ECOR 4907

Capstone Project

HydraSmart Final Report

Prepared for:

Professor Safaa Bedawi

Professor Lynn Marshall

By:

Abdal Alkawasmeh, 1011987252

Ahmad Alkawasmeh, 101108706

July 5, 2024

Abstract

Have you ever thought of how clean the water you drink? Do you have to constantly drink bottled water? Are you tired of second guessing if you should drink tap water? water quality has been an immersing issue over the years and water quality systems have become more in demand.

In this project, we implement an automated water quality system which allow users to remotely test the quality of a water sample. Using a few metrics, the system provides the user with water quality readings and determine the drinking water's safety by displaying a message to the user. By using this product, the user will be able to carry and sample any water from anywhere.

This product ensures that the user consumes safe and drinking water and has many applications to its users. However, it is more beneficial for those travelling or who do not have access to filtered water. For example, teachers may wish to check the water quality on a field trip before the children start to consume it.

Acknowledgements

Special thanks to Professor. Bedawi and Professor. Marshall for their continued support and dedication. It has been a pleasant and an amazing experience working with you. Although this project is coming to an end, you have left us with an everlasting experience that reinforced professional practice and discipline throughout the duration of the project. This has not only been instrumental to our success in this project but will also be a major factor in our success throughout our careers.

We would also like to acknowledge Professor. Dansereau and thank him for the valuable feedback he provided us with.

Sincerely,

Ahmad and Abdal Alkawasmeh

Table of Contents

1 Introduction 7
1.1 Background 7
1.2 Motivation 7
1.3 Problem Statement 8
1.4 Proposed Solution 9
1.5 Overview Of Remainder of Report 9
2 The Engineering Project 10
2.1 Health and Safety 10
2.2 Engineering Professionalism 11
2.3 Project Management 12
2.4 Justification of Suitability for Degree Program 12
2.5 Individual Contributions 14
2.5.1 Project contributions 14
2.6 Report contributions 14
2.6.1 Proposal 14
2.6.2 Progress report 15
2.6.3 Final report 15
3 Background 17
3.1 Software 17
3.1.1 FlutterFlow 17
3.1.2 Firebase 18
3.1.3 Arduino IDE 18
3.1.4 Fusion 360 18
3.2 Hardware 19
3.2.1 Temperature Sensor 19
3.2.2 EC Sensor 19
3.2.3 pH Sensor 20
3.2.4 Turbidity Sensor 20
3.2.5 TDS Sensor 21
3.2.6 Arduino Uno R4 Microprocessor 22
4 Requirements 23

5 Design 25

5.1 Frontend Design 25

5.2 Backend Design 27

5.3 Hardware Design 29

6 Generation and Analysis of Alternative Solutions 30

6.1 Microprocessor 30

6.2 Temperature sensor 31

6.3 pH sensor 32

6.4 TDS sensor 33

6.5 Turbidity sensor 34

6.6 EC sensor 34

6.7 Software 36

7 Implementation 38

7.1 Frontend Implementation 38

7.1.1 Page Initialization 38

7.1.2 Dynamic Variables 38

7.1.3 Dynamic Elements and Characteristics 39

7.1.4 Single-Action Buttons 39

7.1.5 Multi-Action Buttons 40

7.2 Backend Implementation 40

7.2.1 Firebase-Application Link 40

7.2.2 Firebase Data Update and Query 41

7.2.3 Data Processing 42

7.2.4 Inter-node Communication 44

7.2.5 User Authorization 45

7.2.6 Sensor Integration 45

7.3 Hardware Implementation 46

8 Testing 47

8.1 Software Testing 47

8.1.1 Mobile Application Testing 47

8.1.2 Arduino Testing 48

8.2 Hardware Testing 49

8.2.1 Temperature Sensor Testing 49

8.2.2 Turbidity Sensor Testing 50
8.2.3 Electrical Conductivity Sensor Testing 53
8.2.4 Total Dissolved Solids Sensor Testing 56
8.2.5 pH Sensor Testing 58
9 Project Milestones 61
10 Conclusions 62
10.1 Conclusion 62
10.2 Recommendations for Future Project Extensions 62
10.3 Reflections 63
References 64
Appendix A: Mobile Application Screens 68
Appendix B: Water Bottle Housing Engineering Drawings 70
Appendix C: Project Progression Gantt Chart 73

1 Introduction

1.1 Background

We go about our everyday lives without giving much thought to the quality of water that we consume daily. Especially as we need to consume a substantial amount of water daily, which is approximately 3.7 litres, to maintain healthy body functions. Thus, it is important to pay close attention to the quality of the water we consume on a daily basis.

Currently, the most widely used technique of determining water quality is by using test strips, which are rectangular pieces of paper that are saturated with a chemical agent, once these paper strips are submerged in a water sample, the chemical agent will react with the water and change colour. This colour indicates the test result, which then the user will need to match to a result chart in order to read the test result. This method of testing the water quality is not accurate and cannot provide exact quantitative results.

Another downside to using the test strips pertains to users with a visual impairment. Due to the inherent nature of the test strips relying on a colour change to convey the test results, users with visual impairments related to colour blindness may face difficulties using the test strips.

1.2 Motivation

The goal of this project was to develop a more suitable solution for monitoring the quality of drinking water and to address the downsides of using traditional test strips.

For users with visual impairments, it can be challenging for them to discern fine differences in shades of a certain colour. Which is the mechanism that test strips rely on to convey the test results to the user. This inherently introduces a barrier for this group of users, and this method of evaluating the water quality may even become unusable for them.

During our research about the existing solutions for monitoring drinking water quality, it became apparent that there are currently no solutions for evaluating the quality of drinking water that are portable, convenient, and simple to use by the end user. This creates a high demand for such a solution as it has a wide range of applications and use cases.

For parents, it would put their minds at ease after becoming aware of the water quality their children will consume, especially during times of uncertainty. Such as when travelling or visiting a new city where they are not sure of how that specific municipality treats their water.

For teachers, if they are accompanying children on a field trip, it would be immensely helpful to be able to check the water quality before everyone consumes it. Moreover, if the water happens to be acidic or contains contaminants, it will not provide the necessary hydration at the least, and in the worst-case scenario it would cause everyone to get sick. Thus, a simple test prior to consumption would allow us to mitigate any risks of consuming water that is of unknown quality or content.

1.3 Problem Statement

There are many drawbacks to using traditional test strips for assessing drinking water quality. The test strips have a low accuracy for the test results as the result is conveyed to the user through a colour change. Aside from the lack of a quantitative test result, the colour change mechanism in itself can become problematic for some users. Moreover, users with visual impairments may not be able to use the test strips as they require acute colour perception to evaluate the test results. Additionally, there is a lack of a suitable solution for assessing the quality of drinking water conveniently by the end users. Therefore, it is essential to develop a product that addresses the disadvantages of traditional water quality testing mechanisms and provides a solution to the public for evaluating the quality of drinking water conveniently.

1.4 Proposed Solution

The solution proposed is a complete system (referred to as HydraSmart) for monitoring the quality of drinking water. HydraSmart comprises multiple sensors conveniently built into a water bottle and a mobile application that acts as an interface between the user and HydraSmart. This proposed solution is convenient to use, portable, and provides highly accurate results. Thus, eliminating the disadvantages of traditional mechanisms for evaluating the quality of drinking water, while simultaneously addressing the lack of such a solution for the general public.

1.5 Overview Of Remainder of Report

This report describes the research, design and implementation process of the HydraSmart system. The introduction introduces the team members, the objectives of this system, background and motivation, problem statements and the proposed solution. This gives the reader a thorough understanding and importance of the overall project.

The Engineering project discusses the health and safety concerns of the system, the professionalism, describes the management process, the justification for choosing the project and the contributions of each member. The Background section focuses on the background of the front end and backend, and the hardware components. The requirements section describes the functional and non-functional requirements of the system. The design section describes the design process of the front-end, back-end and the physical water bottle design. The implementation sections describe the detailed implementation process of the system, discussing the Figma mock-ups, the back-end implementation of the mobile app, and the water bottle implementation of wiring, 3D-model and sensor integration. The theory and techniques discuss the justification of the components in the system. Results and discussion show the results of the system we were able to obtain and discuss future improvements.

2 The Engineering Project

2.1 Health and Safety

The system is designed to ensure the health and safety of the user in order to ensure a safe and positive user experience. The design incorporates three hazardous components, electrical equipment, a rechargeable battery, and glassware. These components are capable of causing bodily harm to the end user if the product is mishandled or misused. However, health risks can be mitigated with appropriate precautions and safety-centred design.

Due to the utilization of electronics and electrical components in this project, there is a risk of electrical shock to the end user if the device is misused. Section 6.4 of the Health and Safety Guide outlines the concept of guarding to prevent access to electrical components. Subsequently, the electrical connections and components were placed in a waterproof compartment that is inaccessible to the user. Moreover, the user will not be able to make contact with the wiring and electrical components of the system, thus mitigating the risk of an electrical shock.

One of the system components will be a rechargeable battery pack, which was used to supply power to the system and allow it to be mobile. The battery pack type is Lithium-Ion, which remains in a stable state so long the battery casing is intact. If the casing becomes punctured or compromised at any time, the battery may catch on fire. This would be catastrophic as it can cause severe burns to the user. As per section 6.6 of the Health and Safety Guide, the use of Shielding is recommended for materials or chemicals that may cause explosions and fires. As such, the strategy to mitigate this risk was to encase the battery pack in metallic plates. This will reinforce the original plastic casing making it significantly harder to penetrate.

Two sensor probes are made of glass and could pose a health risk if they break. As per section 6.7 of the Health and Safety Guide, broken glassware should be handled and disposed of safely. Moreover, protective gloves and eyewear should be used when handling broken glass to minimize the risk of cuts. If a sensor probe breaks, the recommended disposal method is to carefully place the broken glass in a cardboard box, taping the box shut, labelling the box with "Broken Glass", and placing it next to the garbage bin. This will minimize the risks and ensure it is handled safely by anyone that comes into contact with it. Additionally, it is recommended to apply a thick layer of tape to the exterior of glassware to prevent it from dispersing when broken. This was applied in the final product to minimize the health risks to the end user.

2.2 Engineering Professionalism

One component of the 4th year engineering project is to demonstrate how our professional responsibilities were met throughout the performance of the project. Previously, we completed a professional practice course, ECOR 4995. Although it placed a strong emphasis on communication skills, it also taught us about professional responsibilities, health and safety, and principles of sustainable development.

Throughout the duration of our project, we continuously utilized the concepts we were taught in professional practice to communicate, amongst the team and with our supervisors, in an effective and professional manner. To achieve this, we ensured our written communication is kept concise and accurate.

Additionally, during our meetings and discussions, we made sure to always be punctual and be present prior to the meeting time. This helps us conduct ourselves in a professional manner and demonstrates that we highly value our supervisors’ time. However, on one occasion, we experienced unforeseen circumstances that prevented us from attending the meeting on time. We immediately informed our supervisors of this and provided an estimated arrival time, as it otherwise would be inappropriate.

The most notable component of professional practice that pertains to the goals of this project is sustainable development. Sustainable development is concerned with preserving the environment and natural resources, in order to avoid impacting future generations' ability to make use of such resources. HydraSmart will help reduce the amount of plastic waste resulting from disposable water bottles, thus helping to preserve the environment from pollution and non-organic waste.

Another aspect of professional practice is health and safety and safeguarding the public. The concept of public safeguarding pertains to making decisions that prioritize public safety. Throughout this project development, public safety was considered the fundamental concept and main motivation. HydraSmart will help upkeep the public safety and health by ensuring the water is fit for consumption, thus reducing the chance of facing experiencing issues caused by drinking water that was not fit for consumption.

2.3 Project Management

We greatly benefited from creating a Gantt chart [Appendix C: Project Progression Gantt Chart] early on to schedule and organize the execution of the project tasks throughout the duration of the project. This helped us carry out the project objectives in an effective and efficient manner. Evidently, we employed the principles of the Waterfall lifecycle model throughout the duration of this project, where we incrementally completed each part before moving on to the next part. This ensured there are no broken dependencies across the different parts of the projects, therefore increasing our efficiency.

We used JetBrains' YouTrack in order to track our progress and identify any issues that must be resolved in order to continue progressing through the project. YouTrack has a Gantt chart tool conveniently built in, seamlessly enabling task allocation and progress tracking.

2.4 Justification of Suitability for Degree Program

The team members consist of Software Engineering students with a parallel experience in software development, however, each member has a unique interest and background experience.

Ahmad employed his previous knowledge about software architecture and design from SYSC 4120 to outline and analyze the system requirements. The knowledge acquired from COMP 3004 allowed him to design the persistence layer of the mobile application. Utilizing Nielson's Heuristics from SYSC 4130 enabled him to design an intuitive and user-friendly interface for the mobile application. His knowledge in software verification and validation from SYSC 4101 assisted him in verifying correct sensor behavior. The knowledge in design and analysis of automatic control systems from SYSC 4505 allowed him to analyze the sensor response characteristics. Additionally, the Introduction to Engineering (ECOR 1010) course allowed him to acquire the necessary expertise in prototyping and 3D design, which was greatly utilized in this project to design a custom water bottle housing.

Abdal's interest is purely vested in software development. More precisely in user interface design and identifying system requirements. He has taken on multiple personal projects in the past involving website design and development, and he has an extensive experience in working with distributed database systems. Some examples of related work include a sandwich ordering system web user interface which was implemented in SYSC 3010. He has also used his knowledge in SYSC 4101 software verification and validation to use various software testing strategies to verify that the sensors and IDE were operating correctly. Also, the software architecture and design, SYSC 4120, helped him in identifying the functional and non-functional requirements of the system.

Therefore, it is evident the expertise and experience required to complete this project has been greatly reinforced by the undergraduate Software Engineering program. Thus, making this project a perfect fit for Software Engineering students.

2.5 Individual Contributions

Please note that due to the limited number of team members in this project, each member contributed to all the tasks outlined below and they are only categorized by each person based on the member that contributed to completing more than 50% of the mentioned task.

2.5.1 Project contributions

Abdal	**Ahmad**
Software testing	Hardware component testing
Development of the mobile application	Development of the mobile application
Integration of pH, TDS and turbidity sensors	Integration of EC, and Temperature sensors
Calibration of sensors	Back-end Testing
Front-end Design	Back-end Design
Back-end Design	Interfacing portable battery power supply into the Raspberry Pi
Hardware Design	Hardware Design

Table 1: Breakdown of Project Contributions of Each Member

2.6 Report contributions

2.6.1 Proposal

Abdal	**Ahmad**
Chapter 2: Engineering Design	Chapter 1: Introduction
Chapter 5: Hardware Required	Chapter 3: Work Plan
Chapter 6: References	Chapter 4: Project Risks and Mitigation Strategies
	Appendences

Table 2: Breakdown of Proposal Contributions of Each Member

2.6.2 Progress report

Abdal	Ahmad
Abstract	Chapter 2: Professional considerations
Chapter 1: Introduction	Chapters 3.2 and 3.3: Design Functionality Verification, Experiment Design and Setup
Chapter 2.1: Health and safety	Chapter 4: Results and Discussions
Chapters 3.1 and 3.4: Generation of Alternative Design and Assessments, Hardware and Software Tools	Chapter 5: Conclusion

Table 3: Breakdown of Progress Report Contributions of Each Member

2.6.3 Final report

Abdal	Ahmad
1: Introduction	2.0: The Engineering Report
2.4: Justification of Suitability for Degree Program	3.1.1: FlutterFlow 3.1.2: Firebase
3.1.3: Background: Fusion 360 3.1.4: Background: Arduino IDE 3.2: Hardware	5.1: Frontend Design 5.2: Backend Design
4: Requirements	7.1: Frontend Implementation 7.2.1: Firebase-Application Line 7.2.2: Firebase Data Query and Update 7.2.3: Data Processing
5.3: Hardware Design	8.2: Hardware Testing

6.1-6.6: Alternative Solutions Analysis	9.0: Project Milestones
7.2.4-7.2.6: Hardware Implementation	10.0: Conclusions
8.1: Software Testing	References and Appendences

Table 4: Breakdown of Final Report Contributions of Each Member

3 Background

3.1 Software

3.1.1 FlutterFlow

FlutterFlow is a complete application development platform which houses all the necessary tools to develop a mobile application from the ground up. It utilizes the Flutter Software Development Kit (SDK) framework alongside the Dart programming language to develop high-quality mobile applications.

The Flutter SDK is universally compatible with Android and IOS devices, as well as multiple programming languages such as Java, C, and Swift. This provides ultimate flexibility enabling application deployment to multiple operating systems simultaneously.

Dart is a modern object-oriented programming language created by Google. This makes it seamless to integrate other solutions created by Google into the mobile application, such as the Firebase data persistence solution by Google. An added advantage of using FlutterFlow, is the ability to compile the Dart application into JavaScript, enabling live testing and editing in a browser window.

While the graphical interface of the mobile application is developed using Dart and Flutter, it is formatted in eXtensible Markup Language (XML), which is a popular format amongst graphical user interface developers. It is known for its flexibility and universal compatibility with a wide range of operating systems and programming languages.

Additionally, FlutterFlow makes it possible to develop for both Android and IOS devices simultaneously. This is enabled through the Flutter Software Development Kit (SDK).

3.1.2 Firebase

To synchronize and store the sensor measurements, we used Firebase. Which is a data storage solution created by Google. Similarly to Dart, Google provides exceptional and seamless integration between its products. This made Firebase the clear choice when deciding on a database solution.

Firebase is a real-time database, meaning data updates are reflected immediately within the database. This is a highly valued feature in applications that rely on live data to achieve a specific functionality. In our project, it is imperative that the mobile application is able to retrieve live data from the sensors to the user.

