University of Arkansas – CSCE Department

Capstone II – Final Report – Spring 2022

Aquaponics Monitoring

William Mendoza, Calder West, Hunter Yarbrough, Payton Smith

Abstract

Have you ever had a hard time maintaining the life of your plants or fish? Today, there exists no all-in-one aquaponics monitoring system that provides convenient, accurate information on the go. The purpose of our project is to find a solution to this problem by developing a mobile application that accurately detects a multitude of different levels within aquaponic systems and alerts the user when any of these measurements are outside of their desired range. This approach requires the use of an aquaponic tank in addition to the appropriate sensors: a light sensor, a hygrometer (to determine if the plants are receiving enough water), a pH sensor (to analyze water acidity), a thermometer (to read water temperature), and a water level sensor (to make sure the water levels in the tank are appropriate). The sensors are installed in the aquaponic tank based on their intended purpose, whereby a wire connects each sensor to individual ports on an Arduino UNO WiFi Rev2. This Arduino is connected to a breadboard, which powers and creates electronic circuits for the Arduino. Over WiFi, the Arduino transmits all measurements collected from the aquaponic system as data to Google Firebase, the host of our server and database. Since Google Firebase gives users the ability to store information in JSON, we chose this as our server language due to its ease of use and compatibility with the data stored in the application. After all of the data collected from the Arduino has been converted into JSON objects and stored into Google Firebase, Firebase transfers this information to the application. The application receives these attributes and displays them in a way that is easily understandable for users. This application seamlessly integrates into the busy day-to-day lives of consumers, as it has one function: to monitor the information it is collecting and notify the user when temperature, pH, water, and light levels in the aquaponic system have exceeded or fallen below their limitations. Now plant parents, new and experienced alike, can live worry-free knowing that they have an around-the-clock technology-powered system in place that will alert them when things go wrong.

1.0 Problem

The practice of cultivating plants is an environmentally friendly hobby to pursue. Not only can it be therapeutic to the hobbyist, it also encourages them to spend more time outdoors. There is even the added potential of personal or financial benefit by growing your own organic food or selling it at a farmers' market.

There is no easy all-in-one way to monitor the health of plants and fish outside of a user's living space on the go. Leaving plants and fish unmonitored can cause issues if both of them require one another to survive. Without proper monitoring equipment, the fish and plants could perish or grow at a slower rate. However, maintaining a garden, or an aquaponics system, could be very difficult for beginners. Beyond simply water and sunlight levels, more delicate plants and fish require strict levels for pH and water temperature. There is currently no simple, all-in-one system in place that can monitor these levels and notify owners when any levels that are measured are outside of their specified boundaries. Because time is such an issue people can become overwhelmed and turn away from aquaponics.

2.0 Objective

The objective of this project is to develop a monitoring system that makes maintaining an aquaponic system simpler for the average home gardener. This system should take the time-consuming work in measuring the water levels, plant moisture, pH levels, water temperature, and light levels in an aquaponic system and automate it, sending the data back to the user and notifying them if anything is abnormal in their system. The data is readily available on the user's smartphone, so they can review it whenever they want and stay on top of their aquaponics system wherever they are. As a result, aquaponics, and gardening in general, will become much more approachable to people trying to pick up the hobby for the first time and for people who simply did not have enough time to attempt it before.

3.0 Background

3.1 Key Concepts

There exists no easy, all-in-one solution for an aquaponics monitoring system that tracks factors like water levels, plant moisture, pH levels, water temperature, and light. There has been a substantial increase in the number of households that have and grow small crops/herbs. Tracking the health of plants can be cumbersome and non-transparent. Plants can take time to maintain, and they require consistent nurturing so that they can persevere and survive. Without the necessary technology to monitor the plants and fish, they could be malnourished or affected by water levels, plant moisture, pH levels, water temperature, and light. A mini aquaponic tank will be used to maintain a healthy living environment for the plants and the fish. The aquaponic ecosystem creates sleek, low-maintenance, self-watering planters for growing different types of plants. The aquaponics tank comes with a self-cleaning mechanism controlled by a pump allowing air to flow to the fish and food to float to the plants. The tank is substantially big enough to fit a small number of plants that could grow conveniently in a household.

An Arduino board acts as our monitor that maintains the health of the plant by tracking factors like water levels, plant moisture, pH levels, water temperature, and light. This is done using a variety of insertable probes. The Arduino sends information to a built HTTP server that parses the incoming requests and extracts the values. The server program is called Google Firebase, which is used as an intermediate communication medium for IoT devices, such as the Arduino. Google Firebase allows information to be stored in JSON, so JSON is the server language of choice since it is easy to control and manipulate data. This allows the application to send information back to the user so that the user has a clear understanding of the quality of their plant based on visual factors such as water levels, plant moisture, pH levels, water temperature, and light. The user is able to sign in or sign up on a mobile application that will receive all the data pulled from the Arduino software and stored in Google Firebase. This information consists of water levels, plant moisture, pH levels, water temperature, and light.

