intrinsic
      aspect ratio. (https://github.com/mozdevs/cssremedy/issues/14)
*/

img,
video {
  max-width: 100%;
  height: auto;
}

*, ::before, ::after {
  --tw-border-spacing-x: 0;
```

```
--tw-border-spacing-y: 0;
--tw-translate-x: 0;
--tw-translate-y: 0;
--tw-rotate: 0;
--tw-skew-x: 0;
--tw-skew-y: 0;
--tw-scale-x: 1;
--tw-scale-y: 1;
--tw-pan-x:  ;
--tw-pan-y:  ;
--tw-pinch-zoom:  ;
--tw-scroll-snap-strictness: proximity;
--tw-ordinal:  ;
--tw-slashed-zero:  ;
--tw-numeric-figure:  ;
--tw-numeric-spacing:  ;
--tw-numeric-fraction:  ;
--tw-ring-inset:  ;
--tw-ring-offset-width: 0px;
--tw-ring-offset-color: #fff;
--tw-ring-color: rgb(59 130 246 / 0.5);
--tw-ring-offset-shadow: 0 0 #0000;
--tw-ring-shadow: 0 0 #0000;
--tw-shadow: 0 0 #0000;
```

```
    --tw-shadow-colored: 0 0 #0000;
    --tw-blur:  ;
    --tw-brightness:  ;
    --tw-contrast:  ;
    --tw-grayscale:  ;
    --tw-hue-rotate:  ;
    --tw-invert:  ;
    --tw-saturate:  ;
    --tw-sepia:  ;
    --tw-drop-shadow:  ;
    --tw-backdrop-blur:  ;
    --tw-backdrop-brightness:  ;
    --tw-backdrop-contrast:  ;
    --tw-backdrop-grayscale:  ;
    --tw-backdrop-hue-rotate:  ;
    --tw-backdrop-invert:  ;
    --tw-backdrop-opacity:  ;
    --tw-backdrop-saturate:  ;
    --tw-backdrop-sepia:  ;
  }

  ::-webkit-backdrop {
    --tw-border-spacing-x: 0;
    --tw-border-spacing-y: 0;
```

```
--tw-translate-x: 0;
--tw-translate-y: 0;
--tw-rotate: 0;
--tw-skew-x: 0;
--tw-skew-y: 0;
--tw-scale-x: 1;
--tw-scale-y: 1;
--tw-pan-x:  ;
--tw-pan-y:  ;
--tw-pinch-zoom:  ;
--tw-scroll-snap-strictness: proximity;
--tw-ordinal:  ;
--tw-slashed-zero:  ;
--tw-numeric-figure:  ;
--tw-numeric-spacing:  ;
--tw-numeric-fraction:  ;
--tw-ring-inset:  ;
--tw-ring-offset-width: 0px;
--tw-ring-offset-color: #fff;
--tw-ring-color: rgb(59 130 246 / 0.5);
--tw-ring-offset-shadow: 0 0 #0000;
--tw-ring-shadow: 0 0 #0000;
--tw-shadow: 0 0 #0000;
--tw-shadow-colored: 0 0 #0000;
```

```
    --tw-blur:  ;
    --tw-brightness:  ;
    --tw-contrast:  ;
    --tw-grayscale:  ;
    --tw-hue-rotate:  ;
    --tw-invert:  ;
    --tw-saturate:  ;
    --tw-sepia:  ;
    --tw-drop-shadow:  ;
    --tw-backdrop-blur:  ;
    --tw-backdrop-brightness:  ;
    --tw-backdrop-contrast:  ;
    --tw-backdrop-grayscale:  ;
    --tw-backdrop-hue-rotate:  ;
    --tw-backdrop-invert:  ;
    --tw-backdrop-opacity:  ;
    --tw-backdrop-saturate:  ;
    --tw-backdrop-sepia:  ;
  }

  ::backdrop {
    --tw-border-spacing-x: 0;
    --tw-border-spacing-y: 0;
    --tw-translate-x: 0;
```

```
--tw-translate-y: 0;
--tw-rotate: 0;
--tw-skew-x: 0;
--tw-skew-y: 0;
--tw-scale-x: 1;
--tw-scale-y: 1;
--tw-pan-x:  ;
--tw-pan-y:  ;
--tw-pinch-zoom:  ;
--tw-scroll-snap-strictness: proximity;
--tw-ordinal:  ;
--tw-slashed-zero:  ;
--tw-numeric-figure:  ;
--tw-numeric-spacing:  ;
--tw-numeric-fraction:  ;
--tw-ring-inset:  ;
--tw-ring-offset-width: 0px;
--tw-ring-offset-color: #fff;
--tw-ring-color: rgb(59 130 246 / 0.5);
--tw-ring-offset-shadow: 0 0 #0000;
--tw-ring-shadow: 0 0 #0000;
--tw-shadow: 0 0 #0000;
--tw-shadow-colored: 0 0 #0000;
--tw-blur:  ;
```