3.1.3 Arduino IDE

The main IDE we used to integrate the sensors into the system was Arduino. Arduino is a software application built on C++. It provides users with a user-friendly interface that is easy to navigate and develop code with. It also comes with many built in libraries allowing for advanced functionalities such as sensor readings. It is also important to note the open-source feature, meaning that users can modify code according to their needs or their specific application.

3.1.4 Fusion 360

To Design the 3-D model of the water bottle, we used fusion 360, a 3-D computer aided design program created by Autodesk. This software is well known for its affordability, flexibility and many use of applications. It’s also compatible with many operating systems such as MacOS, windows, and Sonoma.

It also has many functionalities such as generative design, rendering, and simulation. The generative design uses an existing model structure and improves it by reducing the materials while attempting to improve shear and normal resistance. This software is also equipped with fast rendering abilities allowing the visualization of any 3D models. Fusion also uses finite element analysis which evaluates the mechanical properties of an object. This feature is quite critical in decision making between different material manufacturing types, ensuring the best durability of the designed product [1]

3.2 Hardware

3.2.1 Temperature Sensor

The temperature sensor was a crucial part of our system. It helps provide accurate measurements of other sensors as temperature is part of the formula used to calculate the readings. For example, when temperature increases in a solution, the equilibrium is forced to lower the temperature by absorbing more heat. This results in more formation of H+ ions and therefore a lower pH of a solution [2]

Similarly, with electrical conductivity, as the temperature of the water increases, the electrical conductivity also increases. At higher temperatures, water molecules move more quickly, carrying more charged ions. Therefore, this increases the overall conductivity [3].

3.2.2 EC Sensor

The electrical conductivity sensor determines the concentration of ions/dissolved substances in a solution, where high conductivity usually means a higher amount of salts present in the solution and vice versa [4]

This measurement is used in many applications such as cooling towers and wastewater management, and each measurement in an application could mean various things. Even small amounts of salts can result in high conductivity. It’s also possible that the water could be polluted or an existence of a sewage leak. An EC sensor is made up of two surfaces, platinum and graphite, and a voltage waveform is sent from one surface to the other, providing an EC reading [5]

3.2.3 pH Sensor

The pH sensor is used to determine the alkalinity of water in our system, a very important measure in water quality. This measurement is used in a wide variety of applications such as swimming pools and aquaculture. It's important to understand that a pH value that is too low or too high could determine the presence of heavy metals, or other toxic materials [6]. Most fish and aquatic animals have a pH threshold of 6.5-9, so most aquatic systems add buffers to prevent pH fluctuations. In swimming pools, pH of water is usually kept at 7.2-7.8, and if a pH rises too much, this can result in growth of algae and bacteria. Drinking water should have a pH of 6.5-8.5 for safe human consumption. When the pH is too acidic, it can damage pipes and can cause metal built up. When the pH is too basic, it can cause a change in taste of water [8].

3.2.4 Turbidity Sensor

A Turbidity sensor measures the transparency or clearness of water. A high level of turbidity can indicate sediments or pollutants in water [9]. This measurement is very commonly used in water treatment to determine if water has been treated successfully [10] Turbidity is measured in NTU or FTU, where in developed countries drinking water should be kept under 1 NTU and 5 NTU in developing countries. It's also helpful in determining the number of pollutants in water in the case of financial and technical limitations [11].

A Turbidity sensor measures the transparency or clearness of water. A high level of turbidity can indicate sediments or pollutants in water [9]. This measurement is very commonly used in water treatment to determine if water has been treated successfully [10] Turbidity is measured in NTU or FTU, where in developed countries drinking water should be kept under 1 NTU and 5 NTU in developing countries. It's also helpful in determining the number of pollutants in water in the case of financial and technical limitations [11].

3.2.5 TDS Sensor

A TDS sensor measures the total dissolved solids, more specifically, the amount of minerals in water. There are many sources of dissolved solids such as minerals, agritictural runoff and industrial pollution. While some dissolved solids are beneficial, high TDS values can be harmful. Minerals such as calcium and magnesium are usually present in water and are beneficial for human health. However, high levels of TDS can cause health problems those with certain conditions such as high blood pressure, or kidney disease. TDS is calculated by measuring the electrical conductivity of water and converting it to a TDS reading. The amount of conductivity indicates the number of ions in the water, therefore providing a TDS reading. The unit of measurement used to measure the TDS of a solution is ppm (parts per million), indicates the concertation of one part in a million. The world health organization (WHO) recommends a TDS level of less than 500 ppm for drinking water, but this can vary depending on the source and desired use. For an example water used for plants can have a higher TDS [12].

A TDS sensor measures the total dissolved solids, more specifically, the amount of minerals in water. There are many sources of dissolved solids such as minerals, agritictural runoff and industrial pollution. While some dissolved solids are beneficial, high TDS values can be harmful. Minerals such as calcium and magnesium are usually present in water and are beneficial for human health. However, high levels of TDS can cause health problems those with certain conditions such as high blood pressure, or kidney disease. TDS is calculated by measuring the electrical conductivity of water and converting it to a TDS reading. The amount of conductivity indicates the number of ions in the water, therefore providing a TDS reading. The unit of measurement used to measure the TDS of a solution is ppm (parts per million), indicates the concertation of one part in a million. The world health organization (WHO) recommends a TDS level of less than 500 ppm for drinking water, but this can vary depending on the source and desired use. For an example water used for plants can have a higher TDS [12].

3.2.6 Arduino Uno R4 Microprocessor

An Arduino Uno R4 Wi-Fi is a microcontroller used to integrate the sensors, and other hardware. It combines the processing power of the RA4M1 microcontroller from Renesas with the wireless connectivity power of the ESP32-S3 from Espressif. The UNO R4 Wi-Fi has an expanded memory of 32 kB and clock speed of 48 MHz, enabling precises calculations [13].The 12-bit Digital to analog converter provides simplicity, and compatibility to most hardware. This microcontroller is also equipped with Wi-Fi and Bluetooth built in the ESP32-D3 module, enabling users to connect wirelessly. In addition, this board is equipped with a 24 V tolerance allowing for more hardware integration using a single power source. As an error detection mechanism, this board is able to detect runtime errors and provides detailed explanations about the error [14].An Arduino Uno R4 Wi-Fi is a microcontroller used to integrate the sensors, and other hardware. It combines the processing power of the RA4M1 microcontroller from Renesas with the wireless connectivity power of the ESP32-S3 from Espressif. The UNO R4 Wi-Fi has an expanded memory of 32 kB and clock speed of 48 MHz, enabling precises calculations [13].The 12-bit Digital to analog converter provides simplicity, and compatibility to most hardware. This microcontroller is also equipped with Wi-Fi and Bluetooth built in the ESP32-D3 module, enabling users to connect wirelessly. In addition, this board is equipped with a 24 V tolerance allowing for more hardware integration using a single power source. As an error detection mechanism, this board is able to detect runtime errors and provides detailed explanations about the error [14].

4 Requirements

Our system has two main components designed, the mobile application and hardware. The mobile application serves as a user interface while the hardware is responsible for the functionality of the sensors. Therefore, by combining the two components, we developed a portable drinking water quality monitoring system, ensuring the ease of use and convenience.

The system requirements are portioned into functional and non-functional requirements. The functional requirements specify and describe the behaviour of the system, while the non-functional requirements specify how the system should perform.

Functional requirements:

- The system shall be able to measure the pH level of the water with an accuracy of ± 0.1 in a range of 0-14.
- The system shall be able to measure the amount of Total Dissolved Solids (TDS) in the water with an accuracy of ±10 ppm within a range of 0-1000 ppm.
- The system shall be able to measure the amount of Total Suspended Solids (TSS or Turbidity) in the water with an accuracy of ±10 NTU within a range of 0-1000 NTU
- The system shall be able to measure the Electrical Conductivity (EC) of the water with an accuracy of ±100 µS/cm within a range of 0-10,000 µS/cm
- The system shall be able to measure the Temperature of the water with an accuracy of ±1 °C within a range of 0-100 °C.
- The system shall have a graphical user interface which displays the readings of pH, TDS, TSS, EC.

Non-functional requirements:

- The user interface shall be friendly and intuitive to use.

- The system shall be compatible with both iOS and Android devices.
- The system hardware component shall be placed in waterproof and drop proof enclosure.
- The system shall support only 2.4 GHz Wi-Fi connection for maximum compatibility.

5 Design

5.1 Frontend Design

The frontend design started by creating preliminary user interface mock-ups in Figma. During this process, the goal was to design the mobile application pages in a simplified and high-level fashion. By creating the mock-ups, it streamlined the implementation later in Flutter.

We designed a set of pages for the mobile application user interface. These pages [Appendix A: Mobile Application Screens] were designed as follows:

- Welcome page: a landing page that appears upon launching the application to welcome the user and it contains a single button to proceed
- Login page: the page where a user can create an account or sign in
- Home page: provides an overview of the recent measurements plotted on a graph. It also contains two buttons named “Settings” and “Analytics Dashboard”
- Settings page: this is where the user can view and edit their profile, log out, and switch between a dark and a light interface. It also contains a go-back button to navigate back to the home screen.
- Analytics dashboard page: this page displays a summary about the sensor measurements, and it refreshes every 2 minutes automatically. It displays a button named “Measure” the user can press and take an immediate measurement manually. Additionally, pressing on any of the sensor names will navigate to a page that is dedicated to that specific sensor. In the upper left corner, there is a “go-back” button to navigate back to the home screen.
- pH, EC, TDS, and Turbidity pages: each of these pages contain the same elements, the sensor measurement and its units, a short explanation of what the measurement entails, ideal measurement value, and a “go-back” button to navigate back to the dashboard page.

- E. coli page: this page displays the levels of E. coli bacteria present in the water of the beaches and rivers in Ottawa, which are provided daily by the City of Ottawa.

Additionally, we utilized the principles of usability we learned about in our Human-Computer Interaction course we completed previously. These principles, such as Nielsen's Heuristics, can be used as a general guideline during the design of user interfaces to increase usability, make the interface more intuitive to use, and minimize issues users may face while using the application.

The first principle we applied is referred to as "Recognition Rather than Recall". This principle is the most widely applied across user interfaces. It is concerned with minimizing user memory load by enabling them to recognize an action they need to perform in order to complete a certain task. For example, in smartphones, the gear icon on the home screen allows the user to recognize that it refers to the Settings page without having to recall that. This introduces a sense of familiarity within the interface and allows the user to navigate it seamlessly.

In our case, there is two buttons on the home screen of the mobile application, "Settings" and "Analytics Dashboard". We used a gear icon for the "Settings" button, and a trendline icon for the "Analytics Dashboard" button.

The second principle we applied is referred to as "User Control and Freedom". This principle is concerned with providing the user with complete freedom in navigating the interface as they please. Additionally, it includes the ability to go back or undo an action the user performed previously. This makes the user more comfortable with navigating the interface and be confident to explore the interface without hesitation. Throughout the application pages, we ensured to include a go-back button on every page, providing complete freedom to the user in navigating the application pages. We also used a left-arrow icon for this button, repeatedly applying the "Recognition Rather than Recall" principle.

The third principle we applied is referred to as "Aesthetic and Minimalist Design". This principle is about excluding irrelevant and rarely used information from being displayed on an interface. Otherwise, the extra information displayed on an interface competes with the relevant information and diminishes their relative visibility. Throughout the application interface, we ensured that each page displays only the essential information only. For example, the dashboard page displays ratings of sensor measurements initially, and instructs the user to press on a sensor name to view more information about it. Once the user presses on a sensor name, a secondary page is shown, and it displays detailed information dedicated to that sensor.

This is also related to Fitt's Law predictive model, which states that the time to point to an object is a function of the distance from target object and the size of that object. In other words, the user will be able to point to a larger sized object faster than a smaller sized object. Moreover, the more elements displayed on an interface page, the less attention the user is going to assign to each of these elements. Therefore, it is more preferrable to partition information and elements amongst multiple pages rather than having one large page, effectively enabling the user to pay adequate attention to each page or set of elements.

5.2 Backend Design

We started the backend design process by designing a Firebase database schema to account for the data and measurements we need to persist throughout the system. The database needed to include four tables.

The first table, named "users", was needed to store the profile information of the users. Including user id, photo, name, and email. This information is needed by the application in order to create a new user and authorize them to use the application.

The second table, named “sensorData”, was needed for storing and retrieving sensor measurements. It contains 4 rows that store the measurements of the pH, Electrical Conductivity, Turbidity, and Total Dissolved Solids sensors.

The third table, named “liveMeasure”, was needed to facilitate the communication between the mobile application and the Arduino microprocessor. It contains one row that stores a single integer, either a 1 or 0.

The fourth table, named “eColi”, was needed to relay the current levels of E. coli bacteria present in beaches and rivers in Ottawa, and whether there has been rainfall in the past 24 hours. The table contains 4 rows that store the E. coli levels, and one row that stores the rainfall status for the previous 24 hours.

Additionally, we wanted to ensure that the database and the stored data is kept safe. There are two critical categories of data that are of concern. The first is the Firebase API access key, which is used to update and query data from the database, is stored by the mobile application and the Arduino microprocessor. This key was encrypted using SHA-256 and can only be decrypted by the mobile application and the Arduino at runtime. The second critical data category was the actual data stored within the Firebase tables, this data was encrypted using SHA-1 and similarly to the API access key, it can only be decrypted at runtime.

The sensors are controlled using two Arduino scripts. The first script contains several libraries to function the sensors. Another script was included to contain the credentials of the firebase project and Wi-Fi network. Once the sensor script is run, it automatically connects to the specified network and all sensor data is printed in the serial monitor. This data is also sent to the firebase in real-time which is then shown in the mobile application.

5.3 Hardware Design

The physical design of the water bottle is designed using fusion 360, which consists of three main components, the inner cylinder, the outer cylinder and grommet seals. The inner cylinder's purpose is to fit the sensors. The outer cylinder's purpose is a platform to hold the sensors. The grommets are to ensure proper fitment of each sensor. To ensure the measurements was accurate, we carefully and iteratively measured each sensor's dimensions using calipers. Then, different mounts were designed for each sensor on the bottom of the water bottle ensuring stability of the sensors and accessibility to water. We also took into consideration the dimensions of the breadboard, rechargeable battery and the Arduino board as they are required to power the sensors. The turbidity sensor is short in length, and therefore was placed on the lowermost front position on the inner cylinder.

6 Generation and Analysis of Alternative Solutions

6.1 Microprocessor

When choosing a microprocessor for the system, we considered two options: the Arduino uno and the Raspberry Pi. The Arduino Uno supports both analog and digital inputs, making it compatible with various sensors. It's simpler to program, ideal for development. Additionally, it is more energy efficient making it suitable applications equipped with batteries. However, the Arduino Uno lacks an operating system, which limits the potential of complex projects, and is more suitable for simpler ones [15].

The Raspeberry Pi offers compatibility for digital input and runs a full operating system, providing complex abilities and software support. It's better suited for projects requiring more complexity. However, this option is more expensive, lacks compatibility for analog input and consumes more power.

Due to our system requirements, the Arduino was found to be more feasible. Our sensors requiring both analog and digital inputs, and the Arduino's lower power consumption played important factors in this decision. Although the RaspebrryPi offered more flexibility, it's higher cost and power usage made it less suitable for our needs [16].

Microprocessor/microcontroller	**Advantages**	**Disadvantages**
Arduino Uno R4 Wi-fi	• Includes analog and digital pins • Simple and easy to use • Low power consumption	• Does not use an operating system • Suitable for only small applications • 8-bit CPU architecture
Raspberry Pi	• Has 40 digital pins	• Does not support analog input

	• Suitable for large and complex applications • Runs on an operating system	• More expensive • High power consumption

Table 5: Trade-off Analysis of Microprocessor/Microcontroller

6.2 Temperature sensor

When choosing a temperature sensor, there were several options available. However, the two most competitive were the DFRobot DS18B20 and DFRobot TMP100. The DS18B20 model has it's benefits like being waterproof, smaller in size, and a more affordable option. However, it has a slightly less accuracy, measures up to 125 and takes a few minutes to stabilize [17] [18]. The TMP100 model, on the other hand, offers higher accuracy, stabilizes very quickly, and is compatible with both 5V and 3.3V systems, including the Arduino DUO and UNO models. Given our goal to keep costs low, and the need to measure temperatures of water solutions, we chose the DS18B20 model [19] [20].

Sensor	**Advantages**	**Disadvantages**
DFRobot DS18B20	• Waterproof • Compact size • Affordable	• Less accurate • Measures temperature only up to 125 • Takes time to stabilize
DFRobot TMP100	• High accuracy • Quick stabilization • Compatible with 5V and 3.3V	• Not waterproof • More expensive • Requires soldering

Table 6: Trade-off Analysis of Temperature Sensor

6.3 pH sensor

To choose the optimal pH sensor, we evaluated two popular options: the DFRobot model and the Grove model. The DFRobot model can measure a wider temperature range, from 0 to 60, while the Grove model is limited to 55. Additionally, the DFRobot model has a faster response rate of 1 minute and includes a power LED indicator [21].

The Grove model offers several benefits, such as a smaller probe size, higher accuracy rate, and the convenience of requiring calibration only once [22] . However, it has a slower response rate of more than a minute, shorter probe life of 6 months and a longer cable length, which is not optimal for minimizing materials [23] [24]

Despite the Grove model's higher accuracy and smaller probe size, the DFRobot offers a wide temerpature range, and a faster response rate. Therefore, given our needs and requirements, the DFRobot pH meter was found more suitable.