Probes are essential for monitoring a healthy environment for the fish and the plants. The most important factors include water levels, plant moisture, pH levels, water temperature, and light. The Arduino supports these necessary probes, and the probes are programmed to work correctly with the board using Arduino software. The water level probe ensures that the water level of the tank is never too low for the fish or the plants. If the water level becomes too low, an alert is sent to the user's mobile application. The plant moisture sensor ensures that the plant is getting enough water. If the plant is not getting enough water the user will not only be alerted but will be told there may be a problem with the pump. Regarding pH levels, this is one of the most important factors since certain fish will require different pH levels to survive. The pH level monitor will keep the user up to date to ensure the pH level of the water is never outside of the boundaries the user specified through the mobile application that is safe for the fish. Water temperature and light will also be necessary to sustain healthy fish and plant life, therefore these probes will be programmed so that the user can be aware if these factors are out of specified boundaries. If the light sensor does not recognize a light level within its specified boundaries, then the user can see in the mobile app that the plant needs more light.

3.2 Related Work

[1] Simple Arduino Controlled Aquaponic System is very similar to what our aquaponics system contains and is made to do, except it does not include a mobile application that sends data from the Arduino to the user. This project rather focuses on self-sustainability than user sustainability. A simple Arduino Controlled Aquaponic System takes time to reach an equilibrium. This project will also require 4 full weeks before it works, this is because the system must build up a culture of organisms to reach stability. Before this time, the system claims it can lose fish and plants. There will be required checks for the system regularly during the early stages of the system's life. Aquarium-bought chemical indicators are required to monitor the concentration of nutrients in the system. This project is similar in a lot of the ways our project works except it focuses on an ecosystem of fish and organisms. There also isn't any way to alert the user if the temperature boundary or range is out of specified requirements and does not include a mobile application that can track factors such as water levels, plant moisture, pH levels, water temperature, and light.

[2]The Smart Aquaponics with Dashboard is an aquaponics system with two tubs. The tub in the bottom would have water and fish in it. The tub at the top of the system just contains the plants in it. Plants can

either be planted into pots or planted directly into the tub at the top of this system. The excess water flows out of the tub or the pots and enters the fish tank. The water in the tank will already be oxygen-rich, so there would be no need for an external aerator for this fish tank. This system also cleans the water in the fish tank automatically while the system remains running, so the user is not required to regularly clean the fish tank. There are a lot of similarities regarding the project's hardware. We will be using similar sensors as the Smart Aquaponics with Dashboard including the plant moisture sensor, temperature sensor, and Arduino Uno. This project mainly differs in the aspect that it requires to be wired into the internet rather than being a wireless interface. The data is also sent to a server that provides a dashboard of information on a web page rather than a mobile application. There also is not any way to notify the user in case any of the Smart Aquaponics with Dashboard sensors or probes malfunction or go out of a specified boundary.

[3]The AquaSprouts Garden product is similar to what we want in terms of size but not quite as intuitive due to it being self-cleaning. We want to be able to make an app that allows the user to optimize all of the factors that go into nurturing a plant. We also want to know how our aquaponics system is doing in real-time rather than trusting a self-sustaining machine. The AquaSprouts Garden is special because it brings the revolutionary principles of aquaponic farming into a simple living space for the user. The AquaSprouts Garden circulates the water from the aquarium through the Garden's bed, the AquaSprouts Garden recycles fish waste to provide vital nutrients for growing plants, leaving the water clean and clear without the need for additional filters or frequent water changes. This is a well-designed project that is already being marketed on Amazon. In terms of the hardware and physical layout of how the plants and fish are combined into one system, our project does not drastically differ. There is no simple way of monitoring this system, this system simply monitors itself which can limit the type of plants and fish that you may want to maintain in an aquaponics system. Our system will have a pump that works the same way but will also include a list of probes so that the user can monitor all of the different factors such as water levels, plant moisture, pH levels, water temperature, and light through a mobile application.

4.0 Design

4.1 Requirements and/or Use Cases and/or Design Goals

Requirement: Provide an aquaponics monitoring system that uses an Arduino wired to a pH level sensor, water level sensor, thermometer, light sensor, plant moisture sensor that collects data every ten seconds.
Use Case: Retrieves data from sensors or probes to send off to the server.

Requirement: The Arduino must be able to connect and stay connected to Wi-Fi.
Use Case: Send collected data wirelessly to a Google Firebase database and update the user. If there is no data received from the Arduino then the server should alert the user that there may be a problem with the sensors, internet, or software.

Requirement: The application monitors the data it receives from each of the sensors to determine whether there is an issue with the aquaponic system it is monitoring.
Use Case: Allows the user to upkeep their system without having to physically check on it regularly.

Requirement: The Arduino needs to ping the server to ensure it is connected to the internet.
Use Case: The Arduino needs to stay connected to the internet so that it can send information from the Arduino software.

Design: The Arduino pulls data from the probes or sensors. This data is sent to Google Firebase, then the application pulls the data from the Google Firebase and alerts the user if any of the values are missing or out of the specified range.

