THE UNIVERSITY OF BRITISH COLUMBIA

IoT Monitoring of Aquaponic and Hydroponic Food Production

Design

UBC Electrical and Computer Engineering Capstone 121

Carson Berry
Lynes Chan
Mason Duan
Jayden Leong
Hannah Xu

for

UBC SEEDS Sustainability Program
Bernhard Nimmervoll, UBC Mechanical Engineering

April 16, 2021

Revision History

Date	Author	Sections	Change
2020-10-05	JL	1.0 - 2.0	Creation of document skeleton
2020-10-30	JL	1.0 - 5.0	Added system overview and skeleton of subsystem sections
2020-11-29	LC	3.0 - 4.0	Added the design decisions made for the sensors and the control devices
2021-02-19	CB	3.0 - 4.0	Added details about power architecture, sensors, mounting, hardware, heat management
2021-02-20	JL	1.0 - 2.0	Updated system overview to add brief overview of hydroponics and aquaponics, new system diagram and updated purpose. Updated IoT section with details about the cloud implementation, along with other minor edits to microcomputer section.
2021-02-20	MD	6.0	Updated software section to include use case digagram, screen explanations, and how the mobile application interacts with the back end.
2021-04-11	CB	3.0-4.0, 7.0	Updated power design, sensor firmware, hardware design, added BOM tables, and ordering information.
2021-04-12	JL	1.0-2.0	Added references and further detail to overview and background sections. Updated IoT section with references and implementation details.
2021-04-14	MD	6.0	Updated software section to include final use case diagram design, screen explanations, and how the mobile application interacts with the back end.

Contents

1 System Overview

Since our project is designed to work with both hydroponic and aquaponic systems, this section provides an brief introduction to both hydroponics and aquaponics. Afterward, the purpose of this project is given, and the high-level system design is described.

1.1 Hydroponics and Aquaponics

Hydroponics is a method of agriculture where plants are grown in a nutrient-rich aqueous solution [1]. Nutrients in hydroponics are inputted by the grower into the aqueous solution to nourish plants. This contrasts traditional forms of agriculture, where plants are grown in soil.

Aquaponics is very similar to hydroponics. In aquaponics, plants are also grown in an aqueous solution, however plants recieve nutrients in a different manner. Instead of growers inputting nutrients, plants recieve nutrients from fish waste [2]. To make this possible, fish are grown in separate tanks from the plants, with a filtration tank, and sump tank between them. Aquaponics recieves its name because of this, by being a mixture of aquaculture and hydroponics. A simplified diagram of aquaponics is show in Figure 1. Note that hydroponics systems look nearly the same, minus the fish tank.

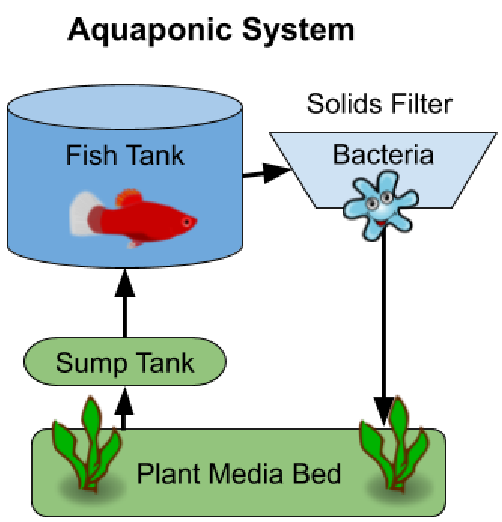

Figure 1: A diagram of a basic aquaponics system

1.2 Project Purpose

The goal of our project is to develop a system to monitor and control critical variables for hydroponic and aquaponic systems. In particular, our project focuses on monitoring water variables as they are most difficult to maintain at healthy levels in aquaponic and hydroponic systems. These variables include water pH, level, leakage, and temperature.

In terms of users, this system will be designed for small scale and hobbyist growers who maintain their aquaponic or hydroponic systems. Note that for this project, small-scale aquaponics or hydroponic systems are defined as systems with under 200 plants and under 3000 gallons of water. Since our users are small-scale growers and hobbyists, the system will utilize open-source and low cost technologies wherever possible to minimize barriers to usage and replication.

Please refer to the included *Requirements* document for further details about system requirements, background, and purpose.

1.3 High-Level System Design

Figure 2 below provides an overview of our design. As you can see, our design contains the following four subsystems: sensors, control, Cloud, and software application.

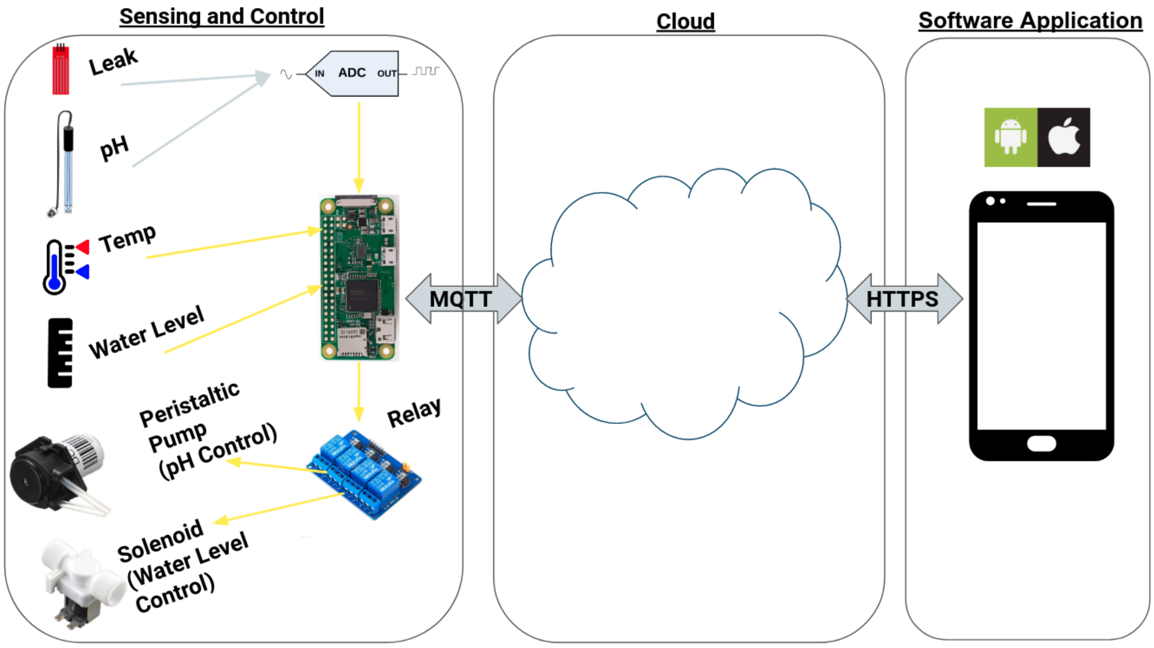

Figure 2: System Level Block Diagram

First, we can see the sensing and control subsystems in the “Sensing and Control” block in Figure 2. This consists of a microcomputer attached to multiple sensors and control devices. As previously mentioned, there are four key sensors in our project shown: the water leakage, pH, temperature, and level sensors. These sensors allow our design to take measurements of the water of aquaponics or hydroponics systems. As such, the pH, temperature, and water level sensors are placed inside water tanks in an aquaponic or hydroponic system. As far as the control systems are concerned, two devices are attached to the microcomputer: a peristaltic pump and a solenoid. These two devices allow our design to automatically adjust pH and water level respectively.

Next, the Cloud subsystem is shown in "Cloud" block in Figure 2. This system's responsibility is to transfer data between a mobile software application and the sensing and control system. It also stores past sensor measurements for later analysis.

Finally, on the software application is show in the "Software Application" block in Figure 2. This is where the user primarily interacts with their hydroponic or aquaponic system. This application displays past and current sensor measurements. It also allows the user to recieve alerts if urgent issues are detected by the sensing and control system, including issues like power failure and dangerous water pH levels.

In the following sections, the design of each of these subsystems will be explored in further detail.

2 Internet of Things (IoT)

The IoT subsystem recieves its name from the design decision to use internet communication to communicate with sensors and control devices installed on aquaponics or hydroponic systems. This enables our design to leverage advantages of internet communications. One key advantage is that the vast majority of our users have internet connected devices including laptops and mobile phones. Another key advantage is the ability for growers to monitor and control their aquaponics systems from anywhere they have internet connectivity.

The IoT subsystem, in our project, refers to all hardware and software needed to communicate between sensors and output devices and the software application. See Figure 3 for an overview of our IoT subsystem design.