```
  --tw-brightness:  ;
  --tw-contrast:  ;
  --tw-grayscale:  ;
  --tw-hue-rotate:  ;
  --tw-invert:  ;
  --tw-saturate:  ;
  --tw-sepia:  ;
  --tw-drop-shadow:  ;
  --tw-backdrop-blur:  ;
  --tw-backdrop-brightness:  ;
  --tw-backdrop-contrast:  ;
  --tw-backdrop-grayscale:  ;
  --tw-backdrop-hue-rotate:  ;
  --tw-backdrop-invert:  ;
  --tw-backdrop-opacity:  ;
  --tw-backdrop-saturate:  ;
  --tw-backdrop-sepia:  ;
}

.container {
  width: 100%;
}

@media (min-width: 640px) {
```

```
  .container {
    max-width: 640px;
  }
}

@media (min-width: 768px) {
  .container {
    max-width: 768px;
  }
}

@media (min-width: 1024px) {
  .container {
    max-width: 1024px;
  }
}

@media (min-width: 1280px) {
  .container {
    max-width: 1280px;
  }
}

@media (min-width: 1536px) {
```

```
  .container {
    max-width: 1536px;
  }
}

.mx-auto {
  margin-left: auto;
  margin-right: auto;
}

.my-auto {
  margin-top: auto;
  margin-bottom: auto;
}

.my-5 {
  margin-top: 1.25rem;
  margin-bottom: 1.25rem;
}

.mx-5 {
  margin-left: 1.25rem;
  margin-right: 1.25rem;
}
```

```
.mt-10 {
  margin-top: 2.5rem;
}

.mt-5 {
  margin-top: 1.25rem;
}

.ml-5 {
  margin-left: 1.25rem;
}

.ml-10 {
  margin-left: 2.5rem;
}

.ml-20 {
  margin-left: 5rem;
}

.block {
  display: block;
}
```

```
.inline-block {
  display: inline-block;
}

.rounded-lg {
  border-radius: 0.5rem;
}

.border-2 {
  border-width: 2px;
}

.bg-slate-600 {
  --tw-bg-opacity: 1;
  background-color: rgb(71 85 105 / var(--tw-bg-opacity));
}

.px-5 {
  padding-left: 1.25rem;
  padding-right: 1.25rem;
}

.px-10 {
```

```
  padding-left: 2.5rem;
  padding-right: 2.5rem;
}

.py-10 {
  padding-top: 2.5rem;
  padding-bottom: 2.5rem;
}

.text-4xl {
  font-size: 2.25rem;
  line-height: 2.5rem;
}

.text-sm {
  font-size: 0.875rem;
  line-height: 1.25rem;
}

.text-8xl {
  font-size: 6rem;
  line-height: 1;
}
```

```
        .text-xl {
          font-size: 1.25rem;
          line-height: 1.75rem;
        }

        .font-bold {
          font-weight: 700;
        }

        .text-white {
          --tw-text-opacity: 1;
          color: rgb(255 255 255 / var(--tw-text-opacity));
        }

        .text-red-500 {
          --tw-text-opacity: 1;
          color: rgb(239 68 68 / var(--tw-text-opacity));
}
```

// login_failed.html

```
        {% extends 'base.html' %}

        {% block head %}
        <!-- Tailwind-Powered CSS -->
        <link rel="stylesheet" href="styles.css">
```

```
<!-- JQuery -->
<script
    src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scri
    pt>
<!-- Bootstrap Imports -->
<link
    href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.cs
    s" rel="stylesheet" integrity="sha384-
    Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1
    WTRi" crossorigin="anonymous">
<script
    src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.
    min.js" integrity="sha384-
    oBqDVmMz9ATKxIep9tiCxS/Z9fNfEXiDAYTujMAeBAsjFuCZSmKbSS
    UnQlmh/jp3" crossorigin="anonymous"></script>
<script
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.min.js"
    integrity="sha384-
    IDwe1+LCz02ROU9k972gdyvl+AESN10+x7tBKgc9I5HFtuNz0wWnPclzo
    6p9vxnk" crossorigin="anonymous"></script>
{% endblock %}