Sensor	**Advantages**	**Disadvantages**
DFRobot pH meter	• Measures a wider temperature range • Faster response time • Includes power LED indicator	• Less accurate • Larger probe size • Requires frequent calibration
Grove pH sensor	• Smaller probe size • Higher accuracy • Only requires one-time calibration	• Slower response time • Shorter probe lifespan • Longer cable length

Table 7: Trade-off Analysis of pH Sensor

6.4 TDS sensor

When deciding on the TDS sensor, we compared two competitive products: the DFRobot Gravity TDS sensor and Grove TDS sensor. The DFRobot sensor has a lower error rate, is waterproof, and can operate with both 3.3V and 5V of power. However, it has a larger wire size, is only compatible with Arduino and requires frequent recalibration [25]. In contrast, the Grove sensor has a shorter wire length, does not require recalibration and is compatible with both Raspberry Pi and Arduino, however it has its limitations such as not being useable in polluted water, not functioning in temperatures over 70, and lacking waterproof capabilities [26].

Given our comparison, the DFRobot Gravity TDS sensor was found to be more suitable for our system. Despite it's larger wire size and need for frequent calibration, it's waterproof capabilities and lower error rate makes it the more optimal choice for our needs.

Sensor	**Advantages**	**Disadvantages**
DFRobot Gravity TDS analog sensor	• High accuracy • Waterproof probe • Compatible with 3.3V and 5V	• Longer wire size • Not compatible with Raspberry pi • Requires frequent calibration
Grove TDS sensor	• Shorter wire size • No recalibration required • Compatible with the Raspberry Pi and Arduino	• Not suitable for water temperatures over 70 • Cannot be used in polluted water • Non-waterproof probe

Table 8: Trade-off Analysis of TDS sensor

6.5 Turbidity sensor

When choosing a turbidity sensor, we considered two options: the DFRobot and the Grove sensor. The DFRobot model is lighter in weight, more affordable, and has the ability to adjust the threshold. However, it has its limitations, such as only being compatible with the Arduino, a slower response rate and requiring calibration [27]. The Grove sensor is more beneficial in terms of power consumption, size, and output support. However, it is more expensive, slightly heavier, and requires calibration [28].

Despite the slower response time, and requiring calibration, the DFRobot model offers significant advantage in cost and weight. Therefore, based on our analysis, it was found to be a more practical option for our project.

Sensor	**Advantages**	**Disadvantages**
DFRobot Gravity Turbidity sensor	• Lighter weight • More affordable • Adjustable threshold	• Only compatible with Arduino • Slower response time • Requires calibration
Grove Turbidity sensor	• Less power consumption • Smaller size • Supports both digital and analog output	• More expensive • Slightly heavier • Requires calibration

Table 9: Trade-off Analysis of Turbidity Sensor

6.6 EC sensor

When evaluating the DFRobot Gravity EC sensor and the Grove EC sensor, several factors were considered including probe life, temperature range, compatibility, cost, and power consumption.

The DFRobot Gravity EC sensor offers a longer probe life, making it suitable for extended use and is also more cost-effective. Also, it has a greater EC detection range which provides more application flexibility. However, it has it's limitations such as operating in a smaller temperature range, and consumes more power [29]. The Grove EC sensor has the ability to operate in a larger temperature range, providing more flexibility. It's also compatible with both the Arduino and Raspberry Pi. Additionally, it consumes less power. However, some limitations include a higher cost, a shorter probe life, and a low electrical conductivity detection range [30].

Since cost and an EC detection range are very important to this project, and we are exclusively working with the Arduino, the DFRobot is a more suitable choice.

Sensor	**Advantages**	**Disadvantages**
DFRobot Gravity EC sensor	• Longer probe life • Cost-effective • Greater EC detection range	• Narrower temperature range • Only compatible with Arduino • Higher power consumption
Grove EC sensor	• Wider temperature range • Compatible with Arduino and RaspberryPi • Lower power consumption	• More expensive • Shorter probe life • Lower EC detection range

Table 10: Trade-off Analysis of EC Sensor

6.7 Software

For choosing a mobile application development platform, Flutter and React Native were evaluated on several factors such as performance, useability, and suitability.

Flutter has the ability to use a single across multiple platforms such as iOS and Android, ensuring portability. It’s drag and drop widgets allows for a highly responsive user interface, while direct compilation to native ARM provides excellent performance. However, Flutter faces limitations like larger application size, and a smaller library ecosystem and community support due to it’s recent invention.

React Native utilizes JavaScript, a widely used language with many libraries and tools. It also uses native components (UI) to create the app’s user interface, providing a smoother and quicker performance. React native applications usually require a lower minimum SDK version, making it lighter. However, it’s dependency on native modules can impact performance and It’s development process as it requires knowledge of react and native UI components. Additionally, it may experience occasional UI inconsistences [31].

In conclusion, Flutter’s various features, including the drag and drop capabilities and excellent performance, make it a better platform for efficient mobile application development.

Software	Advantages	Disadvantages
Flutter	• Single codebase for iOS and Android • Drag and drop widgets • Direct compilation to native ARM	Larger application size Smaller library ecosystem Limited community support
React	• Uses native component, resulting in a smoother performance • Requires minimal SDK • Extensive libraries and tools	• Dependency on native modules, affecting the performance • Requires knowledge of react • UI inconsistencies

Table 11: Trade-off Analysis Software Development Platform

7 Implementation

7.1 Frontend Implementation

7.1.1 Page Initialization

Implementing the designed user interface in Flutter starts by initializing the pages mentioned in section 5.1. To initialize a page, you create a new blank page and enter a name for that page. After which, you insert the desired elements of the page, such as text, buttons, and icons. All elements are configured as static elements initially, and then can be set to have dynamic characteristics at a later stage. It would be otherwise extremely difficult to attempt to assign dynamic behavior to element at an early stage of the development, which would introduce unnecessary complexity and high coupling between the different pages and their corresponding elements, thus making the application susceptible to bugs and errors.

Once all the pages have been initialized with their elements, we now started to configure these elements according to our design. We started with configuring the text elements to ensure we have the appropriate text formatting prior to configuring buttons. Otherwise, it is likely that we would have to reformat the pages' layouts due to overcrowding.

7.1.2 Dynamic Variables

For the dynamically updated elements, such as the sensor measurements and E. coli levels, they were configured using the "assignFromVariable" function. This function allowed us to dynamically set a value of an element by querying a database table, caching the data returned by the query, then displaying this data within an assigned field in the user interface. The database query needed to be performed at the page parent level in order to work correctly. The page parent level is the layer at which the topmost element is located, which is the first element that was placed on the page.

7.1.3 Dynamic Elements and Characteristics

For nested dynamic element characteristics that require query data processing prior to executing a specific function, such as setting a text highlight color in the dashboard and E. coli pages based on the measurement value, we utilized the “assignFromVariable” function used previously. However, in this case we used it along with an if conditional block. In particular, the function compares a specific value retrieved from the database to a set of possible values. Once the actual value is matched to one of the expected values, the function returns a parameter containing a hexadecimal color code object to the element configurator, which in turn sets the highlight color of a text object on the page. This approach was also used in setting the “Is it consumable?” and “Overall rating” elements on the dashboard page by passing the raw sensor data to a function that calculates the appropriate value to display on the interface.

7.1.4 Single-Action Buttons

To implement the buttons’ navigation features and actions, we utilized the “setAction” and “navigateTo” functions. These functions allow us to define how the interface reacts to the users’ button pressing actions. For navigation buttons, we used the “navigateTo” function to set the page that the interface should transition to upon the button press. As such, the interface navigates back to the previous page when pressing on the left arrow icon in the upper left corner from any page. For buttons that navigate forward in the interface, we set the “navigateTo” function parameter as the name of the desired page to transition to next, such as the “Let’s get started” button on the welcome page transitions to the log in page, “Settings” and “Analytics Dashboard” buttons on the home page, “Sign in” button on the login page, and each of the sensor names on the dashboard page.

7.1.5 Multi-Action Buttons

As for buttons with non-navigation related functions, such as the “Measure” button on the dashboard page, the “Log out” button on the settings page and the “Sign in” button on the login page, we utilized the “customAction” function. This function allows us to define complex and multi-action button functions.

The “Measure” button function is to request an instant measurement from the sensors in real-time, this is achieved by requesting a data update from the Arduino microprocessor. The Arduino then updates the Firebase “sensorData” table with the new measurements and closes the request, informing the application that the new measurements are now available. The mobile application then overwrites the previously cached sensor readings with the new data and automatically updates all the elements related to these readings.

For the “Sign in” and “Log out” buttons, we utilized the global Flutter parameter named “isUserAuth” to keep track of whether the user is currently signed in and signed out. When the user presses “Sign in” from the log in page, the “isUserAuth” parameter is set to true using a “customAction” function that checks if the user authorization was completed successfully. In case the signed in user presses on the “Log out” button in the settings page, a secondary “customAction” function is called to change the value of the “isUserAuth” parameter to false, this triggers another function named “navigateTo” in order to transition the interface back to the log in page.

7.2 Backend Implementation

7.2.1 Firebase-Application Link

The backend implementation started by connecting the designed Firebase database [reference section 5.1] to Flutter and the Arduino microprocessor. Firebase is able to generate a configuration file with all the variables that are needed by an application in order to access the database and its content. The configuration file was obtained from the Firebase dashboard then was made accessible to the Flutter project and the Arduino scripts by saving a copy of it to the parent directory of the Flutter project and Arduino scripts.

The configuration file is formatted in Json to enable the application to fetch the specific variables by performing a search within the configuration file using the inherent parent-child relationship of data contained within a Json file. As such, the environment variables were then instantiated with the corresponding values from the Json configuration file.

Flutter automatically recognizes the Firebase variables defined and configures the “users” table schema with the appropriate number of columns and their corresponding names. Additionally, Flutter deploys a set of rules for the Firebase database in order for the encryption mechanism to function correctly. It is critical that the “users” table is not modified manually under any circumstances to ensure the integrity of the link between Flutter and Firebase.

There is a Firebase SDK available for the Arduino microprocessor that enables it to communicate with the database and it needs to be installed prior to performing any Firebase operations from the Arduino. The Firebase SDK named “FirebaseAdmin” was installed at the same location as the Firebase configuration file, which is the parent directory of the Arduino.

7.2.2 Firebase Data Update and Query

An integral function of the mobile application is making the sensor measurement available to the user. To achieve this, the Arduino relays the measurements produced by the sensors to the Firebase database. Firebase acts as an intermediary host between the mobile application and the Arduino, by storing the sensor data, enabling the mobile application to query and fetch the needed data. Additionally, it enables automatic data refresh in the application when the Arduino updates the Firebase measurements with more recent data.

The Firebase configuration performed by Flutter previously allows it to detect when a data update occurs. As such, it can automatically fetch the updated data and overwrite the outdated data cached at an earlier time.

The Arduino microprocessor was programmed to record the sensors' measurements periodically, specifically, every 20 seconds. Once this periodic function is triggered, the sensor measurements are read from the sensors, then saved locally in a temporary variable. Once the sensor measurements have been collected, they are inserted into the "sensorData" Firebase table using the "db.set()" function provided by the Firebase SDK. After which, the mobile application refreshes its cached data and updates the interface to reflect the updated measurements.

For the E. coli measurements provided by the City of Ottawa, the Arduino fetches the Excel file from the city website every 24 hours. The latest measurements are then extracted and inserted into the "eColi" database table. Since the measurements are from the previous day, we also wanted to account for rainfall as it causes the E. coli levels to increase. This was achieved by utilizing an API provided by Open Weather. To fetch the rainfall forecast, we passed the latitude and longitude of Ottawa within an API call, the API then sends back a Json file with all the weather data for Ottawa. Then we extract the rainfall amount from the Json file by filtering the data using "daily.rainfall", returning the rainfall amount in millimeters. If the value is larger than zero, then we set the rainfall row of the "eColi" table to true, otherwise, it is set to false.

7.2.3 Data Processing

Some dynamic elements of the application rely on the processing of raw data in the backend in order for them to function correctly. The dynamic elements that require this include the "Overall rating", "Is it consumable?", and the measurements' status on the dashboard page.

The “Overall rating” element provides an overall score out of 5 for the water quality based on the EC, pH, TSS, and TDS measurements. The mechanism calculates the score by assigning a score for each of the quality metrics, then sums the 4 individual scores to obtain an overall score out of 5 points. Based on the measurement of each quality metric, a score of 0, 0.625, or 1.25 will be assigned to that metric.

Additionally, the measurement status and “Is it consumable?” elements utilize the overall score calculation function. For the measurement status, a status of “Poor”, “Fair”, or “Good” is assigned to each metric based on whether the individual metric score was 0, 0.625, or 1.25 (respectively). Also, each status is highlighted in the corresponding color for a more visually pleasing user experience. Where a “Good”, “Fair”, or “Poor” status is highlighted in green, purple, or orange (respectively).

For the “Is it consumable?” element, it provides a general decision about whether the water can be consumed by displaying either a “Yes” or “No” answer. It is based on the previously calculated overall score where if the total score is less than 1.8, a “No” answer is displayed, and if the sum is equal to or more than 1.8, then a “Yes” answer is displayed. However, there is one exception to this mechanism, if the pH was assigned a “Poor” status, then an answer of “No” is automatically displayed. Also, a disclaimer was added to the mobile application informing the user that the system is unable to detect pathogens and micro-organisms in the water, in order to avoid any ambiguity with the “Is it consumable?” element of the dashboard page. The table below outlines the range of values for status and score assignment of each quality metric.

Metric / Rating	Good (1.25 points)	Fair (0.625 points)	Poor (0 points)
EC	value < 0.3	0.3 <= value <= 0.8	value > 0.8
pH	7 <= value < 9	9 <= value <= 10.2	value > 10.2 or value < 7
Turbidity	value < 500	500 <= value < 1500	value >= 1500
TDS	value < 150	150 <= value <= 400	value > 400

Table 12: Score and Rating Assessment Mechanism Values

For the E. coli status on the dashboard page, a function checks if all the levels are below the threshold of 200 and whether there has been any rainfall in the past 24 hours. If all the values are below the threshold and there has been no rainfall, then the status is set to "Good". If the E. coli levels are all below the threshold and there has been rainfall, then the status is set to "Poor". If at least one of the E. coli levels is below the threshold and there has been no rainfall, then the status is set to "Fair". On the E. coli page that displays the measurement of each beach or river, each measurement is highlighted in Green or Orange based on whether that specific measurement is below or above the threshold (respectively), except for when there has been rainfall in the past 24 hour, then the measurements are all highlighted in Orange.

7.2.4 Inter-node Communication

In order to implement the real-time measurement feature of the mobile application, a communication link was needed to connect the mobile application and the Arduino microprocessor. This was achieved by utilizing the real-time functionality of Firebase. A third Firebase database named "liveMeasure" containing one field was created, and the field was initialized with the integer 0. When the user requests a new measurement by pressing the "Measure" button on the dashboard page, the integer stored in "liveMeasure" is overwritten with 1. The Arduino script continuously checks if there has been a data update, once this is detected by observing the change of the stored value from 0 to 1, it records the sensor data and inserts them into the "sensorData" table and overwrites the integer stored in "liveMeasure" with

0. Simultaneously, the mobile application detects the data update in both the “sensorData” and “liveMeasure” tables, after which, it overwrites the cached sensor data with the updated data, which are then displayed to the user.

7.2.5 User Authorization

The user is provided with multiple methods to sign in to the application. They have the option to use a conventional email and password configuration, Google sign in, or Apple sign in. This improves the user experience by providing flexibility and convenience.

If the user elects to use a conventional email and password to sign in, they will be required to use a minimum password length of 8 characters, include one upper case and one lower case letter, and include one number. Their sign in information is then encrypted and stored in the “users” database.

If the user elects to use Google or Apple sign in, they will be directed to the appropriate website to continue the sign in process. Once they complete the sign in process with Google or Apple, the user information (excluding their password) is passed back to the application, which creates a new user account using the information passed, then encrypts and stores the profile information in the database. The next time they attempt to sign in to the application, the application recognizes their existing profile, and their information is retrieved.

7.2.6 Sensor Integration

To integrate the sensors once connected to the Arduino board, we installed several libraries such as Arduino DFRobot and Firebase which are required to run the script and obtain readings. We then defined the pins to ensure each reading was properly acquired. We then wrote the complete script to read the values from each sensor using a void() and loop(), which is a requirement of the Arduino IDE. We also included the firebase credentials allowing for data storage. Once the values are obtained, they are stored in Firebase database in real-time.

7.3 Hardware Implementation

To implement the hardware component, several sensors were connected to the Arduino Uno through a breadboard, which controls them using a sketch in the Arduino IDE. Each sensor is connected to three pins: an analog input, ground and power. The Arduino Uno has six analog pins (A0 to A6), two 5V pins, two ground pins, a 3.3V pins, and fourteen digital pins (D0-D13).

The pH sensor was connected to analog pin A5, the EC sensor to analog pin A2, the turbidity sensor to analog pin A0, and the TDS sensor to analog pin A3. The temperature sensor, which has a digital input was connected to digital pin D5. To power the sensors, the Arduino board is connected to a rechargeable battery, providing ease of use and minimal hardware requirements.

The pH and EC sensors must be calibrated for accurate readings using calibration solutions provided by the manufacturer. The pH sensor is entered into calibration mode via the serial monitor command "enterph" and then placed in a 7 pH or 4 pH calibration solution. Similarly, the EC sensor is entered into calibration mode using the "enterec" command. The system will automatically the solution and adjust to the correct value.

Additionally, the water bottle is made using ABS material. This is due to its known durability, and strength. It is also important to note that it is food-safe and used products that come in contact with liquids such as water, making it an excellent choice for this application [32].