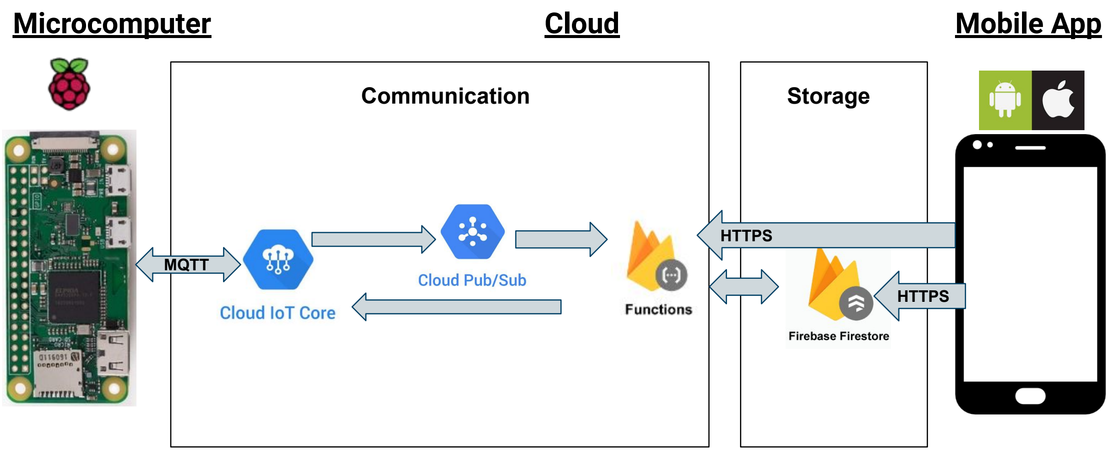

Figure 3: IoT system overview

As seen in Figure 3, our IoT subsystem has 3 major components: the microcomputer, cloud, and mobile devices. The microcomputer and cloud components will be discussed in detail within this section. In the Software Application section, the mobile app component will be discussed further.

2.1 Microcomputer

There are two key functions that the microcomputer serves in our design. The first responsibility is to receive data from sensors, and to control output devices. The second responsibility is to send sensor readings to users, and to recieve commands for output devices from users. For each of these key functions, our team has made chosen design solutions to help address these issues. Consult Table 1 for a description of the key functions and their associated technological strategies.

Table 1: Description of microcomputer key functions and design strategies

Key Function	**Req(s)**	**Design Solution**	**Reason(s)**
Communicate with sensors and output devices	F_1, F_4	Wired connection (e.g. SPI, IIC, GPIO), or wireless connection (e.g Bluetooth) depending on the sensor or control device	• Allows the flexibility to interface with many different sensors and output devices.
Communicate with user devices	F_1, F_2, F_3, F_6	WiFi internet network connection	• Low cost compared other alternatives like LTE • High accesibility as hobbyist and small-scale growers typically have a WiFi network setup within range of their system.

With these key functions in mind, our team selected a capable microcomputer. This microcomputer is the Raspberry Pi Zero W [3], shown in Figure 4.

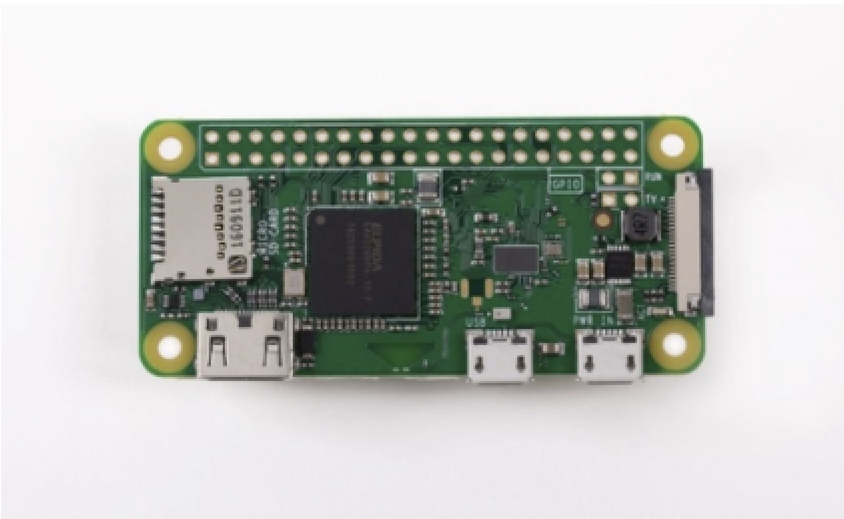

Figure 4: The Raspberry Pi Zero W microcomputer

The Raspberry Pi Zero W was chosen over other microcomputers that could perform the same key functions. This was done for the following reasons:

1. **Low cost:** costs around 40 CAD.
2. **Wireless networking:** supports both WiFi and Bluetooth.
3. **Versatile:** runs a Linux-based operating system called Raspberry Pi OS.
4. **Documentation and support:** Raspberry Pi is a popular platform with extensive tutorials, resources, and forums.
5. **Interfacing:** supports a variety of wired protocols including SPI, I2C. Also has 40 GPIO pins to interface with multiple devices.

Other notable microcomputers that were considered ESP32-based microcontrollers, and the Raspberry Pi 3B. However, the ESP32 was not chosen due to its more limited documentation and inability to run a linux-based operating system (making development slower). The Raspberry Pi 3B was not selected because of its similar functionality to the Raspberry Pi Zero W, but its higher cost of about 80 CAD.

Nevertheless, a final benefit of chosing a Raspberry Pi Zero W is that its software is compatible with other Raspberry Pi models, like the 3B. This gives design flexibility to optionally use more powerful or improved Raspberry Pi models other than the Raspberry Pi Zero W.

2.2 Cloud

The Cloud component of our IoT subsystem sits in between the microcomputer. Cloud in our project refers to cloud computing. This constitutes all the internet connected servers and associated IT infrastructure needed to communicate between the microcomputer (MCU) and the user's mobile devices. As such, the cloud is responsible for the following:

- Relaying information between the MCU and user's mobile devices
- Storing past sensor measurements
- Providing a software interface to the user's mobile devices so they can access past sensor data and communicate with the MCU.
- Scaling to allow monitoring and control of multiple aquaponics or hydroponics systems from a single mobile device.

As such, the cloud contributes to meeting the functional requirements F_1, F_2, F_3, and F_6. Please refer to the *Requirements* document for details about these requirements.

As far as design solutions are concerned, we considered 2 key alternatives based on the above specifications. The first was using IT infrastructure as a cloud proxy to communicate between the microcomputer and the mobile application. The second choice was to communicate directly with the MCU, without any infrastructure in between. These options are summarized in Figure 5.

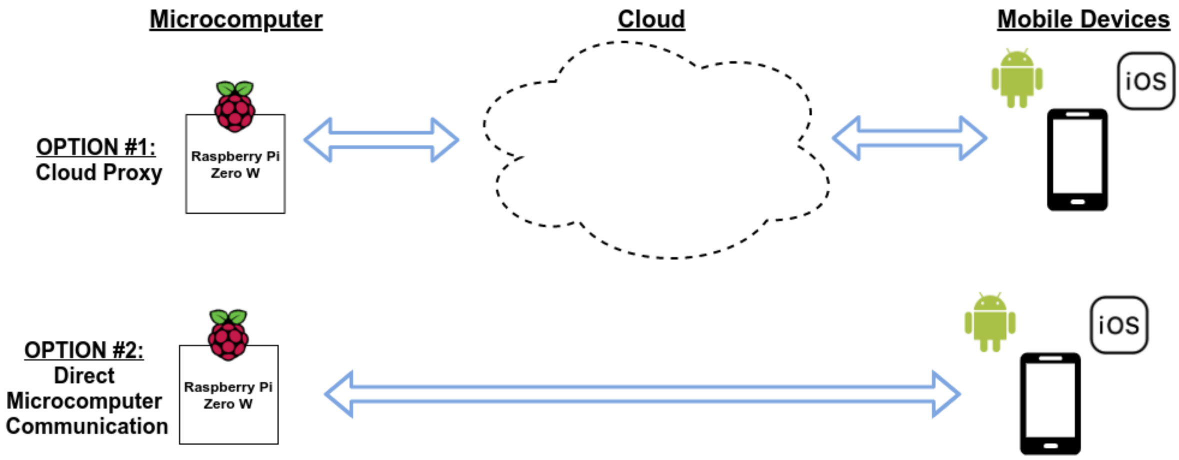

Figure 5: Two MCU to mobile application communication options.

To decide between these two main options, we weighed their strengths and weaknesses relative to their key functions. This comparison is summarized in Table 2.

Table 2: Comparison of MCU to mobile application communication options.