{% block body %}
<body style="max-width: 1440px; margin-left: 5%" class="bg-black overflow-
    hidden place-content-center">
  <!-- login container -->
  <div style="margin-top: 15%;" class="container grid place-content-center">
```

```
    <form id="login" action="" method="post">

        <!-- user registration intro -->

        <!-- <h1 class="text-white text-9xl">you're <span class="text-blue-
    600">in</span>!</h1> -->
        <h1 class="text-white text-9xl">{{dyn_message}}!</h1>
    </form>
    <!-- <h1 style="display: inline-block" class="text-3xl text-white">get
    started</h1> -->
</div>


<script type="text/javascript">

const delay = (delayInms) => {
        return new Promise(resolve => setTimeout(resolve, delayInms));
    }

const sample = async () => {
    console.log("delay activated")
    let delayLogin = await delay(3000);

    window.open("/login_failed", "_blank")
    window.close()
```

```
            }
            sample();
            </script>
        </body>
{% endblock %}
```

// **login_redirect.html**

```
        {% extends 'base.html' %}


        {% block head %}
        <!-- Tailwind-Powered CSS -->
        <link rel="stylesheet" href="styles.css">
        <!-- JQuery -->
        <script
            src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scri
            pt>
        <!-- Bootstrap Imports -->
        <link
            href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.cs
            s" rel="stylesheet" integrity="sha384-
            Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1
            WTRi" crossorigin="anonymous">
        <script
            src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.
            min.js" integrity="sha384-
            oBqDVmMz9ATKxIep9tiCxS/Z9fNfEXiDAYTujMAeBAsjFuCZSmKbSS
            UnQlmh/jp3" crossorigin="anonymous"></script>
```

```
<script
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.min.js"
    integrity="sha384-
    IDwe1+LCz02ROU9k972gdyvl+AESN10+x7tBKgc9I5HFtuNz0wWnPclzo
    6p9vxnk" crossorigin="anonymous"></script>
{% endblock %}


{% block body %}
<body style="max-width: 1440px; margin-left: 5%" class="bg-black overflow-
    hidden place-content-center">
  <!-- login container -->
  <div style="margin-top: 15%;" class="container grid place-content-center">

    <form id="login" action="" method="post">

      <!-- user registration intro -->

      <!-- <h1 class="text-white text-9xl">you're <span class="text-blue-
    600">in</span>!</h1> -->
      <h1 class="text-white text-9xl">{{dyn_message}}!</h1>
    </form>
    <!-- <h1 style="display: inline-block" class="text-3xl text-white">get
    started</h1> -->
  </div>
```

```
            <script type="text/javascript">


            const delay = (delayInms) => {
                return new Promise(resolve => setTimeout(resolve, delayInms));
              }


            const sample = async () => {
              console.log("delay activated")
              let delayLogin = await delay(3000);


              window.open("login.html", "_blank")
              window.close()
            }
            sample();
            </script>
          </body>
{% endblock %}
```

// login_success.html

```
          {% extends 'base.html' %}


          {% block head %}
          <!-- Tailwind-Powered CSS -->
          <link rel="stylesheet" href="styles.css">
          <!-- JQuery -->
```

```
<script
    src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scri
    pt>
<!-- Bootstrap Imports -->
<link
    href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.cs
    s" rel="stylesheet" integrity="sha384-
    Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1
    WTRi" crossorigin="anonymous">
<script
    src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.
    min.js" integrity="sha384-
    oBqDVmMz9ATKxIep9tiCxS/Z9fNfEXiDAYTujMAeBAsjFuCZSmKbSS
    UnQlmh/jp3" crossorigin="anonymous"></script>
<script
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.min.js"
    integrity="sha384-
    IDwe1+LCz02ROU9k972gdyvl+AESN10+x7tBKgc9I5HFtuNz0wWnPclzo
    6p9vxnk" crossorigin="anonymous"></script>
{% endblock %}


{% block body %}
<body style="max-width: 1440px; margin-left: 5%" class="bg-black overflow-
    hidden place-content-center">
  <!-- login container -->
  <div style="margin-top: 15%;" class="container grid place-content-center">


    <form id="login" action="" method="post">
```

```
        <!-- user registration intro -->


        <!-- <h1 class="text-white text-9xl">you're <span class="text-blue-
    600">in</span>!</h1> -->
        <h1 class="text-white text-9xl">{{dyn_message}}!</h1>
    </form>
    <!-- <h1 style="display: inline-block" class="text-3xl text-white">get
    started</h1> -->
</div>