8 Testing

Given the sensitive nature of this project, where improper or inaccurate system behavior could pose health risks to the user, extensive testing was conducted to verify that the system components are functioning correctly. Testing was partitioned into two parts, software testing and hardware testing. The software portion covers the mobile application functions, Arduino scripts, and the Firebase database. The hardware part of testing covers the sensors and their accuracy.

8.1 Software Testing

8.1.1 Mobile Application Testing

We thoroughly tested the mobile application thoroughly in order to ensure all the elements behave as expected. Due to the rapid testing feature of Flutter, we were able to compile the mobile application and test it on a virtual device within a web browser. This also allowed us to check the performance of the mobile application throughout the testing to make sure the app stayed responsive throughout the tests performed.

We first tested all the navigation buttons to verify the correct transitions take place while navigating the different pages. Then we tested the behavior of complex buttons, which trigger the execution of functions in the backend, to ensure the functions are accessible within the current app state and return correct parameters or values back to the function caller.

Once the mobile application was confirmed to be functioning as expected, we tested the mobile application in integration with the Firebase database. We started by manually inserting test values into the database tables, then simulating a user action that would cause the application to fetch the data from the database and process it accordingly. In particular, we were testing the ability of the application to identify the correct set of data in the database, obtain that data, and then process it correctly. During this step, we identified a bug that causes the application to become unresponsive and crash. The issue was related to a while loop that filters the data fetched from the database, where it would keep iterating infinitely when none of the conditions within it are satisfied. To address this issue, we added an additional condition that forces the while loop to return an error code when none of the conditions have been satisfied.

After the mobile application and database were tested, we introduced the Arduino into the testing procedures we previously completed with the exception that all the functions' logic of the Arduino scripts were stubbed in order to isolate the testing only to the communication between the system components. At this stage, we tested the ability of the Arduino to access the database and communicate with the mobile application. Once we were sure that all the components were functional, we moved on to unit testing the Arduino scripts to ensure all the function logic is correct and bug free.

8.1.2 Arduino Testing

Testing the Arduino script is important to ensure the data is displayed and sent to the firebase successfully. A popular software testing library called unitTest, is available in the Arduino IDE as ArduinoTest. This library was used to test to verify matching values using "assertEquals()". To verify the data is sent and retrieved correctly, we implemented a function "TestFirebase()" inside the test script "Testarduino". Inside the function, instances of "read[sensor_name]()" are initialized to retrieve the sensor readings and then pushed to firebase using the built-in function "pushFloat()". The instances of each reading are then compared against the firebase values using the "assertEquals()". Similarly, for testing the accuracy of displayed output, sensor readings are compared with values printed in the serial monitor to verify the correct output. This was done by implementing a function "TestDisplayOuput()", which calls and asserts the instances of the sensors with the "Serial.print()" function.

8.2 Hardware Testing

The hardware testing portion proved to be the most critical as the sensors are the source of data in our system and we must verify both the accuracy and correctness of measurements they produce. We needed to design a set of tests that allow us to compare the actual measurements to the expected measurements of that sensor. This was achieved by formulating multiple solutions of known concentrations, which we can then measure with the appropriate sensor, and compare the results. This testing procedure also helped us in confirming the validity of the calibration performed by the manufacturer. We also used the test results to analyze the transient response characteristics of the sensors. Most importantly, we need to ensure the standard solutions used in testing the sensors were sourced from a reputable source in order to be confident in our results. As such, the standard solutions were obtained from an organization named “Hach”, which specializes in water quality testing and is considered an industry leader due to their proven track record and reputation.

8.2.1 Temperature Sensor Testing

The first sensor we tested is the temperature sensor. This is because the temperature is needed for accurately measuring pH, EC, and TDS. Thus, it is essential to ensure the temperature sensor is functioning correctly before moving on to the other sensors. In order to test the temperature sensor, we boiled distilled water in a clean kettle then we measured the temperature of the water using the temperature sensor. We used a sampling rate of 100Hz and collected measurements for 3 seconds. We then plotted the collected data as shown in the figure below:

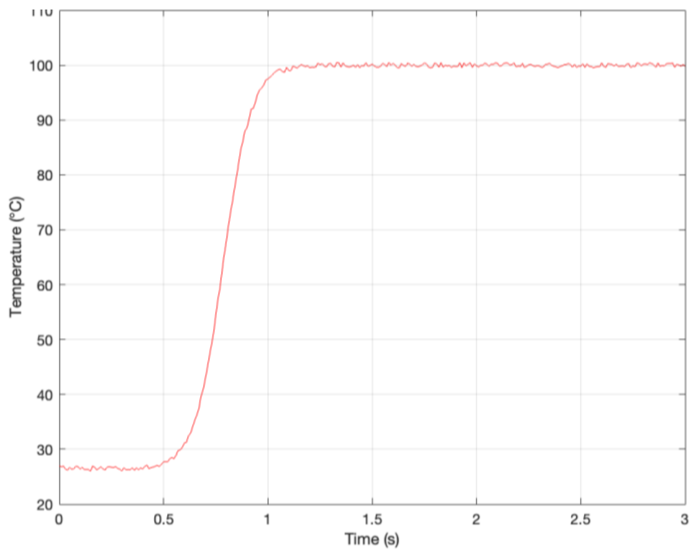


Figure 1: Temperature Sensor Response for Boiled Distilled Water (100 °C)

The temperature sensor settling time was 0.8 seconds, which is similar to the manufacturer specification of 0.75 seconds. Additionally, we can see that it is functioning correctly as the steady state output value is approximately 99.7 °C. Therefore, we can conclude that the temperature sensor is functioning correctly, and it passed the test.

8.2.2 Turbidity Sensor Testing

In order to test the turbidity sensor, we purchased a 4000 NTU standard solution which we used to formulate three solutions of 1000 NTU, 500 NTU, and 100 NTU. In order to dilute the standard solution down to the desired concentrations, we used the following formula to calculate the needed volume of the standard solution:

$$C1V1 = C2V2$$

Where C1 is the concentration of the standard solution, V1 is the volume of standard solution needed, C2 is the concertation of the diluted solution, and V2 is the volume of the diluted solution. Since we only need to calculate the volume of standard solution we need to use, we can rearrange the equation as follows:

$$V1 = \frac{C2V2}{C1}$$

We then used the above formula to calculate the required volume of standard solution for each diluted solution as per the following table:

Volume of standard solution (V1)	**Concentration of diluted solution (C2)**	**Volume of diluted solution (V2)**	**Concentration of standard solution (C1)**
25 mL	1000 NTU	100 mL	4000 NTU
12.5 mL	500 NTU	100 mL	4000 NTU
2.5 mL	100 NTU	100 mL	4000 NTU

Table 13: Turbidity Sensor Test Solutions Dilution Calculations

We then formulated each of the three solutions by mixing the specified volume of standard solution into distilled water, please note that the volume of distilled water to use for each diluted solution is the difference between the volume of diluted solution and the volume of standard solution as follows:

$$volume\ of\ distilled\ water = V2 - V1$$

Once the diluted solutions were formulated, we tested the turbidity sensor by measuring the turbidity of each of the diluted solutions and made sure to thoroughly rinse the sensor probe with distilled water between each test. Then we plotted the collected data to facilitate the analysis of results. The following figures show the collected measurement data for each of the diluted solutions:

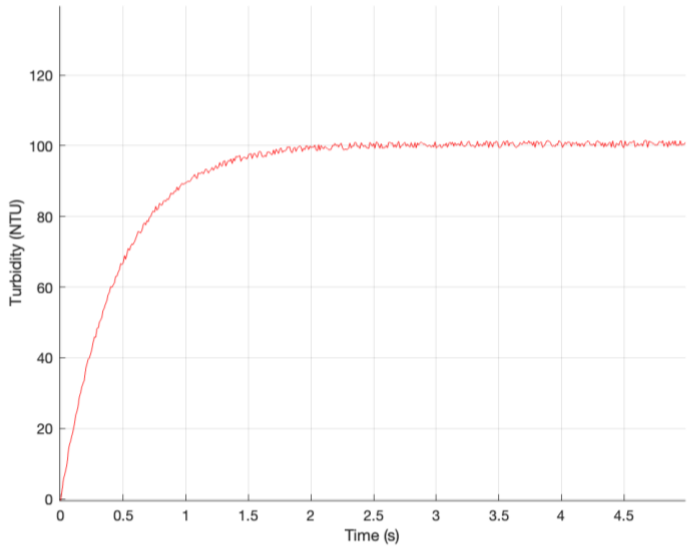


Figure 2: Turbidity Sensor Response for 100 NTU Test Solution

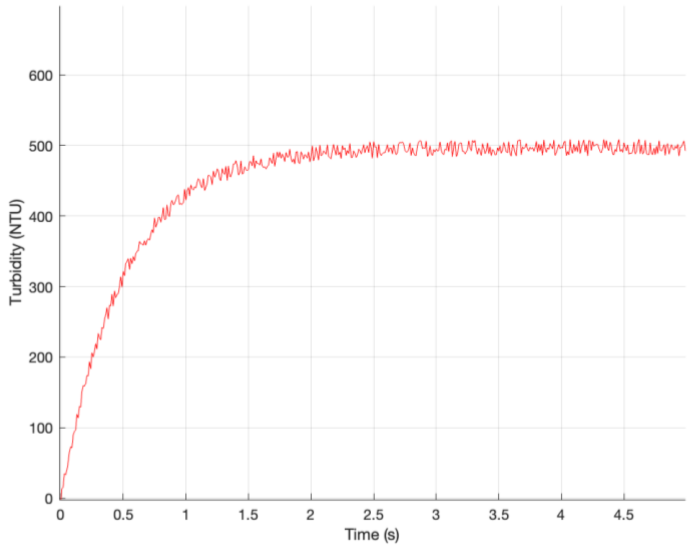


Figure 3: Turbidity Sensor Response for 500 NTU Test Solution

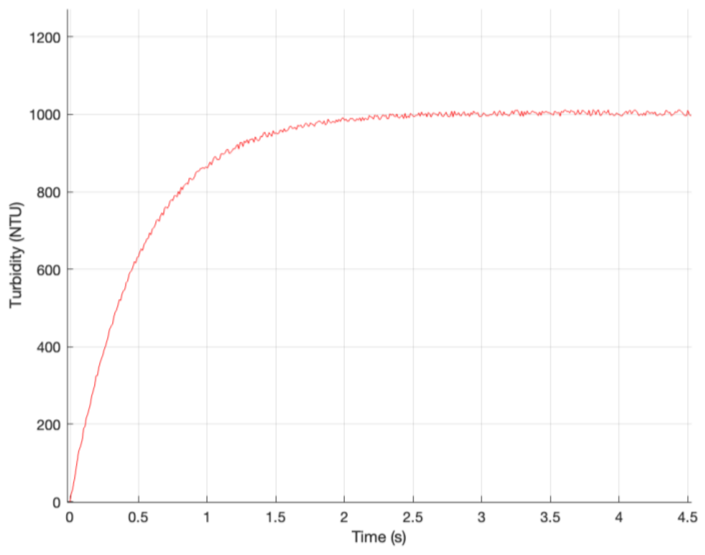


Figure 4: Turbidity Sensor Response for 1000 NTU Test Solution

The manufacturer specifications state that the settling time of the turbidity sensor is 2 seconds. In our testing, we found the settling time to be 1.9 seconds on average, which indicates the sensors' performance is better than what the manufacturer anticipated. The measurements were consistently accurate across the three solutions we tested, with an accuracy of ± 3 NTU, indicating they are functioning correctly. Therefore, we can conclude that the turbidity sensor has passed the tests.

8.2.3 Electrical Conductivity Sensor Testing

In order to test the EC sensor, we purchased a standard solution with a conductivity of 3960 µs/cm in order to prepare 3 test solutions. This was achieved by diluting the standard solution accordingly using the same method from section 8.2.2 Turbidity Sensor Testing. The test solutions were formulated according to the following table:

Volume of standard solution (V1)	Concentration of diluted solution (C2)	Volume of diluted solution (V2)	Concentration of standard solution (C1)
46.7 mL	1850 µs/cm	100 mL	3960 µs/cm
24 mL	920 µs/cm	100 mL	3960 µs/cm
8.8 mL	350 µs/cm	100 mL	3960 µs/cm

Table 14: EC Sensor Test Solutions Dilution Calculations

The EC sensor was then used to measure each of the test solutions while making sure to thoroughly rinse the sensor probe between the tests. For each test, the sensor data output was logged for 5 minutes to ensure the sensor measurement had ample time to settle, then we plotted the data to analyze it. The following figures show the sensor response for each of the test solutions:

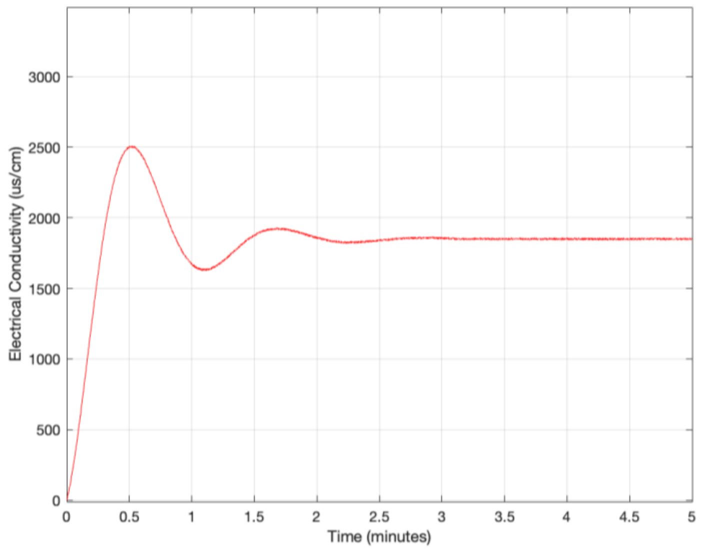


Figure 5: EC Sensor Response for 1850 µs/cm Test Solution

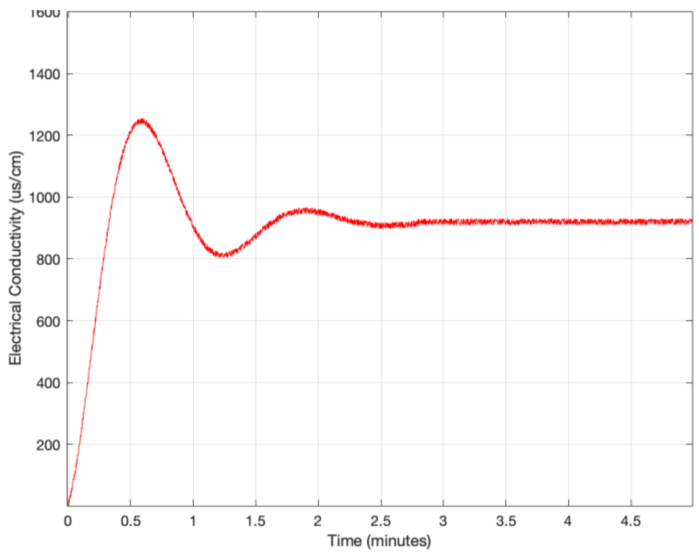


Figure 6: EC Sensor Response for 920 µs/cm Test Solution

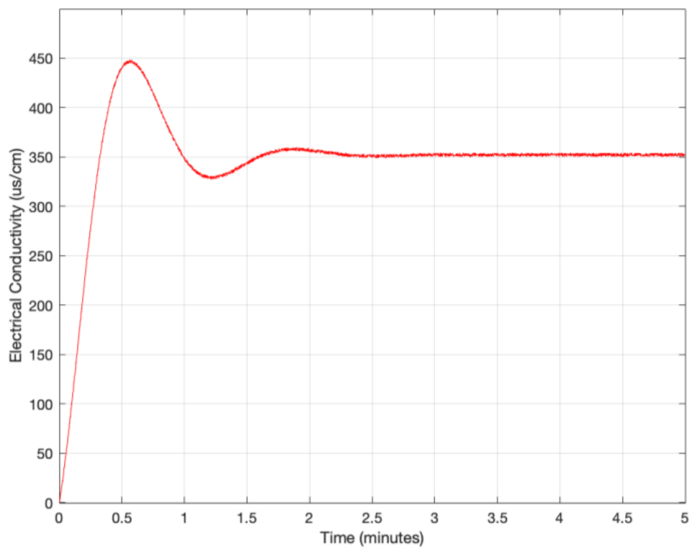


Figure 7: EC Sensor Response for 350 µs/cm Test Solution

We observed that the readings fluctuate initially when the sensor is placed in each of the solutions before stabilizing and providing a constant measurement. Although the manufacturer does not provide a settling time specification, we found that on average it takes 3 minutes for the readings to stabilize. The steady state output values for each of solutions tested were within ± 5 µs/cm of the expected value, indicating the sensor is functioning as expected. Therefore, we can conclude the EC sensor has successfully passed the tests.