Function	Cloud Proxy	Direct MCU communication
Relaying information between the MCU and user's mobile devices	**Advantage:** User devices can communicate the MCU anywhere with internet connectivity.	**Advantage:** No cost for proxy service. **Disadvantage:** User devices must be on the same local network as the microcomputer.
Storing past sensor measurements	**Advantage:** Cloud proxy services can automatically provision data storage to meet unlimited demand inexpensively.	**Disadvantage:** Users must be on the same local network as the MCU to access data.
Providing a software interface to the user's mobile devices	**Advantage:** Cloud proxy services can offer APIs and pre-built tools to connect with mobile devices.	**Advantage:** communicating directly with an MCU can reduce software interface complexity and maintenance.
Scaling to allow monitoring and control of multiple aquaponics or hydroponics systems.	**Advantage:** Cloud proxy services offer automatic provisioning of resources based on demand	**Disadvantage:** limited to monitoring only systems on the same local network.

As seen in Table 2, the advantages of the cloud proxy option outweigh the direct MCU communication option. Most notably, the cloud proxy option offers more flexibility as it is not limited to being on the same private network as the MCU. It also is more capable of storing large amounts of past sensor data.

After choosing to use the cloud proxy option, we had a choice between 3rd party service providers, or managing our own cloud computing resources. We ultimately chose to use 3rd party IoT service providers over managing our own cloud computing resources for the following reasons:

1. **Low cost:** The estimated cost is less than 5 CAD/month.
2. **High implementation speed:** Service providers manage cloud computing resources, instead of developers needed to create and manage these from scratch.
3. **Easy maintainance:** Service providers typically provide web consoles for easy maintainability of resources.
4. **Automatic scalability:** Service providers automatically scale resources to match the amount of usage. For instance, more memory in a database is automatically provisioned.

Some popular 3rd party IoT service providers include Microsoft Azure, Google Cloud IoT, and Amazon Web Services (AWS) IoT. These service providers offer very similar services at comparable prices. So, our team decided to develop using Google Cloud IoT because it offered a better software interface to our mobile application.

This software interface will be discussed in further detail in the following sections. In addition, the architecture of our cloud system will be described in terms of two aspects: cloud communication and cloud storage.

2.2.1 Cloud Communication

Referring back to Figure 3, we see that the cloud communication component transfers data between the MCU and cloud storage. These components therefore help satisfy the functional requirements F_1, F_2, and F_3, which require communication between the mobile application and the sensing and control system.

From Figure 3, we also see that the cloud commmunication part is made of 2 key components which are Google Cloud services. These are Google IoT Core, Google Cloud Pub/Sub, and Google Firebase Cloud Functions. See Table 3 below to see the role of each of these services in our design.

Table 3: Roles of Google Cloud communication services used.

Google Cloud Service	Role
IoT Core	• Provides an API for MCUs to communicate with Google Cloud using the MQTT protocol. • Provides MCU authentication and data encryption using public/private RSA keys
Pub/Sub	• Automatically recieves incoming messages from Google Cloud IoT core. • Allows Google Firebase Cloud Functions to respond to each message that is published by an MCU.
Firebase Functions	• Provides the ability to write functions that run on Google managed servers. • Writes MCU messages (with sensor data) in databases. These messages can then be retrieved by the mobile application. • Sends commands to MCUs through Google IoT Core.

Lastly, note that the communication protocol used between the MCU and Google Cloud IoT Core is MQTT. This was chosen over other possible protocols like HTTPs as it uses lower bandwidth and has less software overhead [4]. These two factors decrease the power consumption and data usage of our design.

2.2.2 Cloud Storage

Referring back to Figure 3, we see that the cloud storage components saves messages from both the MCU and the mobile application.

The key requirement satisfied by this section is F_2, to be able to store past sensor measurements for analysis. However, like the previous section, the satisfy the functional requirements F_1, and F_3, which require communication between the mobile application and the sensing and control system.

Table 4 also shows another Google services that are used in our design: Google

Firebase Firestore.

Table 4: Roles of Google data storage services used.

Google Service	**Role**
Firebase Firestore	• Stores sensor measurements in a NoSQL format. • Stores MCU configurations.

Google Firestore is used for storage of past sensor measurements. This is because Google Firestore charges a cheaper rate for storage than other Google databases, making it preferable for storing large amounts of sensor measurement data [5]. It allows features a subscription model where new sensor readings are updated within our app as soon as they are stored in Firestore [6]. This allows us to display most recent sensor updates in near real-time.

Google Firestore also stores MCU configurations. When these configurations are updated, Google Cloud Functions automatically updates a device configuration. This is used to send commands to the sensing and control system from the mobile application. This design was chosen as it allows the mobile application to both read the current MCU configuration, and change it. That way, the mobile application can make configuration updates based on a MCU's current status. For instance, if the pH level of a given MCU configuration is below the target value, it could be changed to a higher value by updating that MCU's configuration.

2.2.3 Cloud System Implementation

As mentioned in section 2.2.1 above, Google Firebase Cloud Functions are used to save sensor messages, and connect IoT devices to the mobile application. So, just as depicted in Figure 3, Google Firebase Cloud Functions are central to the cloud system and define much of its functionality.

The implementation of this project's Google Firebase Cloud Functions can be found in the piponic_cloud repository (`https://github.com/jaydenleong/piponic_cloud`). The README of this repository also provides further documentation about how the overall cloud system can be setup and maintained.

3 Power

Our system is comprised of various electric components, including the Raspberry Pi microcomputer, sensors, and output devices. Power for the system will be available from two sources, a main, and a backup. The main power source is 12V DC. This is supplied by a commercially available power supply that rectifies and steps-down voltage coming from a 120V wall outlet. This will provide power to the sensor hub during normal operation. That is, when the device is plugged in, and the line-voltage is available.

3.1 Back-up Power

The backup power will be provided by a battery pack with 2 AA batteries, with voltage boosted to 5 V from a DC-DC boost converter. The arrangement of these two inputs can be seen in figure 6. 3 AA batteries could also be used in series to produce a voltage of 4.5V, which will increase the lifespan of the back-up power system and the efficiency of the boost-converter chip. If this arrangement of batteries is used, a voltage divider must be added to the battery-voltage monitoring circuit, so that the maximum voltage of 4.5V is read at max 3.3V. We recommend using 100k and 36.9k resistors for this voltage divider to minimize power loss.

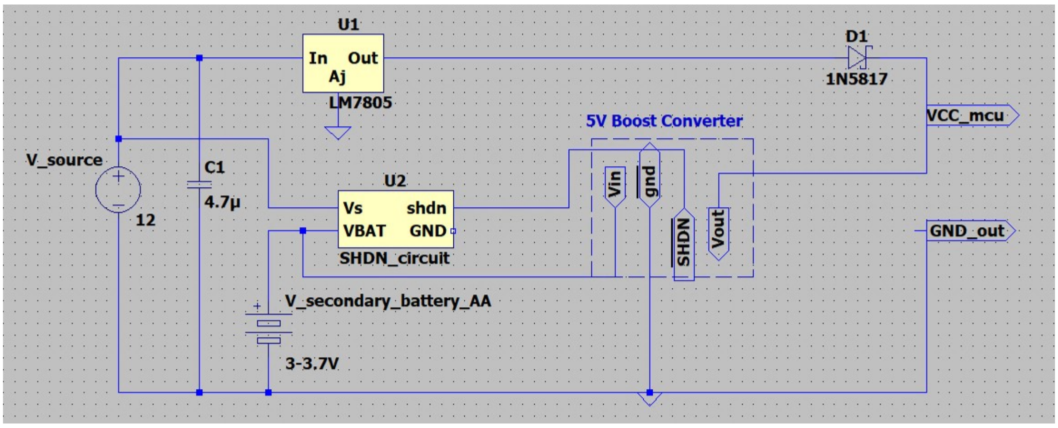

Figure 6: Uninterruptible power supply architecture.

The LM7805 voltage regulators are responsible for delivering all of the current used by the 5V devices during normal operation. It has been observed that all of the 5V devices consume around 200mA during normal operation, and the voltage regulators are rated to 1A. To increase the safety margin associated with heat-burnout, a 15g aluminum TO-220 heat sink was added to the LM7805 chip. Two LM7805 regulators are used in parallel to ensure adequate power is always available. This was also added to the Schottky diode at the output of the LM7805 chip.

To conserve the battery-lifespan, the boost converter is turned off automatically when the 12V line voltage is present. The circuit that accomplishes this can be seen in figure 7 - a simple BJT switch application. In the event of a power failure, the system will be able to run for 8 hours on backup power. This functionality is intended as a way to keep monitoring the condition of the aquaponic or hydroponic systems in the event of a power failure.