Requirement: The mobile application should refresh the data periodically as well as whenever the user requests a refresh.
Use Case: Ensure the app is kept up to date so the user can be made aware if an undesired value is collected.

Requirement: The mobile application allows the user to set specified boundaries regarding pH levels, temperature, and light.
Use Case: Allows the application to determine if any of the measurements are outside of their specified boundaries.

Requirement: The aquaponics tank needs to include a pump and a tub on top of the tank.
Use Case: The pump will need to filter the water, the tank will host the water, and the tub will host the plant or plants.

Requirement: The app supports user signup using an email, password, and ID, and login with email and password.
Use Case: This tethers the user's system's info to one account, allowing the user to access their monitoring data from any supported device.

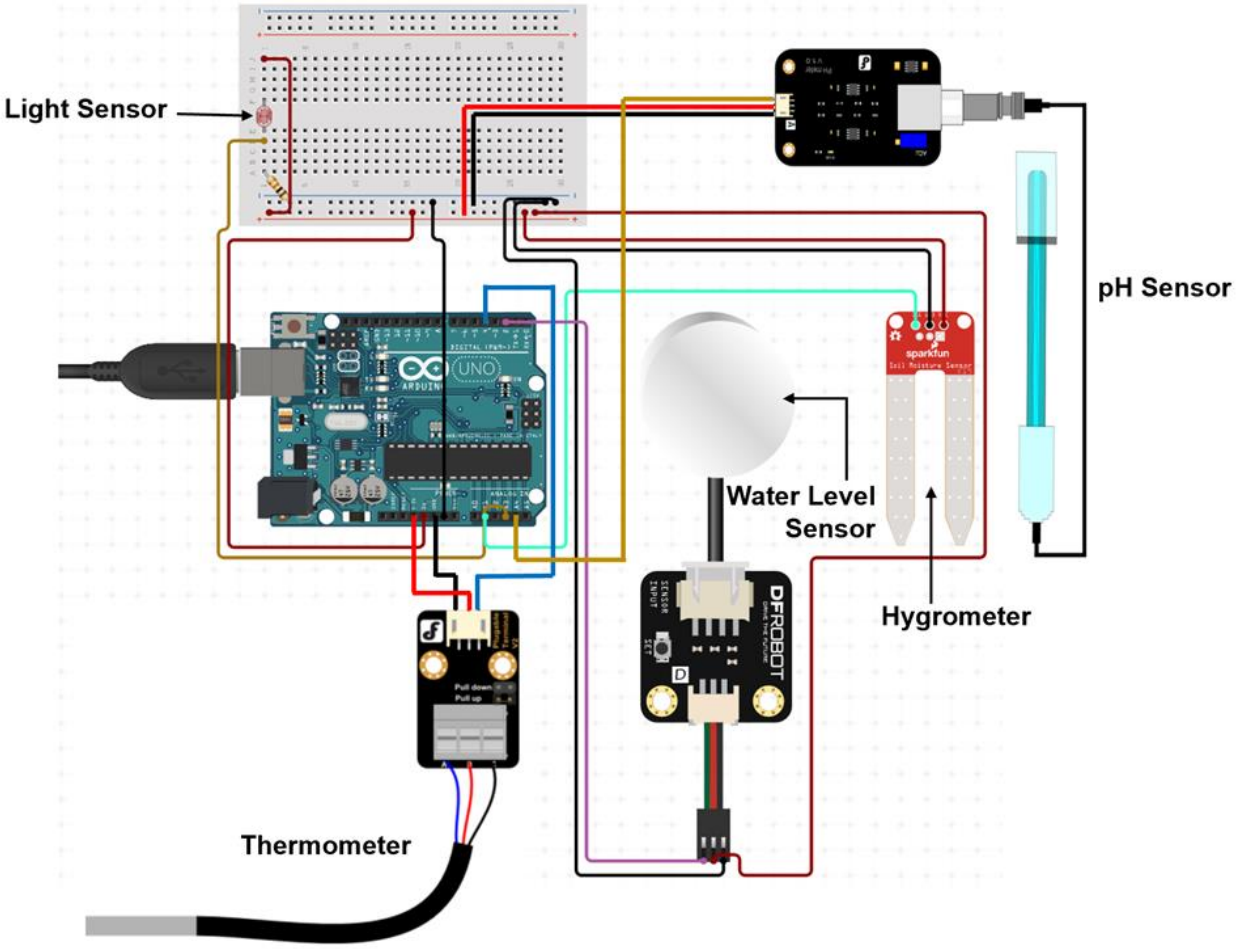
Requirement: The system should support multiple distinct clients out of one server, and it should correctly map the data from a system to its corresponding user account.
Use Case: Allows multiple people to receive/use info from our monitoring app.

4.2 Detailed Architecture

Our project is a system that monitors small-scale aquaponics systems. This system is comprised of two components: a hardware component that consists of an Arduino and its connected sensors, and a software component, which is an iOS app that receives data from the Arduino through a real-time database hosted by Google Firebase.