8.2.4 Total Dissolved Solids Sensor Testing

In order to test the TDS sensor, we used the same standard solution from the EC sensor tests as it is also rated with a TDS content of 3000 ppm. Similarly, we prepared three solutions of varying TDS content in order to perform the tests for the TDS sensor. The test solutions were formulated as per the following table:

Volume of standard solution (V1)	**Concentration of diluted solution (C2)**	**Volume of diluted solution (V2)**	**Concentration of standard solution (C1)**
30 mL	900 ppm	100 mL	3000 ppm
13.7 mL	410 ppm	100 mL	3000 ppm
4 mL	120 ppm	100 mL	3000 ppm

Table 15: TDS Sensor Test Solutions Dilution Calculations

Once the test solutions were prepared, we used the TDS sensor to measure the solutions while logging the data output of the sensor for 10 seconds. The logged output data was then used to plot the following response figures:

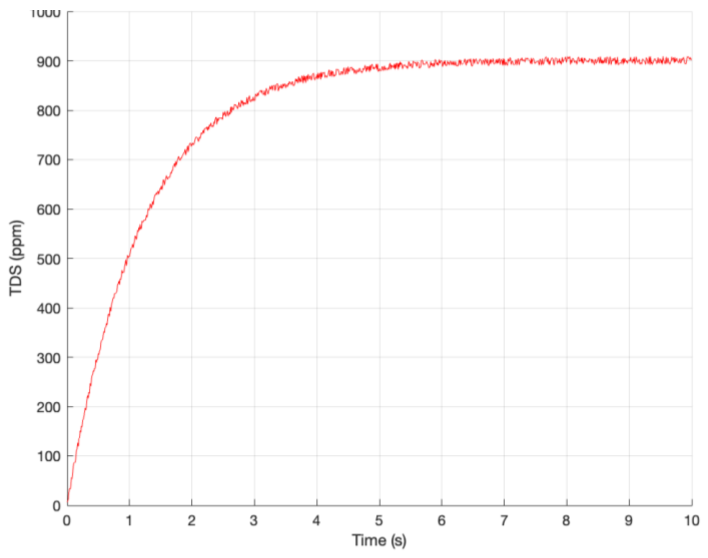


Figure 8: TDS Sensor Response for 900 ppm Test Solution

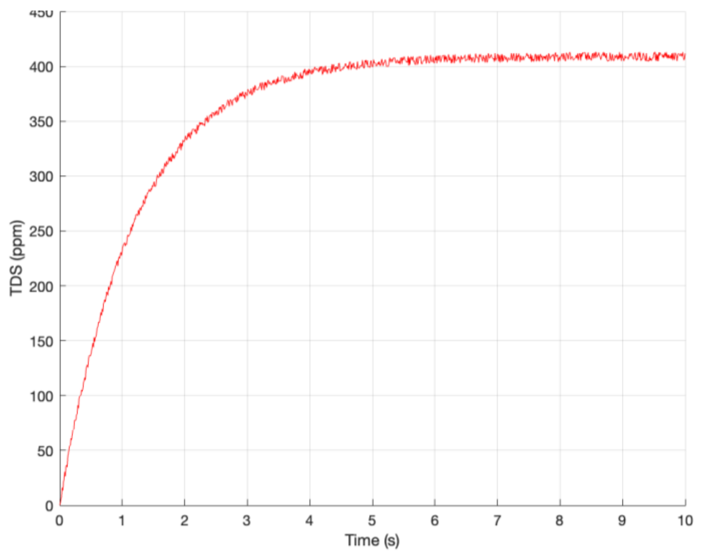


Figure 9: TDS Sensor Response for 410 ppm Test Solution

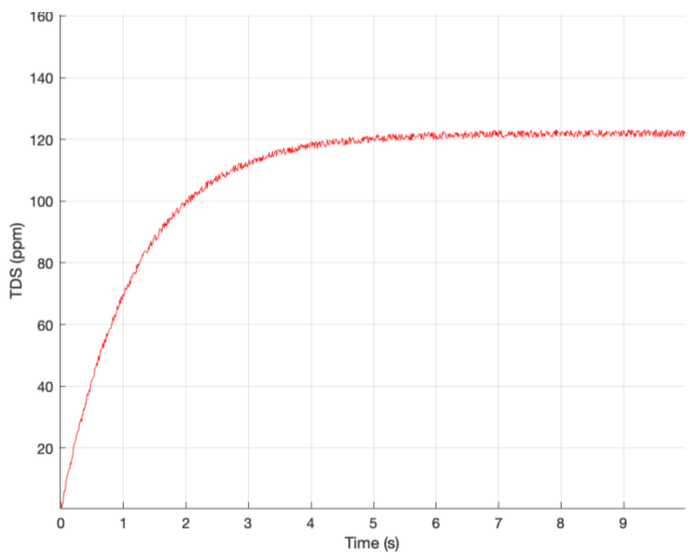


Figure 10: TDS Sensor Response for 120 ppm Test Solution

We observed the TDS sensor response to have an average settling time of 5 seconds. Although the manufacturer did not specify a settling time for the sensor, the response was consistent across the 3 tests we performed. We also observed the steady state output values to be accurate and within ± 9 ppm of the expected result for each solution. Therefore, we can conclude that the TDS sensor has successfully passed the tests as it is functioning as expected.

8.2.5 pH Sensor Testing

In order to test the pH sensor, we purchased 3 standard solutions of pH values of 4.01, 7.0, and 10.01. We then followed the same testing procedure as for the other sensors. The manufacturer specifications state that the pH sensor has a settling time of 8 minutes, as such, we logged the sensor measurements for 10 minutes in order to provide the sensor with adequate time to allow the measurements to stabilize. The measurement data was then plotted to visualize the sensor response as shown in the following figures:

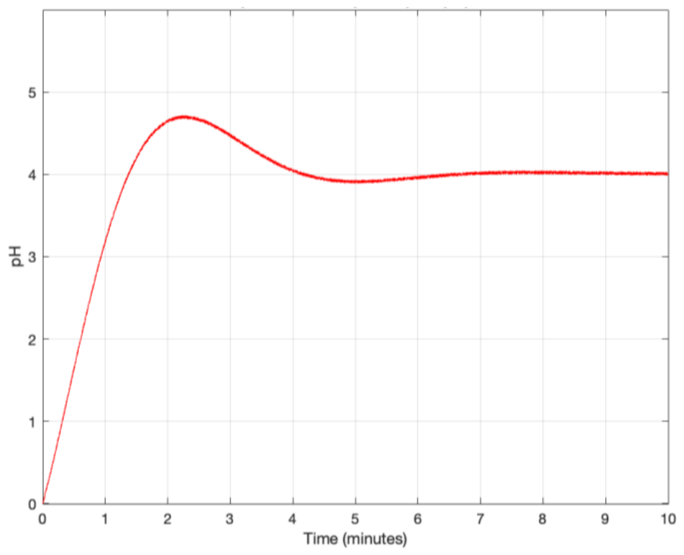


Figure 11: pH Sensor Response for 4.01 pH Test Solution

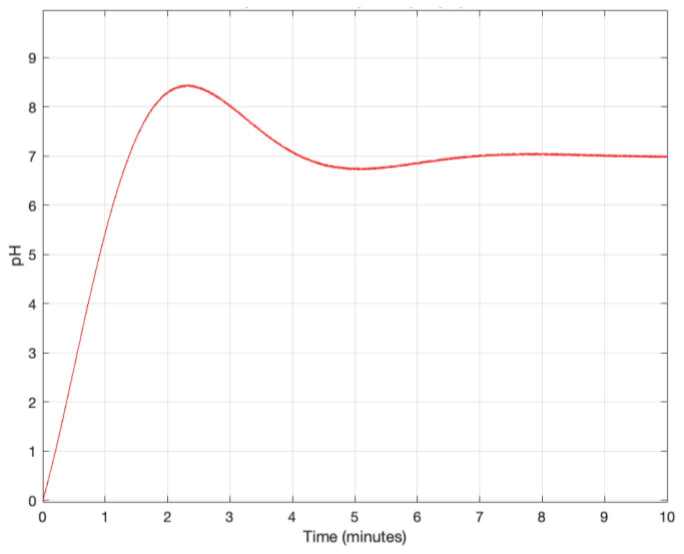


Figure 12: pH Sensor Response for 7.0 pH Test Solution

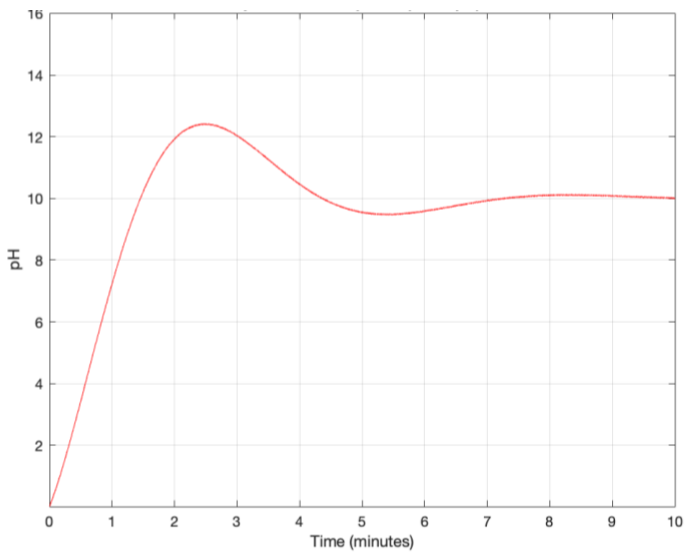


Figure 13: pH Sensor Response for 10.01 pH Test Solution

We observed the average settling time to be 8 minutes, which is exactly what the manufacturer stated. Additionally, the steady state output values were consistent and within ± 0.09 of the expected values across all the tests. Therefore, we can conclude that the pH sensor has successfully passed the tests as its behavior aligns with the expected values.

9 Project Milestones

Below is the updated project timetable with all the major milestones' deadlines. The project Gantt chart with all the other milestones completed throughout the project is attached in the Appendix [Appendix C: Project Progression Gantt Chart].

Milestone	**Due Date**
Project Proposal	January 26, 2024
HW/SW Request Form	February 9, 2024
Progress Report	April 22, 2024
Oral Presentation	April 26, 2024
Poster Fair	June 14, 2024
Final Report and Video	July 5, 2024

Table 16: Major Project Milestones and Due Dates

10 Conclusions

10.1 Conclusion

The focus of our project was to implement a portable and smart drinking water quality monitoring system, in order to address the lack of availability of such product and the disadvantages of using conventional paper test strips to measure water quality. The developed system consists of multiple sensors that infer useful measurements about the water quality, conveniently built into a water bottle, as well as a mobile application that provides an intuitive user interface for the user to interact with the system. This enabled the users to be confident in the quality of water they intend to consume, without compromising convenience or ease of use.

10.2 Recommendations for Future Project Extensions

The physical dimensions of the water bottle housing [Appendix B: Water Bottle Housing Engineering Drawings] are larger than needed. During the design stage of the project, we included extra space in case it was needed during the implementation stage, which caused the final product to be bulky. The water bottle housing would greatly benefit from a reduced size to make it more sleek and easier to transport. Additionally, the updated water bottle housing would benefit from a cap with a built-in flexible straw. This would facilitate in drinking the water contained in the bottle while not affecting the cap functionality of sealing the bottle.

Another possible improvement of the water bottle housing would be the use of smaller sized sensors. This would help in reducing the overall size of the water bottle housing as well. Our initial plan for this project was to incorporate the sensors into the base of the water bottle. This requires surface mount sensors that have a very small footprint of approximately 1 cm by 1 cm. Unfortunately, this size of water quality sensors is not currently available, therefore we had to use conventional sensor probes that have a much larger footprint. If water quality sensors become available in a much smaller size in the future, it would be the best improvement that can be made to this project.

The wiring and circuitry contained within the bottle housing was slightly messy and fragile. Because all the sensors were connected to the Arduino through a breadboard. Although a breadboard makes connecting the hardware components easier, the wires are susceptible to disconnection. To address this, a printed circuit board can be designed for a more secure connection and better organized wiring.

10.3 Reflections

This project was a great experience that although stressful at certain times due to the reduced team size, it presented us with constructive challenges that helped us learn more and develop a resilient work ethic. Additionally, it allowed us to apply a large set of knowledge and skills gained throughout our undergraduate studies, further reinforcing the concepts we learned previously.

Given that all the functional and non-functional project requirement we outlined have been meet, we consider the project successful.

References

[1] “What Is Fusion 360? – Simply Explained,” *All3DP*, Nov. 13, 2019. https://all3dp.com/2/what-is-fusion-360-simply-explained/

[2] Atlas Scientific, “Does Temperature Affect pH?,” *Atlas Scientific*, Jan. 12, 2023. https://atlas-scientific.com/blog/does-temperature-affect-ph/

[3]https://www.researchgate.net/publication/371539432_TEMPERATURE_EFFECT_ON_ELECTRICAL_CONDUCTIVITYEC_TOTAL_DISSOLVED_SOLIDS_TDS_OF_WATER_A_REVIEW#:~:text=In%20general%2C%20as%20the%20temperature,which%20increases%20the%20overall%20conductivity.

[4] “What is EC? How can you differentiate between EC and TDS?,” *www.westlab.com*. https://www.westlab.com/blog/what-is-ec-how-can-you-differentiate-between-ec-and-tds

[5] D. O’Donnell, “Why Electrical Conductivity of Water is Important for Industrial Applications,” *Sensorex*, Oct. 08, 2019. https://sensorex.com/electrical-conductivity-water-important-industrial-applications/

[6] “Why Is pH Important?,” *Atlas Scientific*, Aug. 07, 2021. https://atlas-scientific.com/blog/why-is-ph-important/#:~:text=Whether%20it%20be%20for%20drinking]. A pH probe contains two electrodes that measure the hydrogen ion in a solution. A voltage is generated when there is an ion exchange between the two electrodes

[7] Atlas Scientific, “How Does A pH Probe Work?,” *Atlas Scientific*, Oct. 18, 2021. https://atlas-scientific.com/blog/how-does-a-ph-probe-work/].

[9] Atlas Scientific, “Why Is Turbidity Important?,” *Atlas Scientific*, May 14, 2022. https://atlas-scientific.com/blog/why-is-turbidity-important/]. Currently, the most common method used to treat

water is chlorine. This is due to its capabilities of reacting with organic matter in water which helps in disinfection

[10] [mcoulter, “Chlorination and Turbidity,” *Thompson-Nicola Regional District*. https://www.tnrd.ca/services/water-sewage/chlorination-and-turbidity/]. Turbidity sensors use light to detect suspended particles in water by measuring the light transmittance and scattering rate

[11] “Turbidity_sensor_SKU__SEN0189-DFRobot,” *wiki.dfrobot.com*. https://wiki.dfrobot.com/Turbidity_sensor_SKU__SEN0189#:~:text=It%20uses%20light%20to%20detect (accessed Jul. 01, 2024).]

[12] “Best Home RO Water Purifiers Online - Pureit Water India,” *www.pureitwater.com*. https://www.pureitwater.com/blog/post/understanding-total-dissolved-solids. (accessed Jul. 01, 2024).

[13] *Arduino.cc*, 2024. https://docs.arduino.cc/hardware/uno-r4-wifi/

[14] “Arduino® UNO R4 WiFi,” *Arduino Official Store*. https://store.arduino.cc/products/uno-r4-wifi#:~:text=Expanded%20memory%20and%20faster%20clock

[15] “Difference Between Arduino and Raspberry Pi [Comparison Table],” *webbylab*. https://webbylab.com/blog/arduino-vs-raspberry-pi-key-differences-comparison-table/

[16] “Arduino vs Raspberry Pi: What’s The Difference?,” *InterviewBit*, Nov. 11, 2021. https://www.interviewbit.com/blog/arduino-vs-raspberry-pi/

[17] “A Review of 11 Temperature and Humidity Sensors - DFRobot,” *www.dfrobot.com*. https://www.dfrobot.com/blog-45.html (accessed Jul. 01, 2024).

[18] “TMP100_Temperature_Sensor__SKU_TOY0045_-DFRobot,” *wiki.dfrobot.com*. https://wiki.dfrobot.com/TMP100_Temperature_Sensor__SKU_TOY0045_ (accessed Jul. 01, 2024).

[19] “DS18B20 Temperature Sensor For Arduino - DFRobot Wiki,” *wiki.dfrobot.com*. https://wiki.dfrobot.com/Gravity__DS18B20_Temperature_Sensor__Arduino_Compatible__V2_SKU__DFR0024 (accessed Jul. 01, 2024).