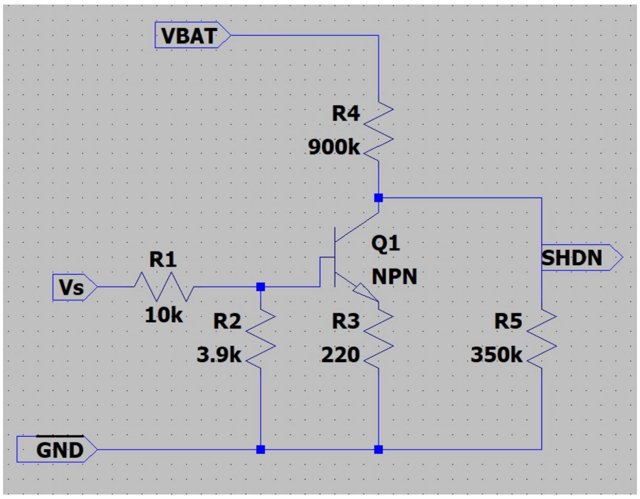

Figure 7: Boost converter shutdown circuit. This circuit disables the AA batteries when 12V is present.

For boosting 2 AA batteries up to 5 V, the Pololu 5V Step-Up Voltage Regulator U1V11F5 is a high quality, low cost, easily accessible option that can be implemented by hobbyists with minimal difficulty. It can take anywhere between 0.5 and 5 V input and will automatically boost it to 5.02 V output.

3.2 Power Budget

Our monitoring system consumes on average 810 mW when on back-up power. Most of this comes from the operation of the raspberry pi. A breakdown of the average power use can be seen in figure 8.

The control devices and relays operate only when line power is available (that is, when not on back-up power), and when all control devices are active the system consumes an additional 5W.

3.3 Current Protection

The output voltage will be connected via Schottky diode to the line voltage in, before being connected to the Raspberry Pi. This will effectively OR the voltage source seen by the Raspberry Pi. Several other circuit topologies were simulated and tested, and the Schottky diode topology represents the best functionality in both

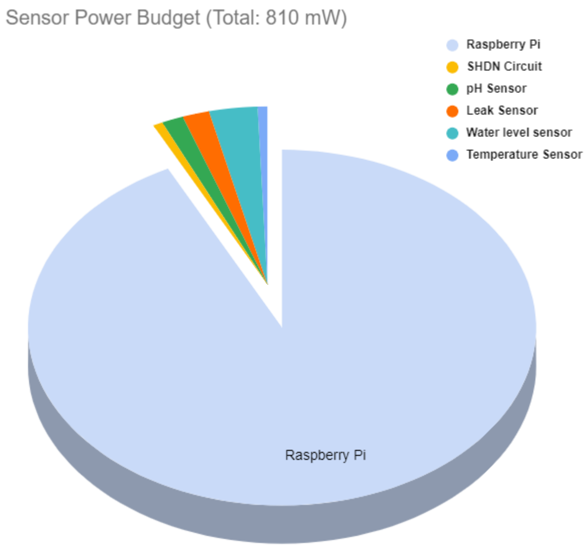

Figure 8: Average monitoring system power breakdown.

Table 5: Power current protection device comparison.

Device Type	Current Protection	Power Loss	Line Voltage	Backup Voltage
N Mosfet	No	2 mW	5 V	5 V
P Mosfet	Yes	2 mW	5 V	4 V
Schottky Diode	Yes	60 mW	4.7 V	5 V

normal operation and backup power mode. The other topologies that were modelled prior to making the design decision were a P-channel mosfet or an N-channel mosfet.

The N-channel mosfet does not achieve the desired functionality of 1-way current protection, so this solution is disregarded. The P-Channel mosfet does an excellent job minimizing power loss, however in the back-up power scenario, drains an unacceptable amount of current such that the line voltage will be 4V, which is too low for our applications. Therefore, the best solution is the schottky diode, despite its relatively high power consumption, because it keeps the line voltage at an acceptable level in both operating conditions.

This system is intended to be used with a LiPo battery pack as a backup battery source, however due to safety considerations, the implementation of this will be limited to a recommendation and design only. If one is using LiPo batteries to provide backup power, please familiarize yourself with the appropriate safety procedures associated with using LiPo technology.

For keeping a LiPo battery fully charged while plugged in, and boosting the voltage a 3.7V LiPo cell to 5V when using, the PowerBoost 1000 Charger by Adafruit is our recommended breakout board. It supports load-sharing which means the board can supply power to the hub at the same time as charge the battery. It accomplishes this with 2 special ASICs. The input line voltage can be directly input into the PowerBoost 1000, which will automatically manage OR-ing the power for the Raspberry Pi.

4 Sensors

Our design is centered around 4 types of sensors: Temperature, pH, water-leak, and water level. These sensors focus on the the status of the water in the system because this will effectively address the needs of fish, plants, and microbes, as water is the media through which these species interact.

4.1 Sensor Type Selection

There are many distinct indicators of ecosystem health when monitoring a hydroponic or aquaponic system [7]. There's aqueous concentration of carbon dioxide and oxygen. There's accumulation of bio-solids from fish waste in the water. There are three different important types of dissolved nitrogen: nitrate, nitrite, and ammonium. The alkalinity and pH of the water are both very important as well. The water level, flow-rate, and temperature are not to be overlooked either.

Implementing a system that monitors all of these important summary variables is possible, however would require significant investment of both time and money. Limited as we are in both time and money, we decided to focus on the most important, the lowest cost, and the easiest to implement sensors.

By building our project with simple sensors, we make our design accessible to as many people as possible. Our sensor-framework can easily be expanded upon by future developers or users, who can choose to purchase more expensive sensors if they wish to.

We identified 4 types of sensors that fit the above criteria

1. pH
2. Temperature
3. Water-level
4. Water-leak

The other sensor types were either too expensive or did not add enough valuable information to the monitoring system for them to be considered.

4.2 Analog to Digital Conversion

The Raspberry Pi does not have an analog to digital converter (ADC) built in. Several of the sensors that are being used, and that future designers could use will require the use of an ADC, so we added one to our project. The ADS1115 is a 16 bit, 4 channel, programmable gain, I2C ready ADC, which can operate between 3.3 and 5 volts. These specifications meet the needs of this project by maximizing resolution, minimizing pin use, minimizing power consumption, and having sufficient channels. The ADS1115 was designed to be a very power-efficient board by

Currently 4 of the ADC channels are being used, leaving 4 free for new features. The current channels in use are:

- External Leak Sensors
- Enclosure Internal Leak Sensors
- pH Sensor
- Battery voltage

The pins of the Raspberry Pi are designed to operate at 3.3 V, however have large enough pull-up and pull-down resistors that they can handle 5V as well. To maximize longevity, and minimize chance of pin burnout, the ADS1115 ADC is operated at 3.3V.

By operating the ADC at 3.3V, the maximum voltage it can read from one of its input channels is now 3.3V also. If any component cannot run at 3.3V, then it must be reduced to 3.3V from it's higher voltage (such as 5V) so as to not damage the ADC. This can be accomplished with a voltage divider.

4.3 Temperature

Temperature is important in aquaponic or hydroponic growing systems because different species all have their own preferred temperature range. If a user is growing multiple species in a single system, they may wish to control the temperature so that all of the species they are growing are within the functional temperature limits.

We considered three types of temperature sensor. We chose DS18B21 ($10) temperature sensor over the DS18B20($5) and I2C BME280($20) temperature sensor. This decision was made because:

- The DS18B21 provides temperature values every minute and allows the Implementation of a system that can display values in real time. Requirement: F_1

- The DS18B21 is waterproof and the other alternatives are not. Choosing this sensor leads to a system that is water resistant. Requirement: NF_4
- The DS18B21 is in the mid price range and contributes to an affordable system. Requirement: NF_1

The temperature sensor communicates over 1-wire interface with the Raspberry Pi. The Raspberry pi operating system has convenient protocols for 1-wire interfaces that automatically receive the data from these devices and store them in a given location in memory. The script that reads the temperature, quite simply, checks this location in the memory for the temperature, and performs some simple cleanup.

4.4 pH

In hydroponic and aqauponic systems, the pH has been described by experts as the *Master Variable* [7]. This means that it controls or influences many metabolic processes. It is essential that the pH of the system does not go outside of the maximum or minimum tolerable limits of the organisms present in the growers system.

For these reasons, we deemed the pH to be an essential part of our monitoring solution.

We considered many types of pH sensors among many and initially chose the PH-4502C($39) over the LeoEc($60) because:

- The PH-4502C is a low power sensor (less than 0.5W) and will therefore provide better lifespan in the event of a power failure. Requirement: F_5
- The PH-4502C provides pH values with a delay of 0.5s. Requirement: F_1
- The PH-4502C is lower priced compared to the LeoEc and middle priced compared to other pH sensors that don't meet our requirements. This choice contributes to an affordable system. Requirement: NF_1

The PH-4502C is intended to be used by mapping pH 7 to 2.5V, 14 to 0V, and pH 0 to 5V. These values rarely seem to be present in reality, necessitating significant calibration.