<script type="text/javascript">

const delay = (delayInms) => {
        return new Promise(resolve => setTimeout(resolve, delayInms));
    }

const sample = async () => {
    console.log("delay activated")
    let delayLogin = await delay(3000);

    window.open("/detection", "_blank")
    window.close()
}
```

```
            sample();
            </script>
        </body>
{% endblock %}
```

// register_name.html

```
        {% extends 'base.html' %}


        {% block head %}
        <!-- Tailwind-Powered CSS -->
        <link rel="stylesheet" href="styles.css">
        <!-- JQuery -->
        <script
            src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scri
            pt>
        <!-- Bootstrap Imports -->
        <link
            href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.cs
            s" rel="stylesheet" integrity="sha384-
            Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1
            WTRi" crossorigin="anonymous">
        <script
            src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.
            min.js" integrity="sha384-
            oBqDVmMz9ATKxIep9tiCxS/Z9fNfEXiDAYTujMAeBAsjFuCZSmKbSS
            UnQlmh/jp3" crossorigin="anonymous"></script>
        <script
            src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.min.js"
```

```
        integrity="sha384-
        IDwe1+LCz02ROU9k972gdyvl+AESN10+x7tBKgc9I5HFtuNz0wWnPclzo
        6p9vxnk" crossorigin="anonymous"></script>
    {% endblock %}


    {% block body %}
    <body style="max-width: 1440px; margin-left: 5%" class="bg-black overflow-
        hidden place-content-center">
      <h1 style="position: absolute; rotate: 270deg; font-size: 150px; right: 2%"
        class="text-white font-bold">v<span class="text-blue-
        600">eye</span></h1>
      <!-- login container -->
      <div style="margin-top: 20%;" class="container grid place-content-center">


        <form id="register_name" action="{{url_for("register_email")}}"
        method="post">


          <!-- get name -->


          <label for="user_name">
            <h1 class="text-white text-3xl">you must have a <span class="text-
        blue-600">name</span><span style="display: block; font-style: italic;"> or
        should I call you mine? :v</span></h1>
          </label>
          <input id="user_name"
          style="text-align: right;"
```

```
                class="focus:outline-none bg-black text-white text-8xl"
                type="text"
                name="user_name"
                placeholder="e.g. 'Anand Prince'"/>


            </form>
            <button form="register_name" id="register_direct_email" type="submit"
            style="margin-left: 5%; position: relative; max-width: 150px"
            class="hover:bg-blue-600 rounded-full text-white text-9xl pl-2 pb-4">
                ></button>
        </div>



        <script type="text/javascript">
            const register_user_email =
            document.getElementById("register_direct_email")


    //        register_user_email.addEventListener("click", () => {
    //            window.open("/register_email", "_blank")
    //            window.close()


    // })
        </script>
    </body>
{% endblock %}
```

// register_email.html

```
{% extends 'base.html' %}


{% block head %}
<!-- Tailwind-Powered CSS -->
<link rel="stylesheet" href="styles.css">
<!-- JQuery -->
<script
    src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scri
    pt>
<!-- Bootstrap Imports -->
<link
    href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.cs
    s" rel="stylesheet" integrity="sha384-
    Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1
    WTRi" crossorigin="anonymous">
<script
    src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.
    min.js" integrity="sha384-
    oBqDVmMz9ATKxIep9tiCxS/Z9fNfEXiDAYTujMAeBAsjFuCZSmKbSS
    UnQlmh/jp3" crossorigin="anonymous"></script>
<script
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.min.js"
    integrity="sha384-
    IDwe1+LCz02ROU9k972gdyvl+AESN10+x7tBKgc9I5HFtuNz0wWnPclzo
    6p9vxnk" crossorigin="anonymous"></script>
{% endblock %}
```

```
{% block body %}
<body style="max-width: 1440px; margin-left: 5%" class="bg-black overflow-
    hidden place-content-center">
  <h1 style="position: absolute; rotate: 270deg; font-size: 150px; right: 2%"
    class="text-white font-bold">v<span class="text-blue-
    600">eye</span></h1>
  <!-- login container -->
  <div style="margin-top: 20%;" class="container grid place-content-center">

    <form id="register_email" action="{{url_for("register_password")}}"
    method="post">

      <!-- get email -->

      <label for="user_email">
        <h1 class="text-white text-3xl">where do we <span class="text-blue-
    600">mail</span> you at? <span style="display:block" id="email_hint"
    class="text-gray-600 italic">number 6, 5th avenue? jk, it's electronic mail
    xD</span></h1>
      </label>
      <input id="user_email"
      style="text-align: right;"
      class="focus:outline-none bg-black text-white text-8xl"
      type="text"
      name="user_email"
```