The aquaponics system itself is monitored on-site by several probes connected to an Arduino board. Our system consists of five sensors to measure data from the aquaponics system. Three of these sensors are placed in the lower tank with the fish and monitor the conditions of the water to ensure the water is safe for the fish and plants. We have a pH sensor to measure the acidity of the water, a thermometer submerged in the water to measure temperature, and a water

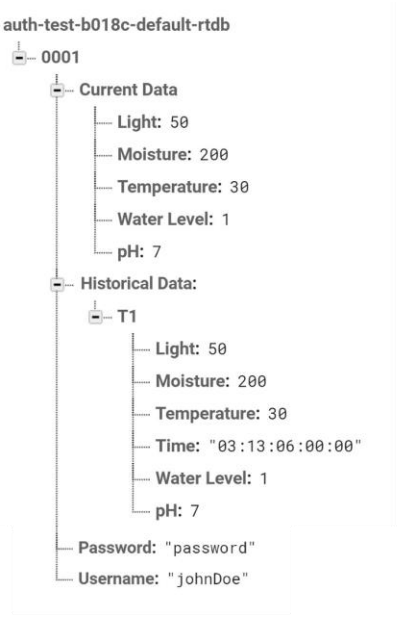
level sensor to ensure water does not overflow out of the tank. The last two sensors are placed in the upper portion of the aquaponics system to collect data on the plants: a light level sensor that determines if the plants are getting the correct amount of light and a hygrometer to measure if the plants are getting any water, which ensures that the water pump hasn't malfunctioned. Due to the number of sensors and wires in our system, we are also using a breadboard to better organize each individual sensor's wiring and connect the Arduino's 3.3V and ground pins to multiple different sensors. However, one sensor in particular, our water thermometer, kept malfunctioning when connected to our breadboard. After testing multiple different temperature sensors to no avail, we purchased one that came with its own separate board and did not need to be connected to the breadboard, which now works perfectly. A schematic of the Arduino and sensor system can be seen below.



For the Arduino board itself, we are using the Arduino UNO Wi-Fi Rev2 for its on-board Wi-Fi connectivity, which allows the measured data to be sent to the database and, eventually, the user's app. The board takes the data from each of the sensors and sends it to the app so that it can be evaluated to ensure each of the values is within their desired ranges. The board sends the

data frequently, around once every 10 seconds, to ensure that the aquaponics system is not running incorrectly without the user's knowledge.

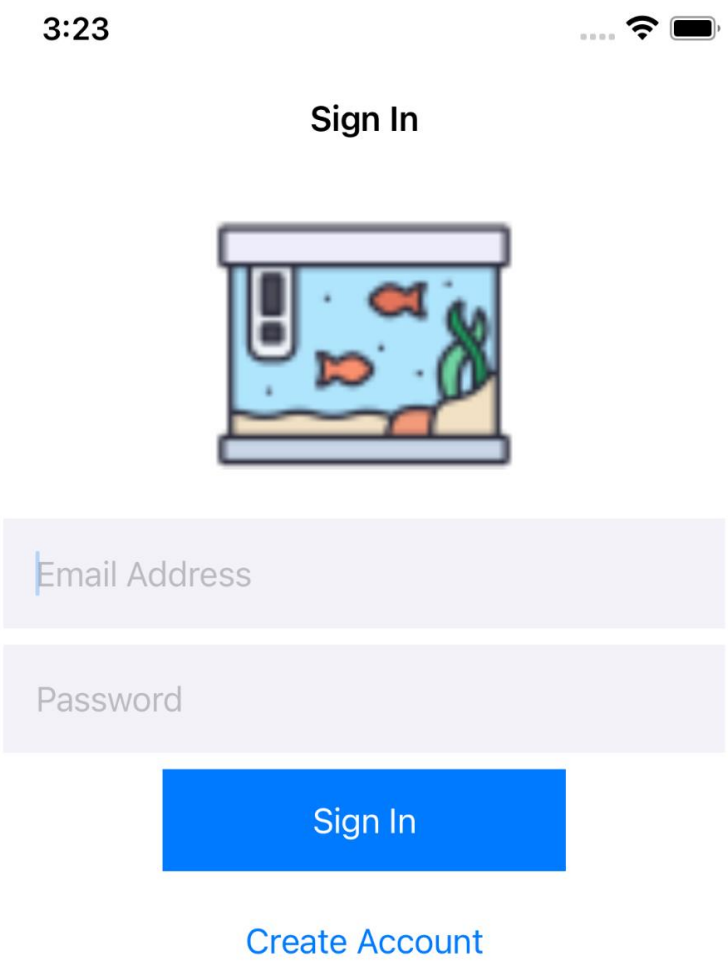
The server and app comprise the second component of this project. The server is hosted using Google Firebase and receives data from the Arduino, stores the data as JSON objects, and sends the data to the app. The database is formatted as such. When a user signs up for the first time with an email, password, and Arduino ID number, a new path is created in the database with the Arduino ID number as the root. The email, password, path to current sensor values, and a path to past sensor values are stored as children of the Arduino ID number. The path that stores current values only stores the five sensor values, which are updated every ten seconds. The section that stores the past recorded data stores all five sensor values and a timestamp. Each time the Arduino records data from the sensors, it updates the values under the path to the current values and pushes a new JSON object to be a child of the path that stores past values. The image below is what the database structure looks like after a user has signed up and the first set of sensor readings have been pushed to it.