This pH sensor performed poorly in testing, and so another type of pH sensor was purchased; the SEN0161-V2 from DFRobot. This electrolytic pH probe works on the same principle of the one above but has a different amplification board that does not have any mechanical calibration requirements. It also meets all of the other requirements outlined above.

Figure 9: pH sensor Kit

4.5 Water Level

Measuring the water level enables growers to ensure that they have adequate water in their system for their plants and fish. If the water level is too low the system can heat-up quickly or change in temperature or chemical concentration. The water level is measured with the intention of automating the water refilling process.

There are different ways to measure the water level. We chose the Binary Water Level Detector, SEN0204-ND ($13). The other alternatives are the Ultrasonic Sensor, HC-SR04 ($4) and the E-tape Liquid Level Sensor ($60).

We made this decision because:

- The SEN0204-ND provides the minimum data to maintain a desired water-level F_1
- The SEN0204-ND is small and can scale very well to differently sized systems. This makes the sensor easier to install on many more types of aquaponics systems compared to alternatives. NF_5
- The SEN0204-ND is low cost. The HC-SR04 and other Ultrasonic Sensors have a very short lifespan and break very easily, especially in environments

where water is involved. So in the long term, the SEN0204-ND contributes the most to an affordable system as it is IP67 rated. NF_1, NF_4

4.6 Water Leak Sensor

For indoor aquaponic or hydroponic systems a leak may be a catastrophic failure that could cause significant property damage or could result in eviction of the grower(s). Detecting leaks, is therefore of paramount importance to ensuring long-term social sustainability of aquaponic or hydroponic systems in urban environments.

Leak sensors could be implemented in many ways. We chose a very low cost hobbyist leak sensor available online as seen in figure 10.

In our design, this sensor is also placed inside the enclosure to detect if any of the system electronics may be compromised.

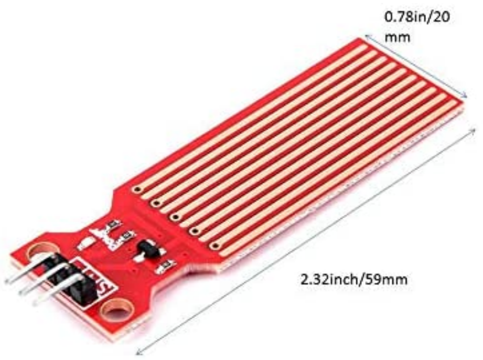

Figure 10: Water leak sensor

We made this decision because:

- The leak sensor provides the minimum data to detect a leak, and can be read continuously by the ADC. F_1, F_3
- This sensor is low cost. NF_1, NF_4

5 Control Systems

We considered 3 types of output devices: Air pump to oxygenate the water, Water pump to control the water levels, and the pH adjuster to control the pH levels

5.1 Air Pump

Introducing oxygen to a system can be done by creating movement in the water. This is normally done by having an air pump create movement within the tank. The problem arises due to the scope of our project. The air pump needs to work for systems of up to 3000 gallons of water. There is too much variability in the size of the system and introducing an air pump complicates this control system greatly. We have decided not to include this within the scope of our project.

5.2 Water Pump

We considered having a simple control system that would automatically keep the desired water level with the use of water pumps. Once we focused on the scope of our project, a water pump is not ideal being controlled through the RaspberryPi due to 2 reasons. The first reason is that the scope of our project is to design for systems up to 3000 gallons of water. Installing a water pump system suitable for a RaspberryPi would be unreasonable due to the relatively small pumps available through the RaspberryPi. The second reason is if we were to install a reasonably size pump, this would take far too much power for our entire system to work in parallel with this pump.

5.3 pH Adjuster

There are two ways to change and maintain the pH levels of the system: with a liquid solution or a solid object usually peat moss or driftwood. We have chosen to use the liquid solution and have decided to use a Peristaltic Pump ($12) to control this liquid solution due to the following reasons:

- The solids can produce discolouration in the water. This will not only make physical observation harder, a feature we might implement is a waste detecting system, and discolouration affects that. The liquid solution will not interfere with the appearance of the system and provide accurate values in real time. (F_4)
- The liquid solution is a lot easier to measure. The volume of a liquid is much easier to measure and transport compared to something solid. This results in a much easier installation. (NF_2, NF_5, NF_7)
- The Peristaltic Pump is cheaper and easier to design compared to something that needs to transport a solid. This leads to an affordable system (NF_1, NF_2)
- There are many types of liquid solutions that can be used while the solids needed are very specific and usually hard to find. This allows users to customize more easily compared to the solids. (NF_5, NF_7)

The peristaltic pump will pump a basic solution into the aquaponic or hydroponic system to raise its pH, because in most aquaponic system, the bacteria acidifies the water. We recommend that the solution to be pumped into the system is potassium hydroxide and calcium hydroxide in an alternating manner to provide some micronutrients as well as pH buffering.

6 Software Application

To implement the software application to display the aquaponic's data to the user, we considered various aspects, like the technology we were going to use, how we were going to implement said technology, and the trade-offs we made for each decision.

6.1 Technology

The two main technologies we considered to implement the software application were web and mobile. We ultimately choose to use a mobile based approach to implement our software application than a web based approach. The main factor that that determined this decision was ease of use. In a mobile based approach users would be able to quickly access the application, as user preferences are saved and tied in with the mobile application. This means that the user doesn't have to re-authenticate time after time when trying to access their aquaponics data. Another big factor we considered when choosing a software approach was accessibility. Almost everyone has their mobile devices on them at any given time during the day, which would provide greater accessibility to the application. A final consideration that we explored was push notifications. With a mobile based approach, we are able to send push notifications to users when extremely urgent matter arises, whereas in a web based approach this isn't even an option.

6.2 Implementation

We considered many different options we could take to create a mobile application. We debated on whether or not we would want to create a native application or build one with a cross platform language. Given the time and nature of this project, we ultimately decided on a cross platform approach. With a cross platform approach, we would be able to develop for both Android and IOS with half the effort and half the code base. One of the draw backs however is that we might lose niche native functionality specific to either Android or IOS. Since our software application is simple, we believe that the niche native service loss wouldn't be detrimental to the project. We choose Flutter as the our cross-platform language to implement this mobile application over other cross platform languages such as React Nastive, or Xamarin since Flutter allows for quick implementation of native feeling applications.

6.3 Use Case Overview

Upon entering the application, the application checks if the user has previously been authenticated. Here users can authenticate themselves by either signing up or signing in. Upon successful sign in or creation of an account, the user would be taken to the application's home screen. The ability to authenticate users to access to their account and aquaponioc system meets our non-functional requirement NF_2.

The home screen displays a summary of one of the existing systems that the user has registered with the application. The summary includes the name of the system, the status of the system, last time the information was updated, the warnings for the system, and a list of all the sensors and it's measured values. The summary of the aquaponic system on the home screen meets our function requirement F_1. From the home screen, the user is able to navigate to the sensor chart screen, the aquaponic system settings screen, and the user setting screen.

The sensor chart screen displays historic measurements made by the sensors on a chart. The user is able to select which sensor to view and the date range in which the measurements were taken. The ability to view historic sensor measurement within a given date range meets our function requirement F_2.

The system settings screen allows the user to set the range of acceptable values for the aquaponic sensors and well as calibrate the pH sensor. The user can also change the sensor measurement interval on this screen. The ability to adjust and calibrate sensors meets our functional requirement F_3.

The user setting screen is where the user is able to logout or delete their account if they so choose. In the user setting screen, the user is also able to add additional aquaponic systems to their account provided they know the aquaponic systems identification. The ability to add and monitor additional aquaponic systems meets our non-functional requirement NF_5.

If the user is already logged in the application, the user may receive notifications if one or more of their aquaponic system is in a critical state. Upon tapping the application, the user will be brought to the home page with the summary of the system in the critical state. The ability to send alerts to growers when a aquaponic system enters a critical state fulfills our function requirement F_3.

An overview of how the user can interact with the software application is highlighted and illustrated in the use case diagram below in figure 9.