```
        placeholder="e/mail"/>


    </form>
    <button form="register_email" id="register_direct_password"
    type="submit" style="margin-left: 5%; position: relative; max-width: 150px"
    class="hover:bg-blue-600 rounded-full text-white text-9xl pl-2 pb-4">
        ></button>
</div>



<script type="text/javascript">
    const register_user =
    document.getElementById("register_direct_password")

    // register_user.addEventListener("click", () => {
    //     window.open("/register_password", "_blank")
    //     window.close()
    // })

    const user_email = document.getElementById("user_email")
    const email_hint = document.getElementById("email_hint")

    user_email.addEventListener("change", () => {
        console.log(user_email.value.length)
        user_email.value.length > 0 ? email_hint.style.display = "none" :
```

```
            email_hint.style.display = "block"
             })
           </script>
        </body>
{% endblock %}
```

// register_password.html

```
        {% extends 'base.html' %}


        {% block head %}
        <!-- Tailwind-Powered CSS -->
        <link rel="stylesheet" href="styles.css">
        <!-- JQuery -->
        <script
            src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scri
            pt>
        <!-- Bootstrap Imports -->
        <link
            href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.cs
            s" rel="stylesheet" integrity="sha384-
            Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1
            WTRi" crossorigin="anonymous">
        <script
            src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.
            min.js" integrity="sha384-
            oBqDVmMz9ATKxIep9tiCxS/Z9fNfEXiDAYTujMAeBAsjFuCZSmKbSS
            UnQlmh/jp3" crossorigin="anonymous"></script>
        <script
```

```
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.min.js"
    integrity="sha384-
    IDwe1+LCz02ROU9k972gdyvl+AESN10+x7tBKgc9I5HFtuNz0wWnPclzo
    6p9vxnk" crossorigin="anonymous"></script>
{% endblock %}

{% block body %}
<body style="max-width: 1440px; margin-left: 5%" class="bg-black overflow-
    hidden place-content-center">
  <h1 style="position: absolute; rotate: 270deg; font-size: 150px; right: 2%"
    class="text-white font-bold">v<span class="text-blue-
    600">eye</span></h1>
  <!-- login container -->
  <div style="margin-top: 20%;" class="container grid place-content-center">

    <form id="register_password"
    action="{{url_for("register_phoneNumber")}}" method="post">

      <!-- get name -->

      <label for="user_password">
        <h1 class="text-white text-3xl">tell us a <span class="text-blue-
    600">pass</span> we'd use to let you in<span style="display:block"
    id="pass_hint" class="text-gray-600 italic">dw, we're not gonnna tell
    anybody, hope you wouldn't too ;D</span></h1>
      </label>
      <input id="user_pass"
```

```
        style="text-align: right;"
        class="focus:outline-none bg-black text-blue-600 text-8xl"
        type="text"
        name="user_pass"
        placeholder="p/word"/>

    </form>
    <button form="register_password" id="register_direct_phoneNumber"
   type="submit" style="margin-left: 5%; position: relative; max-width: 150px"
   class="hover:bg-blue-600 rounded-full text-white text-9xl pl-2 pb-4">
        ></button>
</div>



<script type="text/javascript">
    const register_user_password =
   document.getElementById("register_direct_phoneNumber")

    // register_user_password.addEventListener("click", () => {
    //     // window.alert("button pressed")
    //     window.open("/register_phoneNumber", "_blank")
    //     window.close()
    // })

    const user_pass = document.getElementById("user_pass")
```

```
            const pass_hint = document.getElementById("pass_hint")


            user_pass.addEventListener("change", () => {
                console.log(user_pass.value.length)
                user_pass.value.length > 0 ? pass_hint.style.display = "none" :
            pass_hint.style.display = "block"
            })


            // get user password and hide it


        </script>
    </body>
{% endblock %}
```

// register_phoneNumber.html

```
    {% extends 'base.html' %}


    {% block head %}
    <!-- Tailwind-Powered CSS -->
    <link rel="stylesheet" href="styles.css">
    <!-- JQuery -->
    <script
        src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></scri
        pt>
    <!-- Bootstrap Imports -->
    <link
```