[21] “PH_meter_SKU__SEN0161_-DFRobot,” *wiki.dfrobot.com*. https://wiki.dfrobot.com/PH_meter_SKU__SEN0161_

[22] “Grove - PH Sensor Kit (E-201C-Blue) - Seeed Wiki,” *wiki.seeedstudio.com*. https://wiki.seeedstudio.com/Grove-PH-Sensor-kit/

[23] “PH_meter_SKU__SEN0161_-DFRobot,” *wiki.dfrobot.com*. https://wiki.dfrobot.com/PH_meter_SKU__SEN0161_

[24] “Grove - PH Sensor Kit (E-201C-Blue) - Seeed Wiki,” *wiki.seeedstudio.com*. https://wiki.seeedstudio.com/Grove-PH-Sensor-kit/

[25] “ec meter - DFR0300 - Gravity: Analog Electrical Conductivity Sensor | DFRobot Electronics,” *www.dfrobot.com*. https://www.dfrobot.com/product-1123.html

[26] “Grove - TDS Sensor - Seeed Wiki,” *wiki.seeedstudio.com*. https://wiki.seeedstudio.com/Grove-TDS-Sensor/

[27] “Grove - Turbidity Sensor Meter for Arduino V1.0 - Seeed Wiki,” *wiki.seeedstudio.com*. https://wiki.seeedstudio.com/Grove-Turbidity-Sensor-Meter-for-Arduino-V1.0/

[28] “Gravity: Analog Turbidity Sensor For Arduino - DFRobot,” *www.dfrobot.com*. https://www.dfrobot.com/product-1394.html

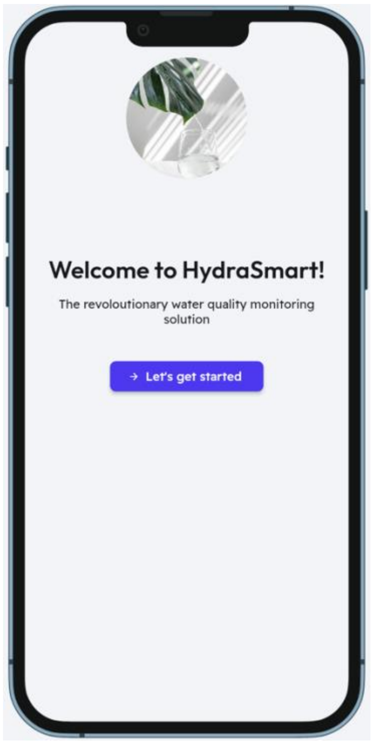
[29] "Gravity_Analog_Electrical_Conductivity_Sensor_Meter_K=10_SKU_DFR0300-H-DFRobot," *wiki.dfrobot.com*. https://wiki.dfrobot.com/Gravity_Analog_Electrical_Conductivity_Sensor_Meter_K=10_SKU_DFR0300-H

[30] "Grove - EC Sensor Kit | Seeed Studio Wiki," *wiki.seeedstudio.com*, Jan. 06, 2023. https://wiki.seeedstudio.com/Grove-EC-Sensor-kit/

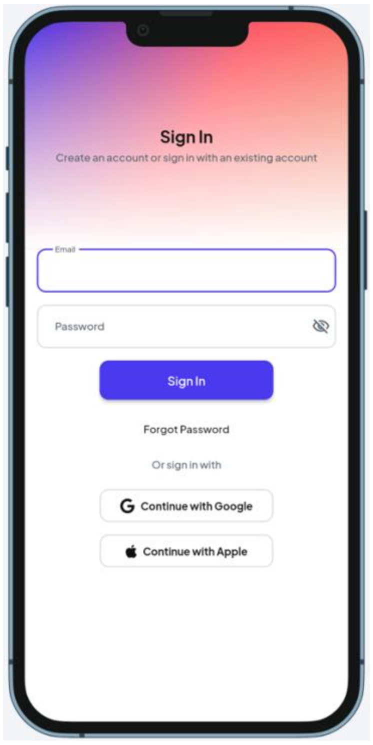
[31] "Flutter vs. React Native: Which One to Choose in 2023?," *Simplilearn.com*. https://www.simplilearn.com/tutorials/reactjs-tutorial/flutter-vs-react-native

[32] "CFR - Code of Federal Regulations Title 21," *www.accessdata.fda.gov*. https://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfcfr/cfrsearch.cfm?fr=177.1020

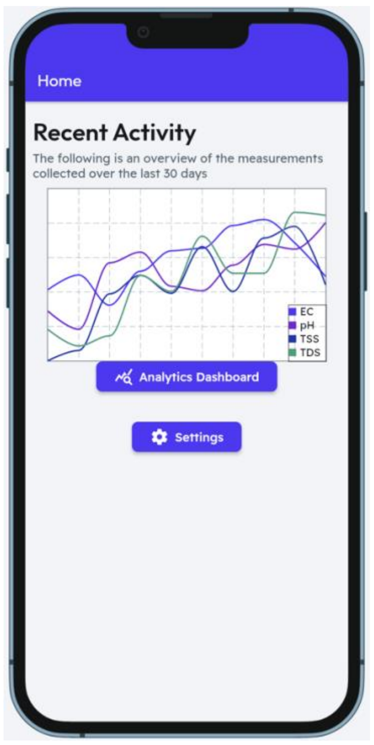
Appendix A: Mobile Application Screens


Welcome to HydraSmart!

The revoloutionary water quality monitoring solution

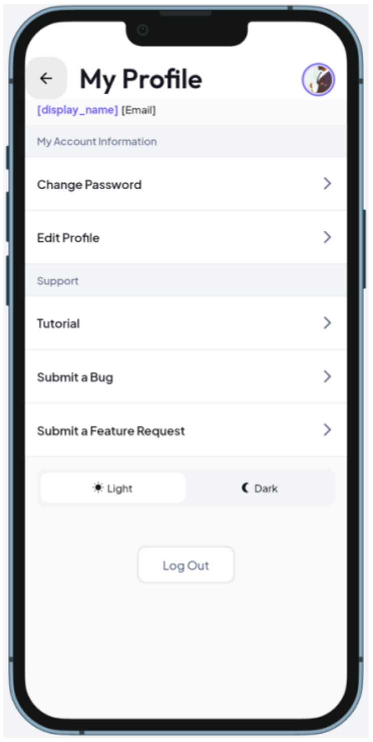

Sign In

Create an account or sign in with an existing account


Home

Recent Activity

The following is an overview of the measurements collected over the last 30 days


My Profile

[display_name] [Email]

My Account Information

Change Password

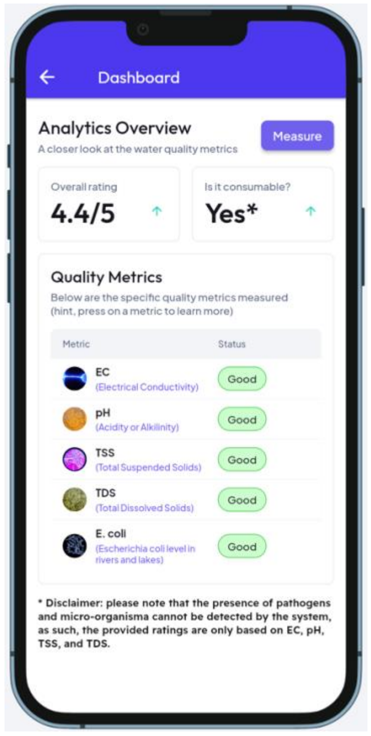
Edit Profile

Support

Tutorial

Submit a Bug

Submit a Feature Request


Dashboard

Analytics Overview

A closer look at the water quality metrics

Overall rating 4.4/5

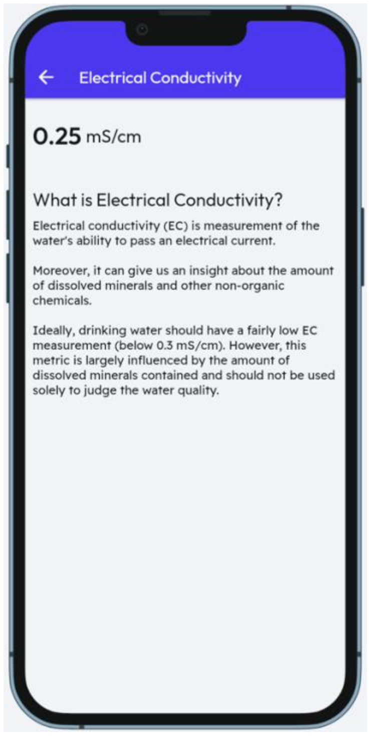
Is it consumable? Yes*

Quality Metrics

Below are the specific quality metrics measured (hint, press on a metric to learn more)

Metric	Status
EC (Electrical Conductivity)	Good
pH (Acidity or Alkilinity)	Good
TSS (Total Suspended Solids)	Good
TDS (Total Dissolved Solids)	Good
E. coli (Escherichia coli level in rivers and lakes)	Good

* Disclaimer: please note that the presence of pathogens and micro-organisma cannot be detected by the system, as such, the provided ratings are only based on EC, pH, TSS, and TDS.


Electrical Conductivity

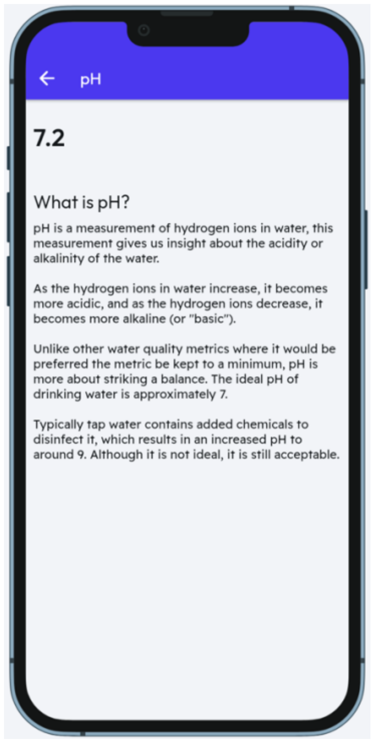
0.25 mS/cm

What is Electrical Conductivity?

Electrical conductivity (EC) is measurement of the water's ability to pass an electrical current.

Moreover, it can give us an insight about the amount of dissolved minerals and other non-organic chemicals.

Ideally, drinking water should have a fairly low EC measurement (below 0.3 mS/cm). However, this metric is largely influenced by the amount of dissolved minerals contained and should not be used solely to judge the water quality.



7.2

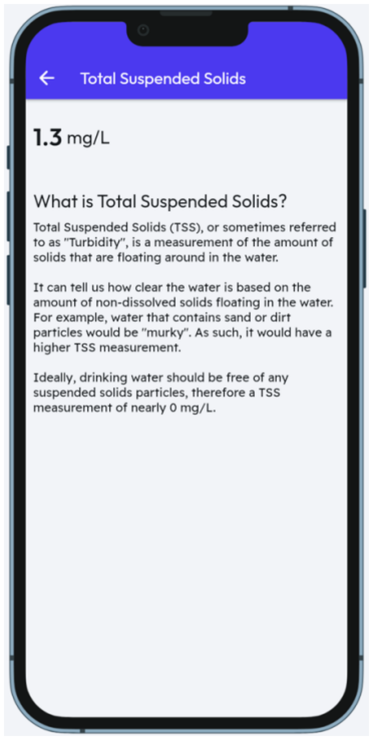
What is pH?

pH is a measurement of hydrogen ions in water, this measurement gives us insight about the acidity or alkalinity of the water.

As the hydrogen ions in water increase, it becomes more acidic, and as the hydrogen ions decrease, it becomes more alkaline (or "basic").

Unlike other water quality metrics where it would be preferred the metric be kept to a minimum, pH is more about striking a balance. The ideal pH of drinking water is approximately 7.

Typically tap water contains added chemicals to disinfect it, which results in an increased pH to around 9. Although it is not ideal, it is still acceptable.



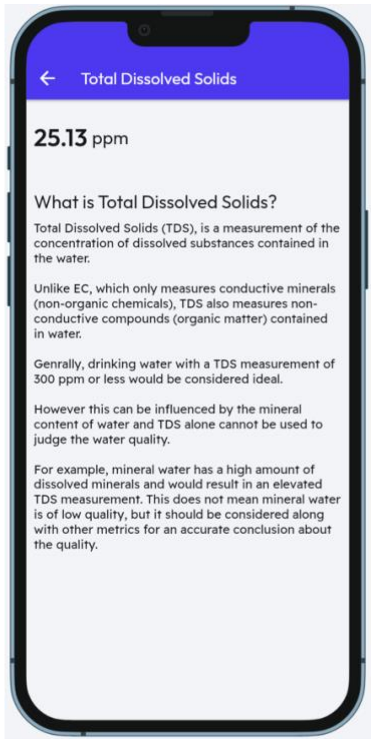
1.3 mg/L

What is Total Suspended Solids?

Total Suspended Solids (TSS), or sometimes referred to as "Turbidity", is a measurement of the amount of solids that are floating around in the water.

It can tell us how clear the water is based on the amount of non-dissolved solids floating in the water. For example, water that contains sand or dirt particles would be "murky". As such, it would have a higher TSS measurement.

Ideally, drinking water should be free of any suspended solids particles, therefore a TSS measurement of nearly 0 mg/L.



25.13 ppm

What is Total Dissolved Solids?

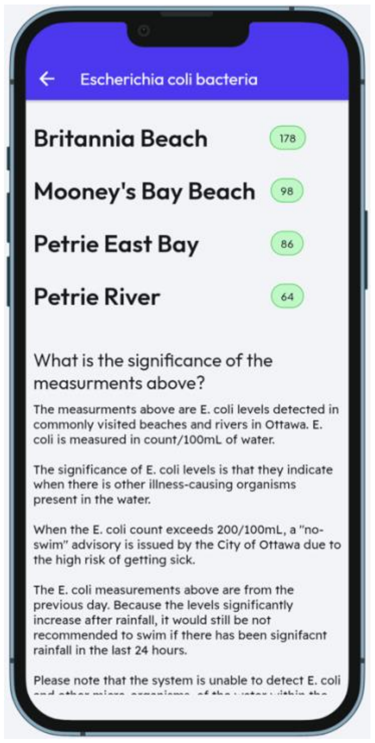
Total Dissolved Solids (TDS), is a measurement of the concentration of dissolved substances contained in the water.

Unlike EC, which only measures conductive minerals (non-organic chemicals), TDS also measures non-conductive compounds (organic matter) contained in water.

Genrally, drinking water with a TDS measurement of 300 ppm or less would be considered ideal.

However this can be influenced by the mineral content of water and TDS alone cannot be used to judge the water quality.

For example, mineral water has a high amount of dissolved minerals and would result in an elevated TDS measurement. This does not mean mineral water is of low quality, but it should be considered along with other metrics for an accurate conclusion about the quality.



Britannia Beach	178
Mooney's Bay Beach	98
Petrie East Bay	86
Petrie River	64

What is the significance of the measurments above?

The measurments above are E. coli levels detected in commonly visited beaches and rivers in Ottawa. E. coli is measured in count/100mL of water.

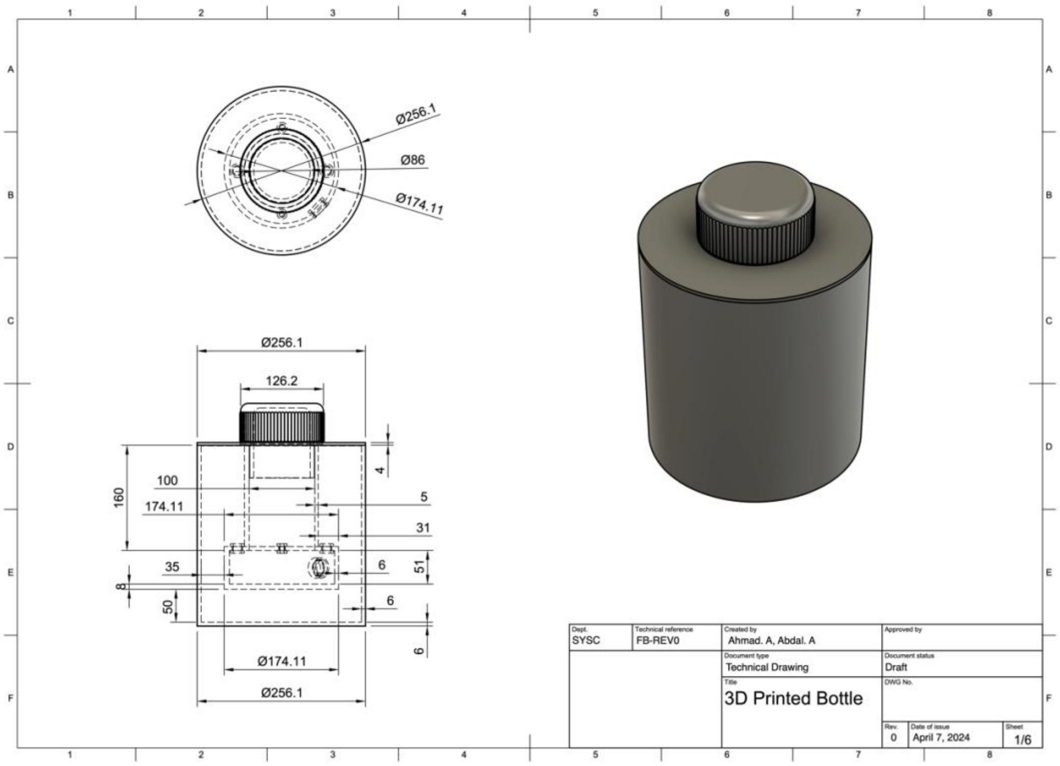
The significance of E. coli levels is that they indicate when there is other illness-causing organisms present in the water.

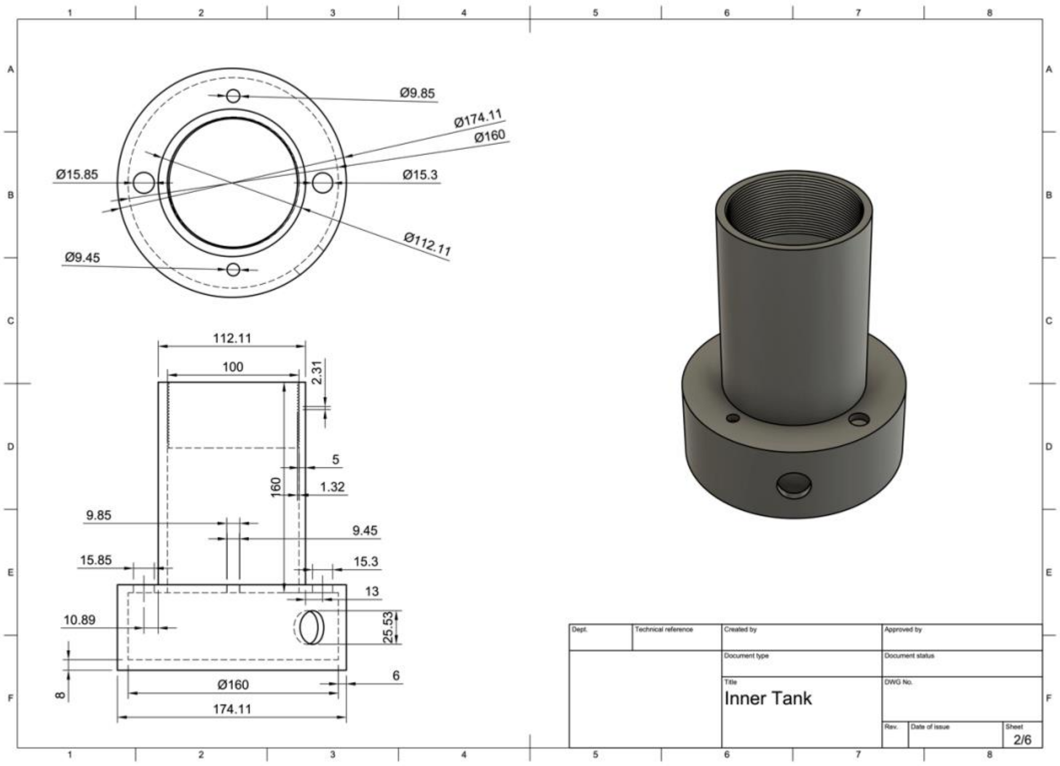
When the E. coli count exceeds 200/100mL, a "no-swim" advisory is issued by the City of Ottawa due to the high risk of getting sick.