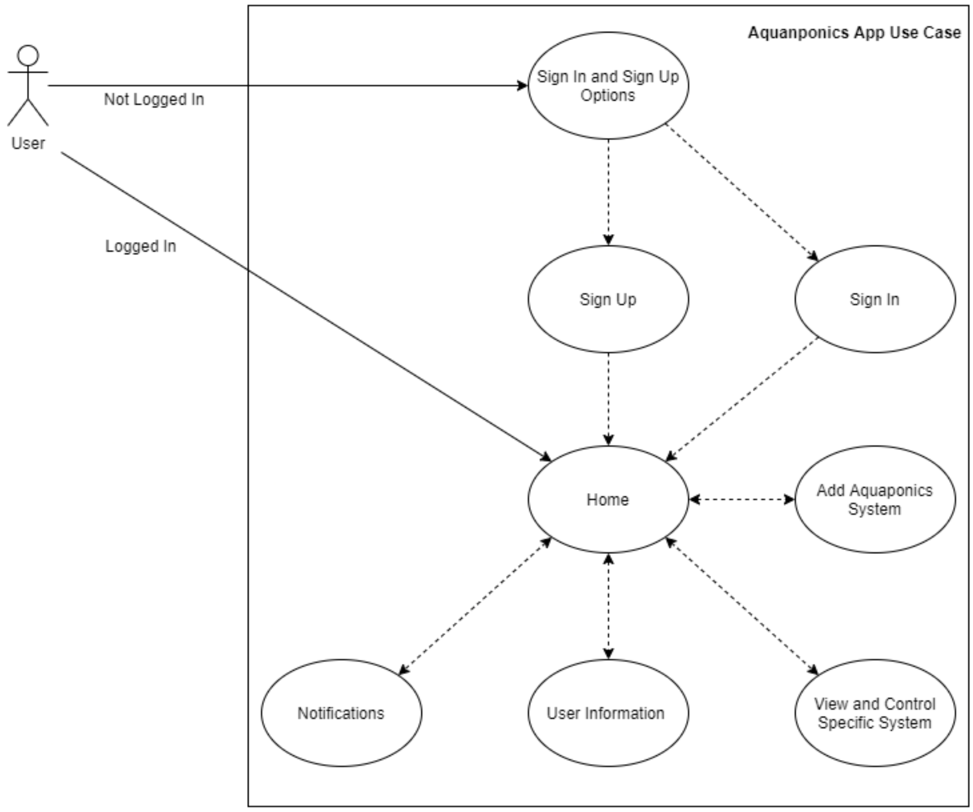

Figure 11: Use Case Diagram

6.4 Screens

Below are the different screens in the software application and an explanation of their intended user interactions as well as how they interact with the overall IOT infrastructure.

6.4.1 Sign In and Sign Up Options Screen

The sign in and sign up options screen will be presented to users who aren't previously authenticated on the application. The sign in and sign up options screen will give the user the option to either Sign Up an account or login to an existing account. The user can Sign Up with the application by providing an email and password or by Signing in using a valid Google account. An image of the application's sign in and sign up options screen can be seen below in figure 12.

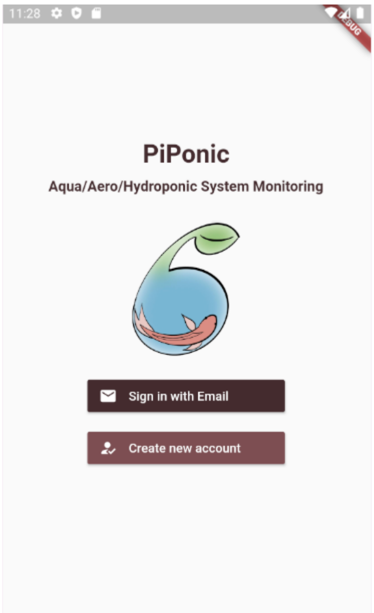

Figure 12: Sign In and Sign Up Options Screen

6.4.2 Sign In Screen

The sign in screen is presented when the user taps on the "Sign in using Email" option in the sign in and sign up options screen. This page contains two input fields for the email address and password. Unsuccessful authentication submissions won't let the user into the application whereas successful authentication submission will direct users to the application's home screen. Each login attempt will trigger a request to Firebase to confirm the users credentials and to get the user's unique User ID. The email input box verifies that it's a valid email address by checking if the '@' character is present within the email submission. Similarly password input box verifies that the password is sufficient by the password length is at least 6 characters. An image of the sign in screen can be seen below in figure 13.

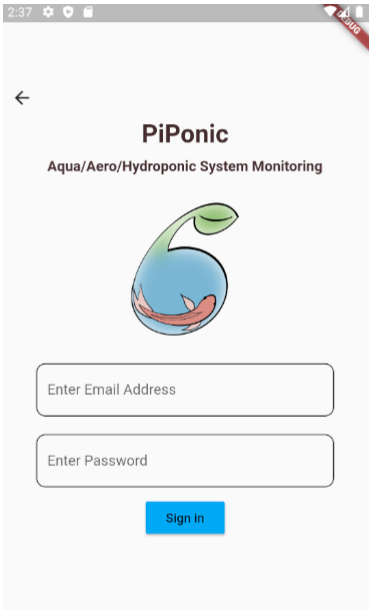

Figure 13: Sign In Screen

6.4.3 Sign Up Screen

The sign up screen is presented when the user taps on the "Sign up with Email" option in the sign in and sign up options screen. This page contains two input fields for the email address and password. Unsuccessful sign up submissions won't let the user into the application whereas successful sign up submission will direct users to the application's home screen. Each login attempt will trigger a request to Firebase to create the user account by using the submitted user credentials and to get the user's unique User ID. The email input box verifies that it's a valid email address by checking if the '@' character is present within the email submission. Similarly password input box verifies that the password is sufficient by the password length is at least 6 characters. An image of the sign up screen can be seen below in figure 14.

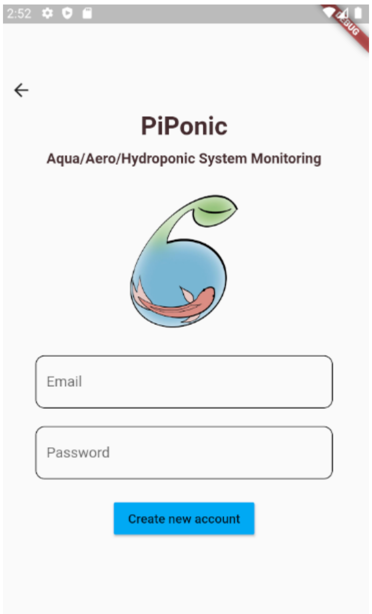

Figure 14: Sign Up Screen

6.4.4 Home Screen

The home screen is presented when the user successfully authenticates with the application. If there are existing aquaponic system(s) associated with the users account, the home screen will show a summary of one of the existing systems. The home page would show a summary of the chosen system including the name of the system, the status of the system, last time the information was updated, the warnings for the system, and a list of all the sensors and it's measured values. The values are of the aquaponic system is directly tied to the Firebase backend; this means that the values on the home page will update in real time whenever a there is a new reading in the Firebase backed. Am image of the home page can be seen below in figure 16.

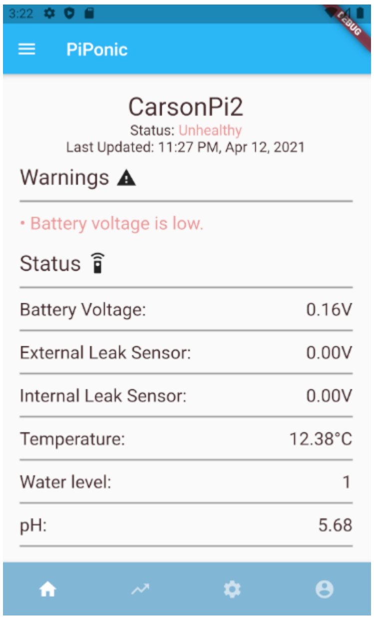

Figure 15: PiPonic home screen

6.4.5 Aquaponic/Hydroponic List

Tapping on the hamburger menu button on the top left of the application, the list of the user's aquaponic systems will slide out. Here the user is able to switch between the different aquaponic systems associated with their account. The list of the aquaponic systems is directly tied to the Firebase backend; this means that the list of aquaponic systems will change in real time whenever a there is a change to the users access to different aquapnic systems.An image with the location of the hamburger menu button and the list of the user's aquaponic systems is shown below in figure 14.

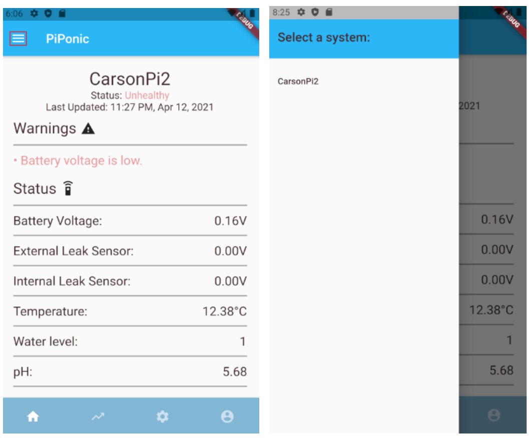

Figure 16: Location of the hamburger menu button and the list of the user's aquaponic/hydroponic systems

6.4.6 Chart Screen

Tapping on the rising trend icon in the bottom navigation bar brings the user to the chart screen where the user is able to see a chart of their sensor values and the time at which they were measured. Here the user is able to change the different sensors that they want to see and are also able to change the range of time for which the measurements were taken. New measurements are queried from the Firebase backend and the graph is changed whenever a user changes a sensor to view or changes a time range. A screenshot of the charts screen can be seen below in figure 15.