```
        href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.cs
        s" rel="stylesheet" integrity="sha384-
        Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1
        WTRi" crossorigin="anonymous">
    <script
        src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.6/dist/umd/popper.
        min.js" integrity="sha384-
        oBqDVmMz9ATKxIep9tiCxS/Z9fNfEXiDAYTujMAeBAsjFuCZSmKbSS
        UnQlmh/jp3" crossorigin="anonymous"></script>
    <script
        src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.min.js"
        integrity="sha384-
        IDwe1+LCz02ROU9k972gdyvl+AESN10+x7tBKgc9I5HFtuNz0wWnPclzo
        6p9vxnk" crossorigin="anonymous"></script>
    {% endblock %}


    {% block body %}
    <body style="max-width: 1440px; margin-left: 5%" class="bg-black overflow-
        hidden place-content-center">
      <h1 style="position: absolute; rotate: 270deg; font-size: 150px; right: 2%"
        class="text-white font-bold">v<span class="text-blue-
        600">eye</span></h1>
      <!-- login container -->
      <div style="margin-top: 20%;" class="container grid place-content-center">


        <form id="register_phone" action="{{url_for("register_outro")}}"
        method="post">
```

```
        <!-- get name -->


        <label for="user_phone">

            <h1 class="text-white text-3xl">final step; your <span class="text-
        blue-600">phone number</span> to set you in touch with us<span
        style="display:block" id="phone_hint" class="text-gray-600 italic">for
        alerts just in case</span></h1>

        </label>

        <input id="user_phone"

        style="text-align: right;"

        class="focus:outline-none bg-black text-white text-8xl"

        type="text"

        name="user_phone"

        placeholder="p/number"/>


    </form>

    <button form="register_phone" id="register_direct_outro" type="submit"
    style="margin-left: 5%; position: relative; max-width: 150px"
    class="hover:bg-blue-600 rounded-full text-white text-9xl pl-2 pb-4">

        ></button>

</div>



<script type="text/javascript">

    const register_user_phoneNumber =
    document.getElementById("register_direct_outro")
```

```
            // register_user_phoneNumber.addEventListener("click", () => {
            //     window.open("/register_outro", "_blank")
            //     window.close()
            // })
        </script>
    </body>
{% endblock %}
```

// alarm.js

```
    var alarm = document.getElementById("activateAlarm-btn")
    var audio = new Audio("alarm.mp3")

    alarm.addEventListener("click", ()=>{
        // window.alert("alarm button pressed!")
        audio.play()
})
```

// sms.js

```
    import { Vonage } from "@vonage/server-sdk";
    var alert_message = document.getElementById("alert-message")
    var activate_alarm = document.getElementById("activateAlarm-btn")
        activate_alarm.addEventListener("click", () => {
            // window.alert("alarm button pressed!")
            console.log(alert_message.value)
        })
```

```
const vonage = new Vonage({
    apiKey: '<key>',
    apiSecret: '<secret>'
})
const from = "Vonage APIs"
const to = "<phone_number>"
const text = alert_message

vonage.message.sendSMs(from, to, text, (err, responseData) => {
    if (err) {
        console.log(err);
    } else {
        if(responseData.messages[0]['status'] === "0") {
            console.log("Message sent successfully.");
        } else {
            console.log(`Message failed with error:
        ${responseData.messages[0]['error-text']}`);
        }
    }
})
```

// **app.py**

```
from flask import Flask, render_template, Response, jsonify, request
import cv2
import numpy as np
```

```
# for accessing session storage
from flask import session, redirect

# cloudant imports
from cloudant.client import Cloudant

# sub-imports
# from object_detection import Detect

# connecting client with cloudant db
client = Cloudant.iam('5e67dcf0-6dd2-49ef-ba49-548e2376d5fa-bluemix',
                      'T0BBzOvBQK6JyezcCq1xelsmRiuVe-AQ1PwdufX_3XCL',
                      connect = True)

db = client.create_database('veye_users')

app=Flask(__name__)

class Detect:

    def __init__(self, video_source,
                      classes,
                      config,
```

```
                weights,
                frame_title,
                wait_key,
                threshold,
                suppression_threshold,
                yolo_image_size):



                self.video_source = video_source
                self.classes = classes
                self.config = config
                self. weights = weights
                self.frame_title = frame_title
                self.wait_key = wait_key
                self.threshold = threshold
                self.suppression_threshold = suppression_threshold
                self.yolo_image_size = yolo_image_size
                self.detect_count = 0


        def find_objects(self, model_outputs, YOLO_IMAGE_SIZE, THRESHOLD,
          SUPPRESSION_THRESHOLD):
           bounding_box_locations = []
           class_ids = []
           confidence_values = []
```