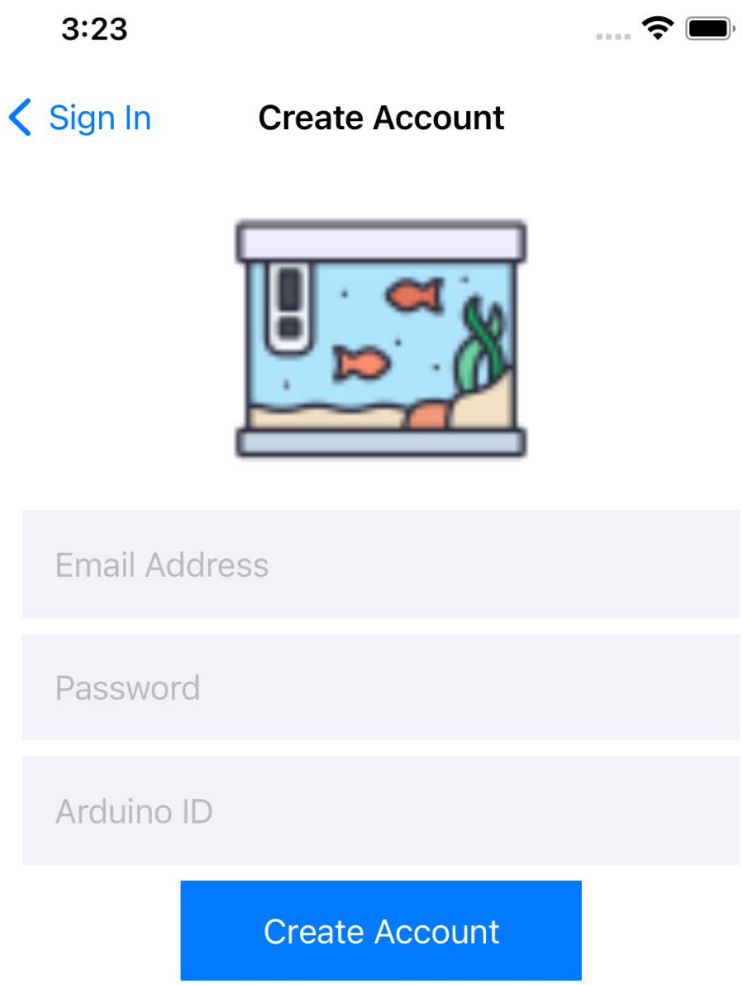


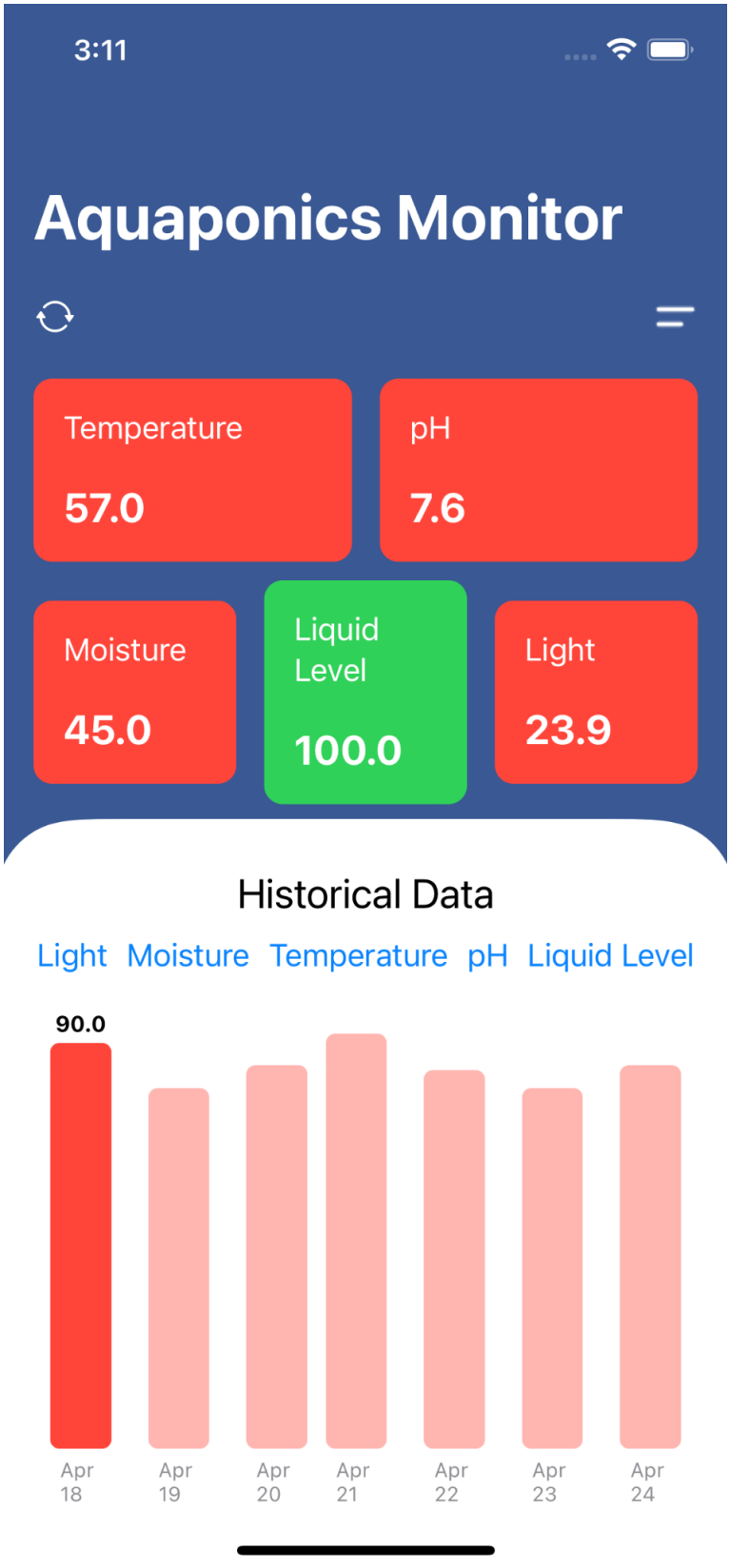
The application is hosted on an iOS device and created as a mobile application. The initial development of the mobile application required research and extensive knowledge of XCode, Swift, and SwiftUI. The frontend of the application is written in SwiftUI, and the backend is written in Swift. Branches were created through GitHub so multiple users were able to modify, change, or create new features for the application. The mobile application is programmed to receive data from Google Firebase. This data is then manipulated so the user is presented with an organized and simple layout of factors such as water levels, plant moisture, pH levels, water temperature, and light. A JSON file is used to store the project number, firebase URL, project id, and storage bucket, which allows the mobile application to seamlessly pull