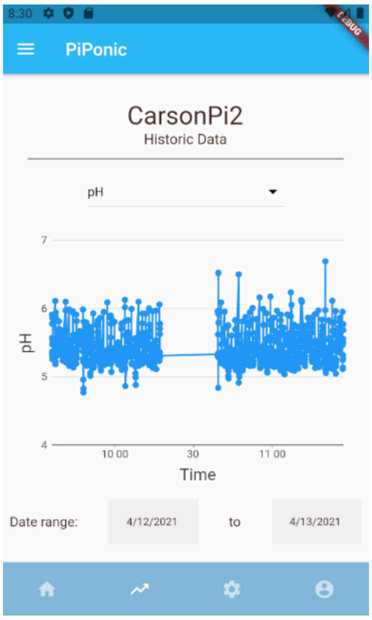

Figure 17: PiPonic Chart Screen

6.4.7 System Settings Screen

Tapping on the gear icon in the bottom navigation bar brings the user to the aquaponic system setting page. Here the user can set the range of acceptable values for the aquaponic sensors and well as calibrate the pH sensor. The user can also change the sensor measurement interval in the "General" tab. When the user enters a new acceptable range or calibrates the pH sensor or changes the sensor measurement interval, the application will update the backend which would then update the system of the new changes. A screenshot of the aquaponic settings screen can be seen below in figure 16.

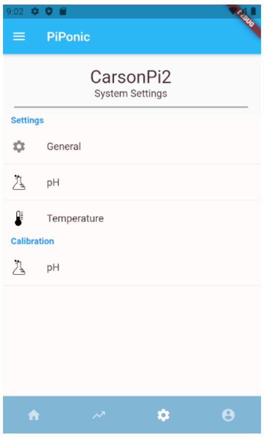

Figure 18: PiPonic system settings screen

6.5 User Settings Screen

Tapping on the user icon in the bottom navigation bar brings the user to the user settings page. Here the user can add additional aquaponic systems to their account to monitor and control. When the user adds additional aquaponic systems to their account, the application would then notify the backend which would then associate the new aquaponic system with the users identification. The application would then automatically add the new aquaponic system in the users application. The user can also logout of their account and delete their account. A screenshot of the user settings screen can be seen below in figure 17.

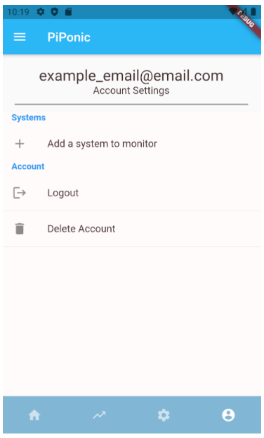

Figure 19: PiPonic user settings screen

6.5.1 User Information Screen

The notification screen is presented when the user clicks on the user icon in the bottom navigation bar. The user screen shows the user information and allows for the user to log out of the application or delete their account.

6.5.2 Add Aquaponics Screen

The add aquaponics screen is presented when the user clicks the profile icon in the bottom navigation bar. The add aquaponics screen contains input fields that pertains to the aquqponic system they are trying to add. Upon successful submission of results, the aquaponic system will be created in the Firebase back-end and linked with the users User ID.

7 Circuit Integration - PCB

In order to maximize simplicity and repeatability of construction, robustness, and lower production costs, a custom printed circuit board was designed. This element integrates several circuit elements, including the power supply architecture, reading from sensors, sending control signals to the relay module, and interfacing with the Raspberry Pi. The PCB design can be seen in figure 20.

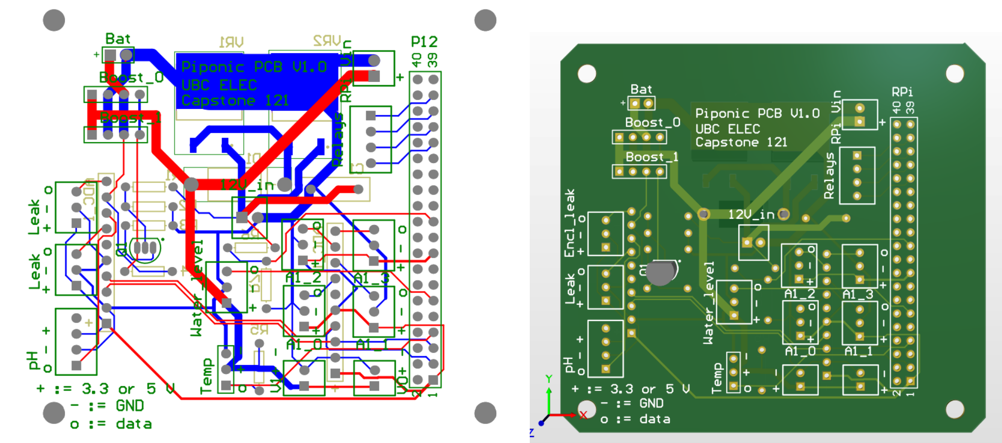

Figure 20: Custom PCB Layout

The integration with the raspberry pi is accomplished by having this PCB take the format of a 'hat'. This means that the board will directly mount onto the Raspberry Pi's male header pins, so that no mistakes can by made by the user in wiring.

All sensors will integrate with the PCB through simple board-mounted screw-down terminal blocks. Although this adds to the size of the final design, it is the simplest for users.

7.1 Component Choices

The components for this PCB were chosen to make manufacturing without a solder-reflow station easy - so almost all of the components are through-hole. The only exception to this are the voltage regulators, which are chosen as SMT components for the built-in heat-dissipation, and for the low profile allowing them to be mounted to the underside of PCB, which will allow for the PCB to have a much smaller footprint.

7.2 Future Development Considerations

This PCB was designed with future development in mind. It features 8 total ADC channels, 4 of which are unused. These ADC channels can be used for additional sensors. It has 2 unused relay-channels for additional control devices. All of the Raspberry Pi header pins are still accessible with this design, so that any type of device or communication channel may be added through jumper wires by future users.

8 Assembly

In consideration of the construction of the system, the NF_4, NF_1 requirements were chiefly considered. That is, that the design is waterproof and low-cost.

All components were roughly modelled in Solidworks 2019 to gauge how large the enclosure needed to be. Models of all necessary components can be seen in figure 21 and 22.

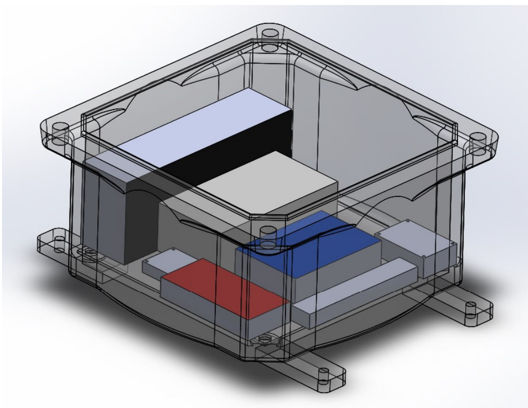

Figure 21: Enclosure elements inside enclosure: Power Supply, Breadboard, Relay Block, Terminal Block, Raspberry Pi, Water-level board, pH sensor board.

There are 5 independent wire ports that must penetrate the enclosure in a waterproof

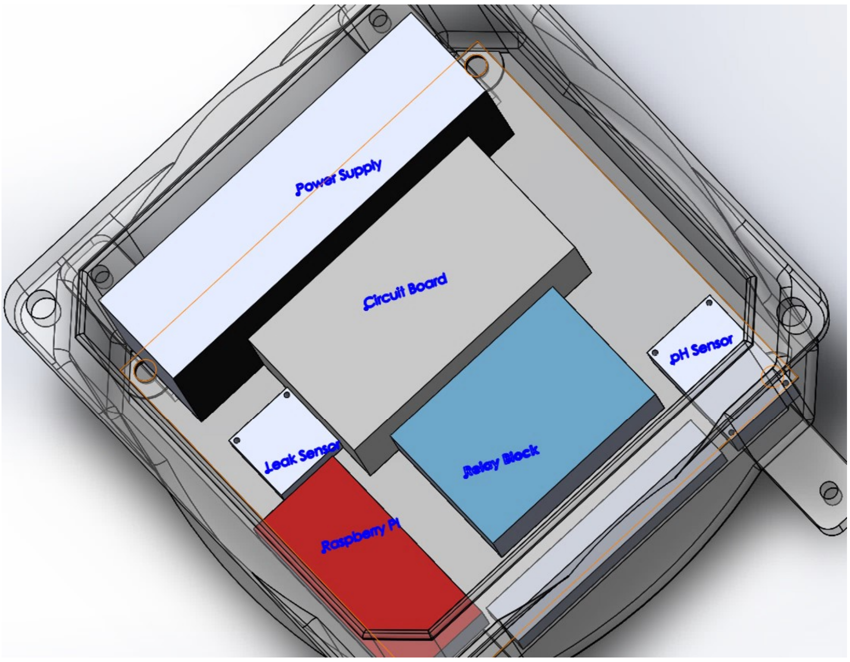

Figure 22: Labelled Enclosure elements: Power Supply, Breadboard, Relay Block, Terminal Block, Raspberry Pi, Water-level board, pH sensor board.

manner. To accomplish this, waterproof nylon cable glands were used. An example of one of these can be seen in figure 23. These devices have a rubber seal which is compressed around a wire inside it, forming a seal.