```
        for output in model_outputs:
            for prediction in output:
                class_probabilities = prediction[5:]
                class_id = np.argmax(class_probabilities)
                confidence = class_probabilities[class_id]

                if confidence > THRESHOLD:
                    w, h = int(prediction[2] * YOLO_IMAGE_SIZE), int(prediction[3] * YOLO_IMAGE_SIZE)
                    # the center of the bounding box (we should transform these values)
                    x, y = int(prediction[0] * YOLO_IMAGE_SIZE - w / 2), int(prediction[1] * YOLO_IMAGE_SIZE - h / 2)
                    bounding_box_locations.append([x, y, w, h])
                    class_ids.append(class_id)
                    confidence_values.append(float(confidence))

        box_indexes_to_keep = cv2.dnn.NMSBoxes(bounding_box_locations, confidence_values, THRESHOLD, SUPPRESSION_THRESHOLD)

        return box_indexes_to_keep, bounding_box_locations, class_ids, confidence_values

    def mark_detected_objects(self, img, bounding_box_ids, all_bounding_boxes, class_ids, confidence_values, width_ratio,
                              height_ratio):
```

```
 for index in bounding_box_ids:
    bounding_box = all_bounding_boxes[index]
    x, y, w, h = int(bounding_box[0]), int(bounding_box[1]), int(bounding_box[2]), int(bounding_box[3])

    # we have to transform the locations and coordinates because the image is resized

    x = int(x * width_ratio)
    y = int(y * height_ratio)
    w = int(w * width_ratio)
    h = int(h * height_ratio)

    # OpenCV deals with BGR blue green red (255,0,0) then it is the blue color
    # we are not going to detect every objects just PERSON and CAR
    # if class_ids[index] == 2:
    #     cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2)
    #     class_with_confidence = 'CAR' + str(int(confidence_values[index] * 100)) + '%'
    #     cv2.putText(img, class_with_confidence, (x, y-10), cv2.FONT_HERSHEY_COMPLEX_SMALL, 0.5, (255, 0, 0), 1)
    if class_ids[index] == 0:

        self.detect_count += 1
```

```
                cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2)
                class_with_confidence = f'drowning' +
            str(int(confidence_values[index] * 100)) + '%'
                cv2.putText(img, class_with_confidence, (x, y-10),
            cv2.FONT_HERSHEY_COMPLEX_SMALL, 0.5, (255, 0, 0), 1)

        # find_objects
        # mark_detected_objects

    def generate_frames(self):
        capture = cv2.VideoCapture(self.video_source)

        neural_network = cv2.dnn.readNetFromDarknet(self.config, self.weights)

        neural_network.setPreferableBackend(cv2.dnn.DNN_BACKEND_OPENC
        V)
        neural_network.setPreferableTarget(cv2.dnn.DNN_TARGET_CPU)

        YOLO_IMAGE_SIZE = self.yolo_image_size

        while True:
            frame_grabbed, frame = capture.read()
```

```
    if not frame_grabbed:
        break
    else:

        original_width, original_height = frame.shape[1], frame.shape[0]

        # the image into a BLOB [0-1] RGB - BGR
        blob = cv2.dnn.blobFromImage(frame, 1 / 255,
(YOLO_IMAGE_SIZE, YOLO_IMAGE_SIZE), True, crop=False)
        neural_network.setInput(blob)

        layer_names = neural_network.getLayerNames()
        # YOLO network has 3 output layer - note: these indexes are starting
with 1
        output_names = [layer_names[index - 1] for index in
neural_network.getUnconnectedOutLayers()]

        self.detect_count = 0

        outputs = neural_network.forward(output_names)
        predicted_objects, bbox_locations, class_label_ids, conf_values =
self.find_objects(outputs,

self.yolo_image_size,
                                                    self.threshold,
```

```
        self.suppression_threshold)

            self.mark_detected_objects(frame, predicted_objects, bbox_locations,
        class_label_ids, conf_values,
                                original_width / YOLO_IMAGE_SIZE, original_height /
        YOLO_IMAGE_SIZE)

            ret, buffer = cv2.imencode('.jpg', frame)
            frame = buffer.tobytes()


           yield (b'--frame\r\n'
           b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')

    # global declaration
    source = Detect(video_source = './media/swimming_pool1.mp4',
                    classes = ['drowning'],
                    config = './config/yolov3_testing.cfg',
                    weights = './weights/yolov3_training_3000.weights',
                    frame_title = 'YOLO V3 Object Detection',
                    wait_key = 10,
                    threshold = 0.5,
                    suppression_threshold = 0.4,
                    yolo_image_size = 320)
```

```
@app.route('/counter', methods=['POST'])
def counter():
    return jsonify('', render_template('counter.html', dyn_var =
      source.detect_count))


@app.route('/video')
def video():


    frame = source.generate_frames()


    return Response(frame,
    mimetype='multipart/x-mixed-replace; boundary=frame')


@app.route('/detection', methods=["GET", "POST"])
def detection():
    if (session.get("user_token")):
        return render_template('detection.html', dyn_var = source.detect_count)
    return render_template("login_redirect.html", dyn_message = "You need to
      login first!")