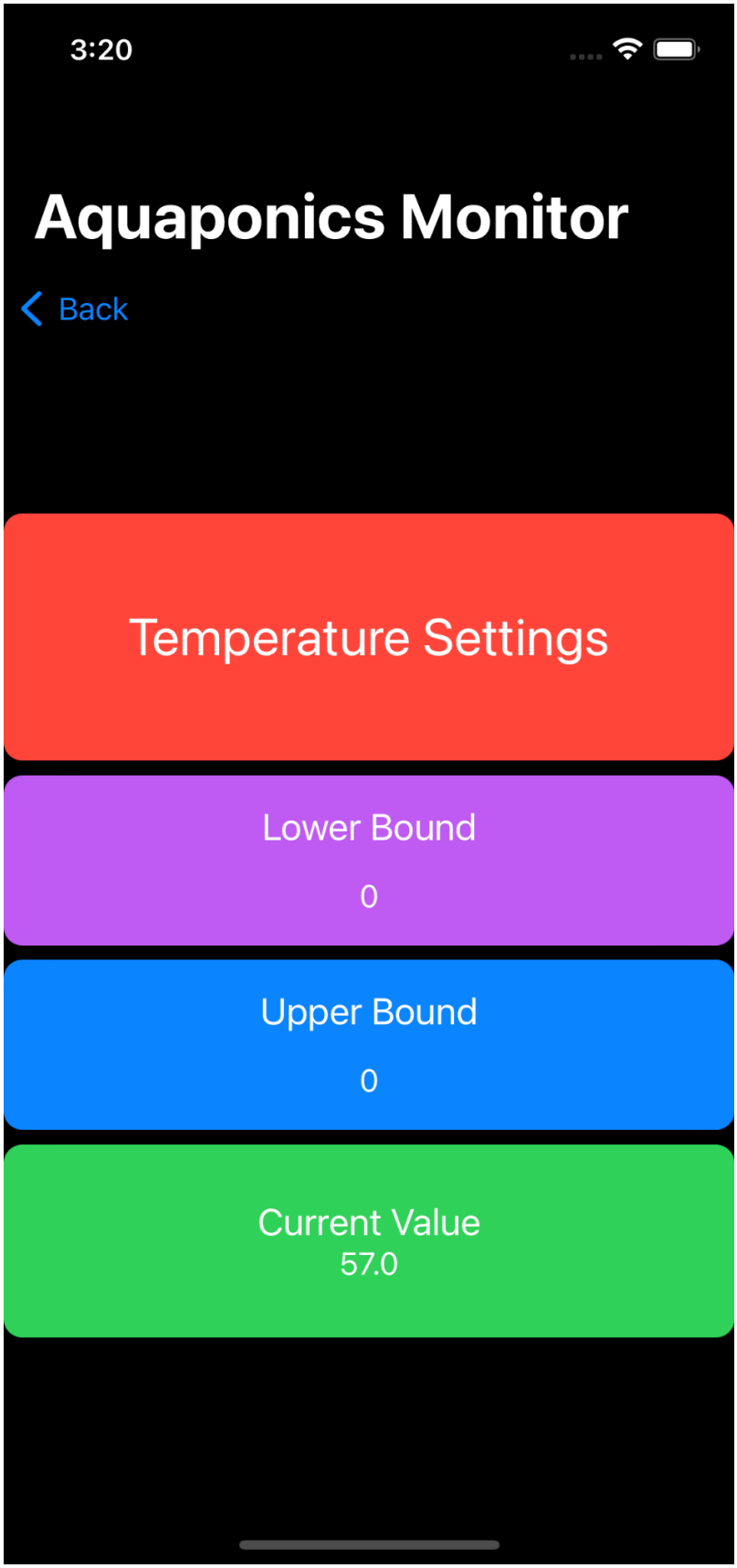
information from Google Firebase and manipulate any incoming data. The data received from Google Firebase updates every ten seconds so the user is shown retrieved data in a timely manner. Default ranges for the values returned from the sensors are implemented into the mobile application in case the temperature is too cold or hot, the pH is too low or high, the light has not been detected during an extensive time period, water levels in the tank are low, and if the plants are receiving water or not.