Figure 23: Waterproof Nylon Cable Gland example.

For ease of maintenance and durability, all of the electronic components are mounted to a board that are mounted to the corners of the main enclosure with double-sided tape. This board is constructed from 1/2” plywood.

9 Bill of Materials

In order to reconstruct this project, one will need to purchase many components. The sources for these are outlined in this section and in the Installation Instructions document.

9.1 Hardware

The hardware BOM can be seen in the table below.

Designator	Description	Quantity	Link
Enclosure	8x8x4' junction box	1	`https://www.homedepot.ca/product/carlon-thermoplastic-junction-box-8x8x4-in/1000403713`
Heatshrink	Variety Pack	1	`https://leeselectronic.com/en/product/15370-kit-heatshrink-variety-pack-170pcs.html`
Cable Gland	7mm	3	`https://leeselectronic.com/en/product/61270-fitting-water-proof-pg-7-light-grey-csa-bi html`
Cable Gland	9mm	6	`https://leeselectronic.com/en/product/6125-fitting-water-proof-pg-9-black.html`
Screws/Bolts	4x1/2", 6x1/2"	Box	

Table 6: Hardware BOM required for building project

9.2 Electronics

9.2.1 Sensor Devices

Designator	Description	Quantity	Link
Temperature	DS18B20	1	`https://www.digikey.ca/en/products/detail/dfrobot/DFR0198/7597054`
Water Level	SEN0204	1	`https://www.digikey.ca/en/products/detail/dfrobot/SEN0204/6579443?s=N4IgTCBcDaIMoFEByAGMKAsIC6BfIA`
pH Sensor	SEN0161-V2	1	`https://www.digikey.ca/en/products/detail/dfrobot/SEN0161-V2/9608223`
Leak Sensor	Analog Leak Sensor	1	`https://www.amazon.ca/gp/product/B07PV9SYLV/ref=ppx_yo_dt_b_asin_title_o03_s00?ie=UTF8&psc=1`

Table 7: Sensor Bill of Materials

9.2.2 Control Devices

Some devices will be necessary to set up the control system aspect. This includes what is seen below in the table BOM. The design currently has 2 unused relay channels that may be used in future expansion.

Designator	Description	Quantity	Link
Relay Module	4 Channel 5V controlled Relay Module	1	https://leeselectronic.com/en/product/31302-4-relay-digital-module-5v.html
Solenoid Valve	12V 3/4" solenoid for connecting with hoses	1	https://leeselectronic.com/en/product/440-44012VSOLENOIDVALVE34SMALLROB104.html
Peristaltic Pump	12V Peristaltic Pump with surgical tubing	1-2	https://www.amazon.ca/Peristaltic-Liquid-Dosing-Silicone-Tubing/dp/B075VN1QZM/ref=asc_df_B075VN1QZM/?tag=googleshopc0c-20&linkCode=df0&hvadid=292901727312&hvpos=&hvnetw=g&hvrand=16446818779147696161&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=9001561&hvtargid=pla-376008502070&psc=1

Power Supply	12V 2A Power Supply	1	https://www.amazon.ca/Adapter-Supply-Switching-100-240V-Transformer/dp/B0711SZCMQ/ref=sxin_9_ac_d_rm?ac_md=2-2-MTJ2IDJhIHBvd2VyIHN1cHBseQ%3D%3D-ac_d_rm&cv_ct_cx=12V%2Bpower%2Bsupply&dchild=1&keywords=12V%2Bpower%2Bsupply&pd_rd_i=B0711SZCMQ&pd_rd_r=1eff5954-32d8-4e80-8994-f3f9e1651a94&pd_rd_w=vjLDB&pd_rd_wg=riHjN&pf_rd_p=cf70c38a-5522-42af-aeb2-119882942fe1&pf_rd_r=Z0P28K1ZC0Q2NJ9QT85Q&qid=1618245824&sr=1-3-12d4272d-8adb-4121-8624-135149aa9081&th=1 or https://www.amazon.ca/ALITOVE-Universal-Regulated-Switching-Transforme dp/B078RY6YY3/ref=sr_1_15?dchild=1&keywords=12V+power+supply&qid=1618245824&sr=8-15

Table 8: Control Devices Bill of Materials

9.2.3 Electronic Parts for PCB

Designator	Description	Quantity	Link
12V_in	Power 12V in	1	https://www.digikey.ca/en/products/detail/on-shore-technology-inc/OSTVN02A150/1588862
A1_0, A1_1, A1_2, A1_3	Header, 3-pin	4	https://www.digikey.ca/en/products/detail/on-shore-technology-inc/OSTVN03A150/1588863
ADC_0, ADC_1	Analog to digital converter	2	https://www.digikey.ca/en/products/detail/adafruit-industries-llc/1085/5761229
Bat	2AA Battery Holder	1	https://www.digikey.ca/en/products/detail/hammond-manufacturing/BH2AAW/3869832
Boost_0, Boost_1	5V Boost converter	2	https://www.pololu.com/product/2562
C1 (4.7uF)	Capacitor	1	https://www.digikey.ca/en/products/detail/tdk-corporation/FA20X7S2A475KRU00/7384281

D1	Power Schottky rectifier, 60 V, 5 A	1	https://www.digikey.ca/en/products/detail/smc-diode-solutions/95SQ015/8021507
Leak Sensor Header	Header, 3-Pin	2	https://www.digikey.ca/en/products/detail/on-shore-technology-inc/OSTVN03A150/1588863
Header 20X2	GPIO shield for RPi	1	https://www.digikey.ca/en/products/detail/adafruit-industries-llc/2223/5629433
pH Sensor Header	Header, 4-Pin	1	https://www.digikey.ca/en/products/detail/on-shore-technology-inc/OSTVN04A150/1588864
Q1	2N3904 NPN General Purpose Amplifier	1	https://www.digikey.ca/en/products/detail/stmicroelectronics/2N3904/603420
R1(10k), R2(1k), R3(220), R4(900k), R5(4.7k), R6(10k), R7(3.9k)	Resistor	6	
Relay header	Relay Control	1	https://www.digikey.ca/en/products/detail/on-shore-technology-inc/OSTVN04A150/1588864
RPi_Vin	Header, 2-pin	1	https://www.digikey.ca/en/products/detail/on-shore-technology-inc/OSTVN02A150/1588862
Temp Sensor header	Header, 3-Pin	1	https://www.digikey.ca/en/products/detail/on-shore-technology-inc/OSTVN03A150/1588863
V1, V2	Header, 2-Pin	2	https://www.digikey.ca/en/products/detail/on-shore-technology-inc/OSTVN02A150/1588862
VR1, VR2	Voltage Regulator	2	https://www.digikey.ca/en/products/detail/texas-instruments/LM7805SX-NOPB/6110585
WL Sensor Header	Header, 3-Pin	1	https://www.digikey.ca/en/products/detail/on-shore-technology-inc/OSTVN03A150/1588863

Table 9: PCB Bill of Materials

References

[1] J. B. Jones, "Hydroponics: a practical guide for the soilless grower." Boca Raton: CRC Press, 2005.

[2] W. Lennard and S. Goddek, "Aquaponics: The Basics," Aquaponics Food Production Systems, pp. 113–143, 2019.

[3] Raspberry Pi Zero W. `https://www.raspberrypi.org/products/raspberry-pi-zero-w/?resellerType=home`

[4] "HTTP vs. MQTT: A tale of two IoT protocols," Google. [Online]. Available: https://cloud.google.com/blog/products/iot-devices/http-vs-mqtt-a-tale-of-two-iot-protocols. [Accessed: 13-Apr-2021].

[5] "Choose a Database: Cloud Firestore or Realtime Database — Firebase," Google. [Online]. Available: https://firebase.google.com/docs/database/rtdb-vs-firestore. [Accessed: 13-Apr-2021].

[6] "Get realtime updates with Cloud Firestore — Firebase," Google. [Online]. Available: https://firebase.google.com/docs/firestore/query-data/listen. [Accessed: 13-Apr-2021].

[7] David Sayson. Aquaponics Expert. Personal Communications. [Date: 25-Sep-2020]