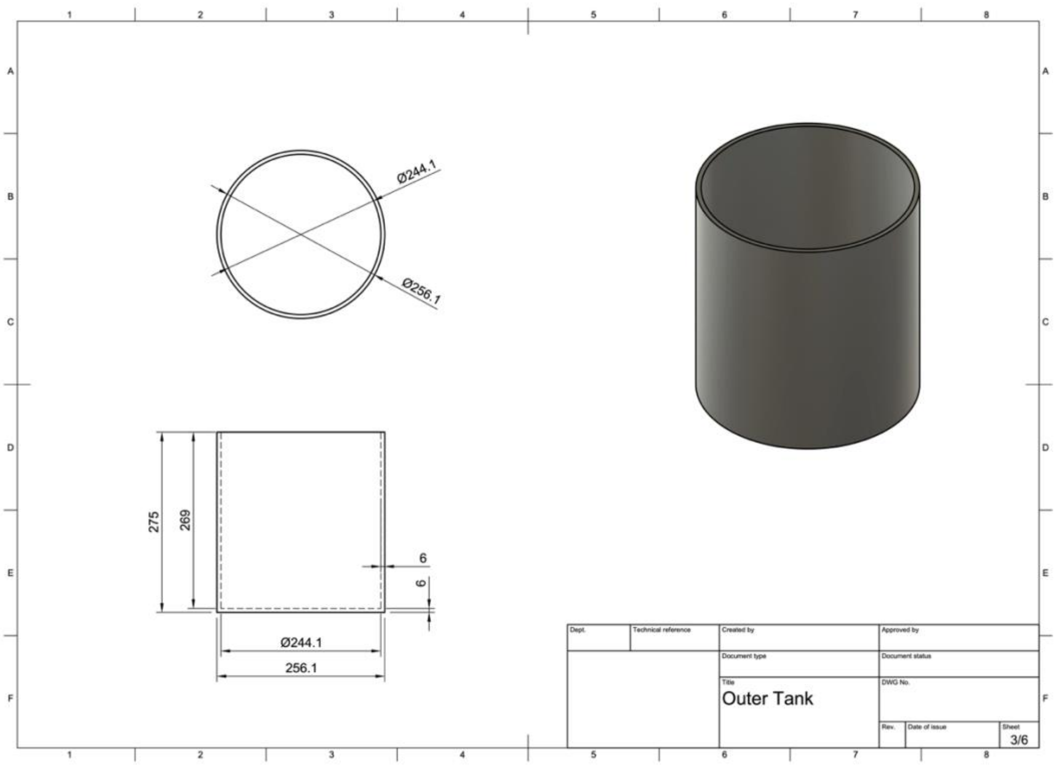
The E. coli measurements above are from the previous day. Because the levels significantly increase after rainfall, it would still be not recommended to swim if there has been signifacnt rainfall in the last 24 hours.

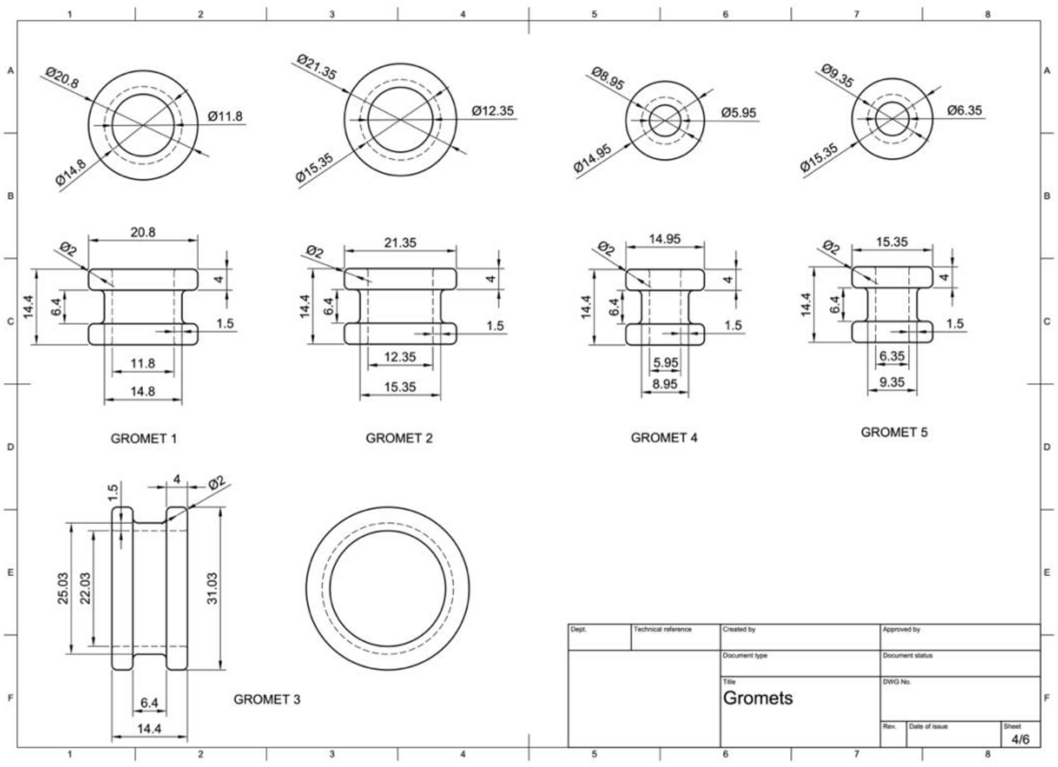
Please note that the system is unable to detect E. coli

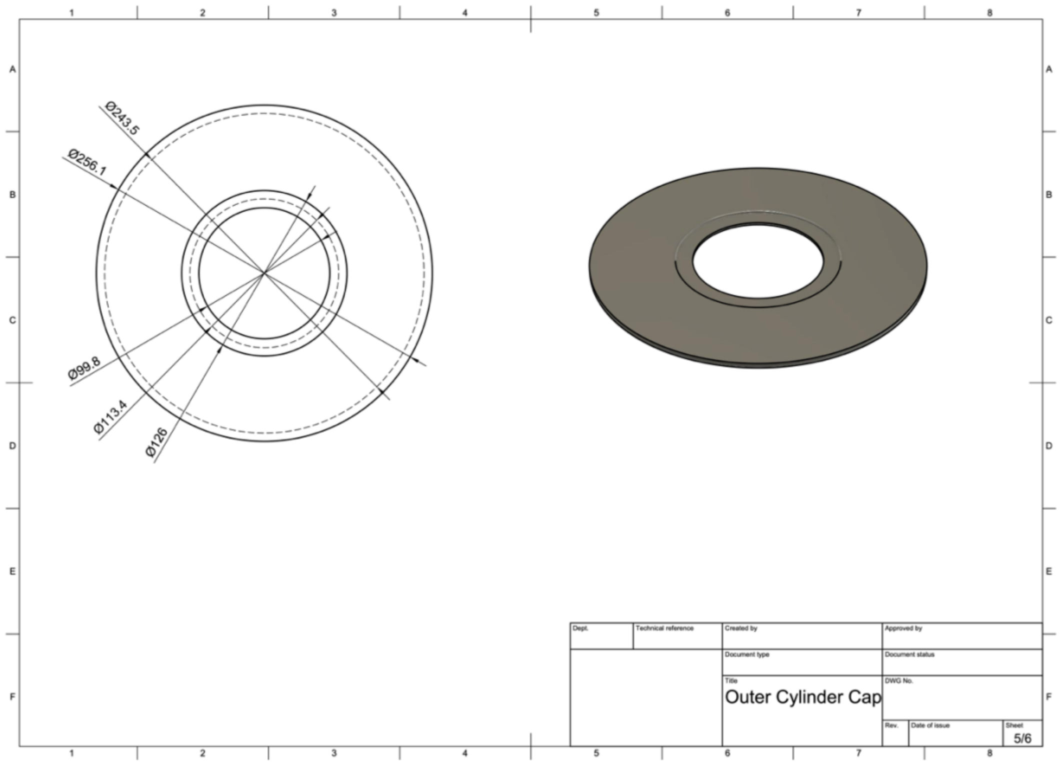
Appendix B: Water Bottle Housing Engineering Drawings

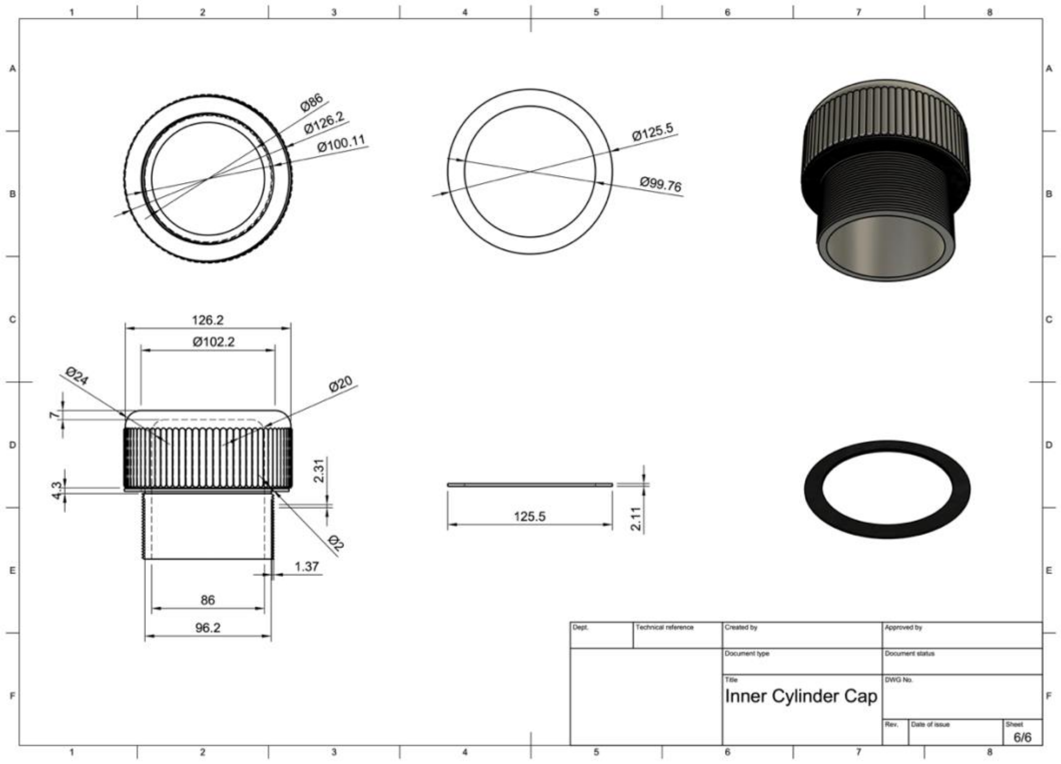




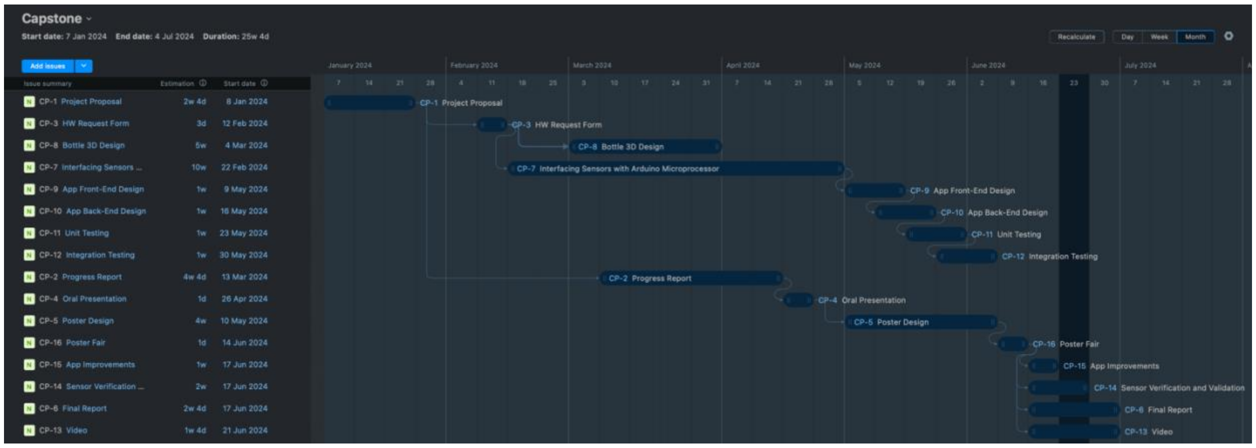

Ø244.1
Ø256.1
275
269
6
6
Ø244.1
256.1
Dept.
Technical reference
Created by
Approved by
Document type
Document status
Title
Outer Tank
DWG No.
Rev.
Date of issue
Sheet
3/6


Ø20.8
Ø11.8
Ø14.8
Ø21.35
Ø12.35
Ø15.35
Ø8.95
Ø5.95
Ø14.95
Ø9.35
Ø6.35
Ø15.35
20.8
Ø2
14.4
6.4
4
1.5
11.8
14.8
GROMET 1
21.35
12.35
15.35
GROMET 2
14.95
5.95
8.95
GROMET 4
15.35
6.35
9.35
GROMET 5
1.5
4
Ø2
25.03
22.03
31.03
6.4
14.4
GROMET 3
Dept.
Technical reference
Created by
Approved by
Document type
Document status
Title
Gromets
DWG No.
Rev.
Date of issue
Sheet
4/6


Ø243.5
Ø256.1
Ø99.8
Ø113.4
Ø126
Dept.
Technical reference
Created by
Approved by
Document type
Document status
Title
Outer Cylinder Cap
DWG No.
Rev.
Date of issue
Sheet
5/6


Ø86
Ø126.2
Ø100.11
Ø125.5
Ø99.76
126.2
Ø102.2
Ø24
Ø20
7
4.3
2.31
Ø2
1.37
86
96.2
125.5
2.11
Dept.
Technical reference
Created by
Approved by
Document type
Document status
Title
Inner Cylinder Cap
DWG No.
Rev.
Date of issue
Sheet
6/6

Appendix C: Project Progression Gantt Chart



Capstone

Start date: 7 Jan 2024 **End date:** 4 Jul 2024 **Duration:** 25w 4d

Issue summary	Estimation	Start date
CP-1 Project Proposal	2w 4d	8 Jan 2024
CP-3 HW Request Form	3d	12 Feb 2024
CP-8 Bottle 3D Design	5w	4 Mar 2024
CP-7 Interfacing Sensors ...	10w	22 Feb 2024
CP-9 App Front-End Design	1w	9 May 2024
CP-10 App Back-End Design	1w	16 May 2024
CP-11 Unit Testing	1w	23 May 2024
CP-12 Integration Testing	1w	30 May 2024
CP-2 Progress Report	4w 4d	13 Mar 2024
CP-4 Oral Presentation	1d	26 Apr 2024
CP-5 Poster Design	4w	10 May 2024
CP-16 Poster Fair	1d	14 Jun 2024
CP-15 App Improvements	1w	17 Jun 2024
CP-14 Sensor Verification ...	2w	17 Jun 2024
CP-6 Final Report	2w 4d	17 Jun 2024
CP-13 Video	1w 4d	21 Jun 2024

Appendix D: Project Proposal Document

ECOR4907

Capstone Project

Portable Water Quality Testing System

Project Proposal

Prepared for

Safaa Bedawi, Ph.D., P.Eng

By

Ahmad Alkawasmeh, 101108706

Abdal Alkawasmeh, 101198752

January 26, 2024

Table of Contents

1 Introduction 4
1.1 Background 4
1.2 Motivation 4
1.3 Project Objective 5
1.4 Specific Goals 5
2 Engineering Design 6
2.1 Project Management 6
2.2 Justification of Suitability for Degree Program 6
2.3 Individual Contributions 7
2.3.1 Report Contributions 7
3 Work Plan 7
3.1 The Project Team 7
3.1.1 Roles and Tasks 7
3.1.2 Teamwork Strategy 8
3.1.3 What we will need to learn 8
3.2 Project Milestones 8
3.3 Schedule of Activities 9
4 Project Risks and Mitigation Strategies 9
4.1 Electrical Shock 9
4.2 Fire and Burns 9
5 Hardware Required 9
5.1 List of Required Hardware 9
5.2 Trade-off Analysis 11
5.2.1 Microprocessor 11
5.2.2 Microcontroller 11
5.2.3 Power Supply 11
5.2.4 Analog to Digital Converter 12
5.2.5 Sensors 12
5.2.6 Potentiometer, Bread Board, and Jumper Wires 12
5.2.7 Display 12
6 References 13
Appendix A: Microprocessor Datasheet 14

Appendix B: Microcontroller Datasheet 14
Appendix C: Battery Pack Datasheet 14
Appendix D: ADC Datasheet 14
Appendix E: pH Sensor Datasheet 14
Appendix F: Turbidity Sensor Datasheet 14
Appendix G: TDS Sensor Datasheet 14
Appendix H: EC Sensor Datasheet 14
Appendix I: Display Datasheet 14

1 Introduction

We are Ahmad and Abdal, 4th year undergraduate Software Engineering students. The purpose of this document is to propose the project we wish to develop and work on throughout the Capstone Project. As well as capture all the fine-grained details of how this project will be carried out. The project we are proposing is an intelligent and portable water testing system.