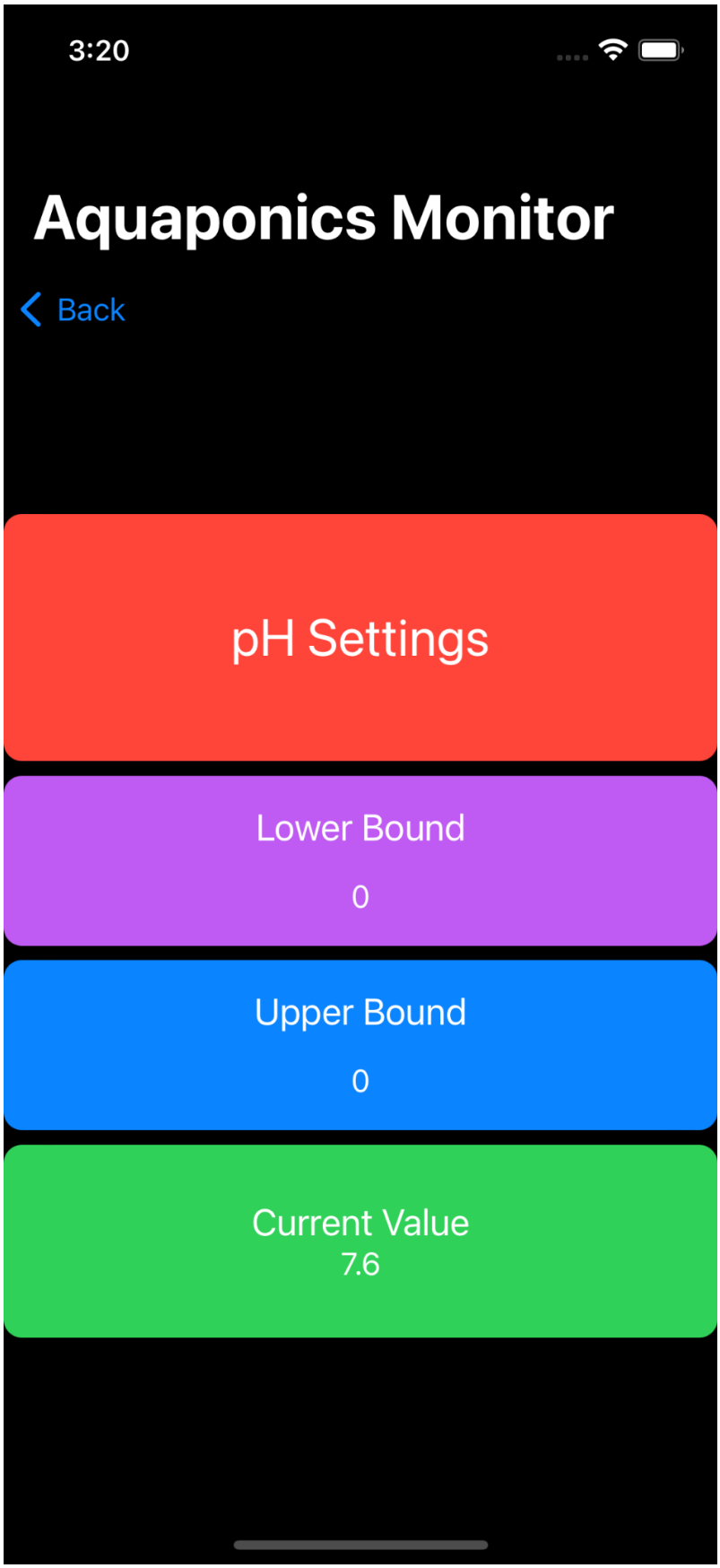
The User Interface of the application is built using SwiftUI, and is very simplistic and offers a clean, organized, and data driven view for the user. When the application is opened the user is offered with a login page, the login page includes an email and password section so that the application promotes a secure experience for their personal aquaponics system. The signup page has fields for an email, password, and Arduino ID. This connects the user and password to the specific Arduino system that the user wants to use. Once the user is connected to an Arduino system through logging in or signing up, the user is brought to a home page. The home page displays the data received from the Arduino system and allows the user to modify the range boundaries for pH level and water temperature. The user can also see historical data of important factors such as temperature and pH. A JSON file hosts all of the historical data in Google Firebase and passes along this data to the mobile application where the user can view the data over the last week. The app has a listener that pulls data from the database every time the database updates its stored values. The app uses the ranges to compare against the received sensor data to determine whether the received data is above or below the range. Since the user can configure the ranges for each of the sensors, the system can be used for a variety of plants and fish. The app also displays each of the five data points recorded from the last week. A notification button is also illustrated near the top right corner so users can view past or present alerts sent from the server. Users can also access or edit their profile information and choose to sign out. System settings and measurement boundaries are also included in case the user needs to specify certain ranges for their system. The data that is displayed on the app is updated every ten seconds. There is a refresh button on the home page that updates the current data by sending a message to the Arduino telling it to take a reading and upload the data to the server. A diagram of how each part is connected is included below along with a screenshot of the layout of the app and how the user interface is organized.