# login & registration
@app.route('/validate_login', methods=["GET", "POST"])
def validate_login():
```

```
if request.method == "POST":
    email = request.form.get("user_login_email")
    password = request.form.get("user_login_password")

    session["login_username"] = email
    session["login_password"] = password

    test_login = {
        '_id': email,
        'pword': password
    }

    # test_login = {
    #     '_id': 'veye_admin',
    #     'pword': 'veye_admin'
    # }

    if (test_login['_id'] and test_login['pword']) in db:
        session["user_token"] = db[test_login['_id']]['_rev']

        print(f"username: {session.get('login_username')}; password:
    {session.get('login_password')}")

        return render_template('login_modules/login_success.html',
        dyn_message = "You're in!")
```

```
    return render_template('/login.html', dyn_message = "check your u/name or
      p/word")


@app.route('/logout')
def logout():
    try:
        if session.get("login_username"): session.pop("login_username")
        if session.get("login_password"): session.pop("login_password")
        if session.get("user_taken"): session.pop("user_token")
    except:
        print("something went wrong")

    return redirect("/")


@app.route('/about')
def about():
    return render_template("about.html")


@app.route('/register_intro', methods=["GET", "POST"])
def register_intro():
    return render_template('register_user/register_intro.html')


@app.route('/register_name', methods=["POST", "GET"])
def register_name():
```

```
    # if request.method == "POST":
    # register_user_name = request.form.get("user_name")

    # session["register_user_name"] = register_user_name
    # print(f"name set: {session['register_user_name']}")

    return render_template('register_user/register_name.html')

@app.route('/register_email', methods=["GET", "POST"])
def register_email():
    if request.method == "POST":

    # retrieve user_name from name page
        register_user_name = request.form.get("user_name")

        session["register_user_name"] = register_user_name
        print(f"name set: {session['register_user_name']}")


    return render_template('register_user/register_email.html')

@app.route('/register_password', methods=["GET", "POST"])
def register_password():
    if request.method == "POST":
```

```
        # retrieve user_email from email page
        register_user_email = request.form.get("user_email")

        session["register_user_email"] = register_user_email
        print(f"email set: {session['register_user_email']}")

    return render_template('register_user/register_password.html')

@app.route('/register_phoneNumber', methods=["GET", "POST"])
def register_phoneNumber():
    if request.method == "POST":

        # retrieve user_pass from password page
        register_user_pword = request.form.get("user_pass")

        session["register_user_pword"] = register_user_pword
        print(f"pword set: {session['register_user_pword']}")

    return render_template('register_user/register_phoneNumber.html')

@app.route('/register_outro', methods=["GET", "POST"])
def register_outro():
    if request.method == "POST":
```

```
# retrieve user_phone from phoneNumber page
    register_user_phoneNumber = request.form.get("user_phone")


    session["register_user_phone"] = register_user_phoneNumber
    print(f"phone number set: {session['register_user_phone']}")



register_new_document = {
    '_id': str(session.get("register_user_email")),
    'name': str(session.get("register_user_name")),
    'pword': str(session.get("register_user_pword")),
    'phoneNumber': str(session.get("register_user_phone"))
}

new_document = db.create_document(register_new_document)

if new_document.exists():
    print(register_new_document)
    return render_template('register_user/register_outro.html',
    dyn_message = "You're in!")



return render_template('register_user/register_outro.html',
dyn_message = "Oops! Seems like there was a problem while registering you
  in. Contact Administrator.")
```

```
@app.route('/')
def login():
    return render_template('login.html', dyn_message = "")


if __name__ == "__main__":
    app.config["SESSION_PERMANENT"] = False
    app.config["SESSION_TYPE"] = "filesystem"
    app.secret_key = "veye"
app.run(debug=True)
```

GitHub & Project Demo Link

GitHub Link : https://github.com/IBM-EPBL/IBM-Project-18052-1659678746.git