1.1 Background

We go about our everyday life without giving much thought to the quality of water we consume on a daily basis. Especially that we need to intake about 3.7 Litres of water per day [1] to maintain healthy body functions, therefore we must pay close attention to the quality of the water we consume.

Currently, the most widely used technique of determining the quality of the water is by using Test Strips. Which are rectangular pieces of paper that are saturated with a chemical agent, once these paper strips are submerged in a water sample, the chemical agent will react with the water and change colour. Then this colour will indicate the test result, which then the user will need to match the colour produced by the paper strip to a result chart with all the possible colours that can be produced by the test strip and what they indicate. This method of testing the water quality is not accurate and cannot provide exact quantitated results.

1.2 Motivation

Therefore, a better solution is needed to replace the paper strips test most commonly used today. With a portable water testing system, you would be able to determine the quality of the water available to you in a pinch. This is essential as there is no other way of finding out whether the water is acidic or if it contains small particles of contaminants, and as such you wouldn't be able to judge if it is safe or healthy for consumption.

There is a wide range of applications for such a system and it can be useful to everyone on a daily basis. However, it can be more beneficial to users such as mothers with newborn babies, teachers, and parents in general. This system would confirm the quality of the water prior to consumption.

For the parents, it would put their mind at ease after being aware of the water quality their children will consume, especially when travelling or visiting a new city for the first time where they aren't sure of how that specific municipality treats their water.

For the teachers, for example if they are accompanying children on a field trip, it would be immensely helpful to be able to check the water quality before everyone consumes it. Moreover, if the water happens to be acidic or contains contaminants, it will not provide the necessary hydration at the least, and worst-case scenario it would cause everyone to get sick. Thus, a simple test prior to consumption would allow us to mitigate any risks of drinking water of unknown quality or content.

1.3 Project Objective

The objective of this project is to develop a portable and intelligent water quality testing system that will enable users to check the quality of the water on the go.

1.4 Specific Goals

More precisely, the system shall provide the user with insight about the water including its pH level, amount of Total Suspended Solids, Total Dissolved Solids, it's Electrical Conductivity or Mineral Content and Temperature. The following is a list of functional and non-functional requirement we are aiming to achieve by the end of this project:

Functional Requirements:

- The system shall be able to measure the pH level of the water.
- The system shall be able to measure the amount of Total Dissolved Solids (TDS) in the water.
- The system shall be able to measure the amount of Total Suspended Solids (TSS or Turbidity) in the water.
- The system shall be able to measure the Electrical Conductivity (EC) of the water.
- The system shall be able to measure the Temperature of the water.
- The system shall be able to run remotely using a rechargeable battery.
- The system shall have a graphical user interface.

Non-Functional Requirements:

- The system shall be able to infer a useful observation(s) about the water quality using the sensor measurements.
- The system's user interface shall have a simple and an advanced mode of operation.
- The simple operation mode shall be tailored to a user with no prior expertise in water quality nor the measurements produced by the sensors.

- The advanced mode of operation shall be tailored to a user with previous expertise in water quality and/or the specific measurements collected.
- The user interface shall be easy to use and intuitive.

2 Engineering Design

2.1 Project Management

Although the project is currently in an early stage, we greatly benefited from creating a Gantt Chart in order to schedule and organize the execution of the project tasks in the upcoming months. We will continue to use and update the Gantt Chart as the project progresses in order to carry out the project objectives effectively and efficiently.

Additionally, we will be using YouTrack in order to track our progress and identify any issues that must be resolved in order to continue making progress. This will also enable effective and informed collaboration between us.

The use of GitHub issue tracking will be considered as well. Although using both YouTrack and GitHub issue tracking will be redundant and unnecessary. Thus, a choice shall be made between YouTrack and GitHub issue tracking.

2.2 Justification of Suitability for Degree Program

We are both Software Engineering students with a parallel experience in software development, however, each person has a unique interest and background experience.

Ahmad has taken special interest in Linux Operating System and hardware components. He has always been keen to take on personal projects that utilize the Raspberry Pi microprocessor, such as a smart mirror and automatic door lock and opener.

Abdal’s interest is purely vested in software development. More precisely in user interface design and development. Additionally, he has taken on multiple personal projects in the past involving website design and development, and he has an extensive experience in working with distributed database systems.

Collectively, we believe that we have developed the suitable skills and experience throughout our degree program, and by taking on personal projects in the past.

2.3 Individual Contributions

2.3.1 Report Contributions

Ahmad Alkawasmeh	Abdal Alkawasmeh
Chapter 1, Chapter 3, Chapter 4, Appendences	Chapter 2, Chapter 5, Chapter 6

Table 1: Breakdown of Report Contributions of Each Member

3 Work Plan

3.1 The Project Team

We are both Software Engineering students with a main knowledge base consisting of Object-Oriented Programming. However, we've also had some exposure to working with hardware in the past, more specifically the Raspberry Pi, where we have worked on personal projects including a smart mirror and garage door opener. This makes us a strong fit for this project as we possess the necessary experience and knowledge to complete this project.

3.1.1 Roles and Tasks

The table below reflects the specific tasks that are assigned to each member, which is based on interest and prior experience of a member with that specific task.

Ahmad	Abdal
Developing the main software functionalities	Developing the main software functionalities
Unit Testing	Integration Testing
Interfacing the EC and pH sensors with the Raspberry Pi	Interfacing the TDS and TSS sensors with the Raspberry Pi
Designing the housing for the system using 3D design software	Developing the GUI
Interfacing the screen and buttons with the Raspberry Pi and GUI	Test fitting all the components into the designed housing

Table 2: Assigned Tasks to Each Member

3.1.2 Teamwork Strategy

The way we will ensure the work is done effectively and efficiently as a team I going to be mostly through GitHub version control and weekly team meetings. The team communication will be ongoing and constant throughout the project. Moreover, since we are siblings and we live together, we are in contact every day and will keep each other informed of what we are currently working on. This is to ensure we are working towards our goals effectively and efficiently.

We will also be using Jet Brains Labs YouTrack to keep track of all ongoing activities and tasks being worked on.

3.1.3 What we will need to learn

Although we are familiar with GitHub version control, we need to learn more about GitHub Issue Tracking. This tool is going to be immensely useful to ensure that issues or incompatibilities, with each of our code parts, are kept track of and solved efficiently.

We will also have to revisit 3D design software as our experience with working with the software PTC Creo was in our first year of university and we need a refresher on how to use it.

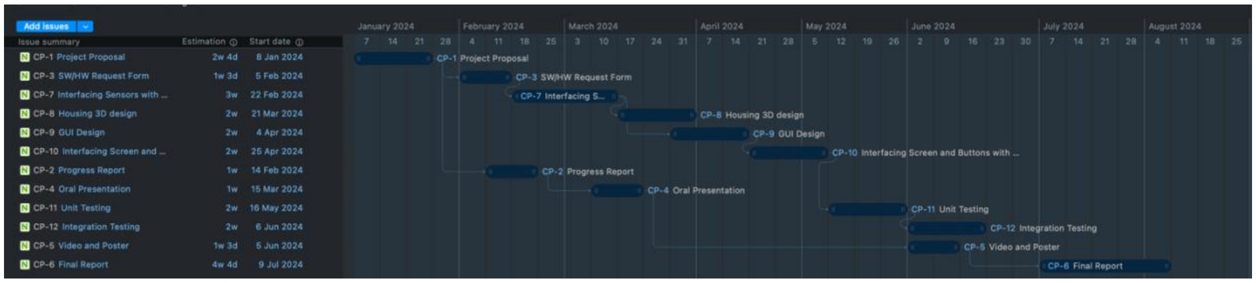
3.2 Project Milestones

The table below reflects the major milestone deadlines and the dates on which they are to be handed in.

Milestone	Date
Project Proposal	January 26, 2024
Progress Report	February 16, 2024
SW/HW Request Form	February 14, 2024
Oral Presentation	March 22, 2024
Video and Poster	June 14, 2024
Final Report	July 31 to August 10, 2024

Table 3: Major Milestones and Associated Due Dates

3.3 Schedule of Activities



Issue summary	Estimation	Start date
CP-1 Project Proposal	2w 4d	8 Jan 2024
CP-3 SW/HW Request Form	1w 3d	5 Feb 2024
CP-7 Interfacing Sensors with ...	3w	22 Feb 2024
CP-8 Housing 3D design	2w	21 Mar 2024
CP-9 GUI Design	2w	4 Apr 2024
CP-10 Interfacing Screen and ...	2w	25 Apr 2024
CP-2 Progress Report	1w	14 Feb 2024
CP-4 Oral Presentation	1w	15 Mar 2024
CP-11 Unit Testing	2w	16 May 2024
CP-12 Integration Testing	2w	6 Jun 2024
CP-5 Video and Poster	1w 3d	5 Jun 2024
CP-6 Final Report	4w 4d	9 Jul 2024

Figure 1: Gantt Chart Outlining Schedule of Project Tasks Execution

4 Project Risks and Mitigation Strategies

The risks associated with this project, although are minimal, they still exist and must be addressed appropriately with a mitigation strategy to avoid any bodily harm to the user.

4.1 Electrical Shock

Due to the utilization of electronics and electrical components in this project, there is a risk of electrical shock to the end user if the device is misused. In order to mitigate this risk. The electrical connections and electronic components will be placed in a compartment that is inaccessible to the user. Moreover, the user will not be able to make contact with the wiring and electrical components of the system, thus mitigating the risk of an electrical shock.

4.2 Fire and Burns

One of the system components will be a rechargeable battery pack, which will be used to supply power to the system and allow it to be mobile. The battery pack type is Lithium-Ion, which remains in a stable state so long the battery casing is intact. If the casing becomes punctured or compromised at any time, the battery may catch on fire [2]. This would be catastrophic as it can cause burns to the user. The strategy to mitigate this risk is to apply metallic plates to the battery pack. This will reinforce the battery casing and would make it much harder to penetrate the original thin plastic casing of the battery.

5 Hardware Required

5.1 List of Required Hardware

The following table captures the details of the hardware needs for our project.

Part	Description	Link	Quantity	Price in CAD (tax inclusive)
Microprocessor	Raspberry Pi 4 8GB	Link	1	(Available at Carleton)
Microcontroller	Texas Instruments EK-TM4C123GXL ARM Development Board	Link	1	$37
Power Supply	PiJuice Hat	Link	1	$118
ADC Converter	Analog to Digital Converter IC MCP 3008	Link	3	$16 ($5.30 per 1 count)
pH Sensor	Gravity Analog pH Sensor Kit	Link	1	$61
Turbidity Sensor	Gravity Analog Turbidity Sensor	Link	1	$15
TDS Sensors	Gravity Analog TDS Sensor	Link	1	$18
EC Sensor	Gravity Analog EC Sensor	Link	1	$107
Temperature Sensor	Gravity Digital Temperature Sensor	Link	1	$12
Potentiometer	Gravity Analog Potentiometer	Link	1	$5
Breadboard	DfRobot Solderless Breadboard	Link	1	$5
Jumper Wires	Male to Male Jumper Wires	Link	1 pack (includes 20 pieces)	$4
Jumper Wires	Female to Male Jumper Wires	Link	1 pack	$4

			(includes 20 pieces)	
Display	5" Touch Display	Link	1	$65

Table 4: Required Components, Website Links, and Price List

5.2 Trade-off Analysis

5.2.1 Microprocessor

There was a choice that needed to be made between the Raspberry Pi microprocessor and Arduino Due microcontroller [3]. Given our extensive experience with the Raspberry Pi, we decided to go with it instead of the Arduino Due.

The main differences between them is their ability to process digital and analog signals. Where the Arduino Due has the ability to process both digital and analog signals, the Raspberry Pi is only able to process digital signals.

Given the majority of the sensors we are aiming to utilize produce analog signal output, we needed to address the downfall of the Raspberry Pi. This will be achieved using the MCP 3008 Analog to Digital Converter.

5.2.2 Microcontroller

The use of a custom, dedicated microcontroller for our system will always be superior to using an off the shelf microprocessor like the Raspberry Pi. We are planning on developing a preliminary prototype with the Raspberry Pi as a proof of concept. Then if time permits, we will develop a dedicated microcontroller firmware for our system using the Texas Instruments EK-TM4C123GXL ARM Development Board. This board is more power efficient, costs a quarter of a Raspberry Pi, and will have a much smaller footprint in comparison to the Raspberry Pi. We have consulted with a graduated colleague, with a vast experience in electronics and embedded systems, about the board choice. He recommended this specific development board due to its form factor and sufficient power capability for our system.

5.2.3 Power Supply

In case of the power supply, the choice was between a power bank that is intended for use with mobile devices or a PiJuice Hat that is specifically intended for use with the Raspberry Pi. The PiJuice Hat has a much smaller footprint and integrates smoothly with the Raspberry Pi, when compared to a power bank that has a larger footprint and has to be plugged in manually every time you need to power the Raspberry Pi. Due to the downfalls of a power bank, the choice was made in favour of the PiJuice Hat.

5.2.4 Analog to Digital Converter

The choice for an Analog to Digital Converter was between the MCP3008 and ADS1x15 [4]. The MCP3008 is capable of 8 channels of input simultaneously where the ADS1x15 is only capable of 4. Also, the MCP3008 cost is one third of the ADS1x15. The better capabilities of the MCP3008 combined with its superior value, has led us to make a choice in favour of the MCP3008.

5.2.5 Sensors

The two makers of sensors we considered are Grove [5] and DfRobot [6]. Both manufacturers' sensors are capable of the same measurements' accuracies. However, Grove priced their sensors much higher than DfRobot and they had less support available for their sensors.

The fact that DfRobot's sensors were of better value combined with the availability of sufficient support and documentation, has led us to make a choice in favour of DfRobot's sensors.

5.2.6 Potentiometer, Bread Board, and Jumper Wires

The purpose of these components will only be for rapid prototyping and testing the system. After which, they will become obsolete as we plan to move on from messy wiring and circuitry to a sleeker solution consisting of a printed circuit board. Thus, the only factor that played a role in choosing these components is price. There is no reason for choosing a higher priced jumper wire, for example, if they both capable of making the required connection.

5.2.7 Display

In case of the display choice, it was the most challenging as it required planning for a much more advanced stage of our project. There was three main factors to consider, connectivity, display size, and price.

In case of the connectivity, the choice was between a display that connected through the GPIO, HDMI connector, or the Display Serial Interface. We are unable to use the GPIO option as we will need to use the GPIO for our sensors connectivity. The HDMI connector is a good option but it will still require the use of GPIO pins for touch input. This leaves us with the DSI which is capable of handling the video stream output to the display and touch input simultaneously.

Once the connectivity method was decided upon, now a suitable size needed to be picked. We wanted to pick a size that is large enough for the user to comfortable interact with the user interface, and simultaneously keeping the device form factor as small as possible. There was two size options we arrived at, 5 and 7 inches displays. The 7 inch display would be too large and nearly the size of an iPad display, thus a more suitable size would be the 5 inch display.

With the connectivity and size options decided, there is one more variable left that plays a role in choosing a display. That is the price. At this point, there is no reason for choosing a more expensive display if it achieves the same functionality as the less expensive option and reflects our needs. The choice was made in favour of the Waveform 5 inch DSI Touch Display. Since it satisfied our needs and it had great reviews, it was the best valued option for our application.

6 References

Your background and motivation sections will be more convincing if you refer to the literature (stats, articles, etc.) I suggest you use a citation manager, such as Mendeley. Otherwise, you will have to format your citations by hand. You can use any citation style (e.g., IEEE, Vancouver, Chicago); just be consistent. You can also copy pre-formatted citations directly from Google Scholar results.

[1] “Water: How much should you drink every day?,” Mayo Clinic, Oct. 12, 2022. https://www.mayoclinic.org/healthy-lifestyle/nutrition-and-healthy-eating/in-depth/water/art-20044256#:~:text=So%20how%20much%20fluid%20does,fluids%20a%20day%20for%20women

[2] “Lithium-ion-batteries | DFES.” https://www.dfes.wa.gov.au/hazard-information/fire-in-the-home/lithium-ion batteries#:~:text=There%20are%20several%20situations%20that,piercing%2C%20and%2For%20vibrations)

[3] “Arduino due,” Arduino Official Store. https://store.arduino.cc/products/arduino-due

[4] A. Industries, “ADS1115 16-Bit ADC - 4 Channel with Programmable Gain Amplifier.” https://www.adafruit.com/product/1085

[5] “Grove - Seeed Studio.” https://www.seeedstudio.com/category/Grove-c-1003.html

[6] dfrobot.com, “Buy Sensors Gravity - DFRobot,” dfrobot.com. https://www.dfrobot.com/topic-282-36.html

Appendix A: Microprocessor Datasheet

Raspberry Pi 4 Model B Datasheet

Appendix B: Microcontroller Datasheet

TM4C123G LaunchPad Evaluation Board Datasheet

Appendix C: Battery Pack Datasheet

PiJuice Hat Datasheet

Appendix D: ADC Datasheet

MCP3008 Analog to Digital Signal Converter Datasheet

Appendix E: pH Sensor Datasheet

Gravity Analog pH Sensor Kit Datasheet

Appendix F: Turbidity Sensor Datasheet

Gravity Analog Turbidity Sensor Datasheet

Appendix G: TDS Sensor Datasheet

Gravity Analog TDS Sensor Datasheet

Appendix H: EC Sensor Datasheet

Gravity Analog EC Sensor Datasheet

Appendix I: Display Datasheet

Waveshare 5" DSI Touch Display Datasheet