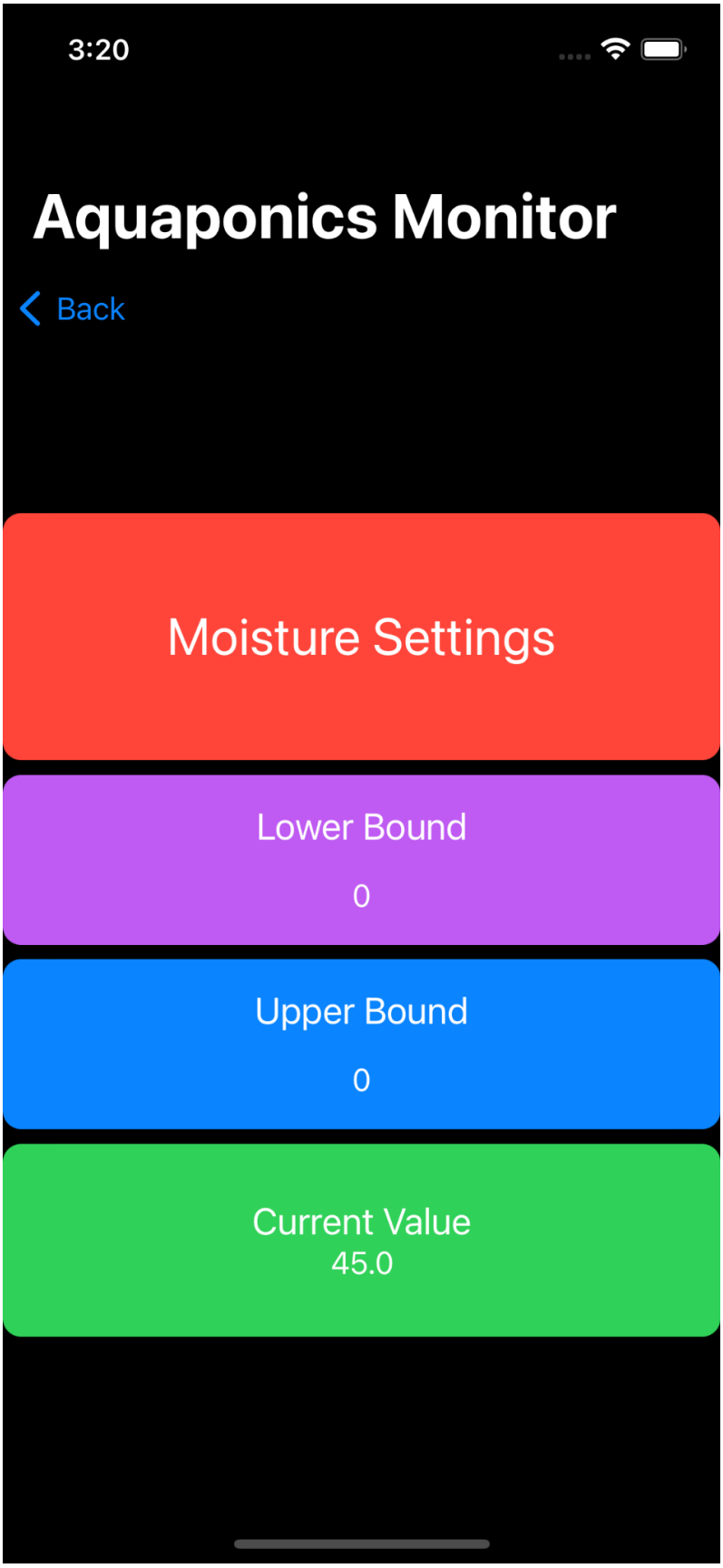

3:23
Sign In
Email Address
Password
Sign In
Create Account

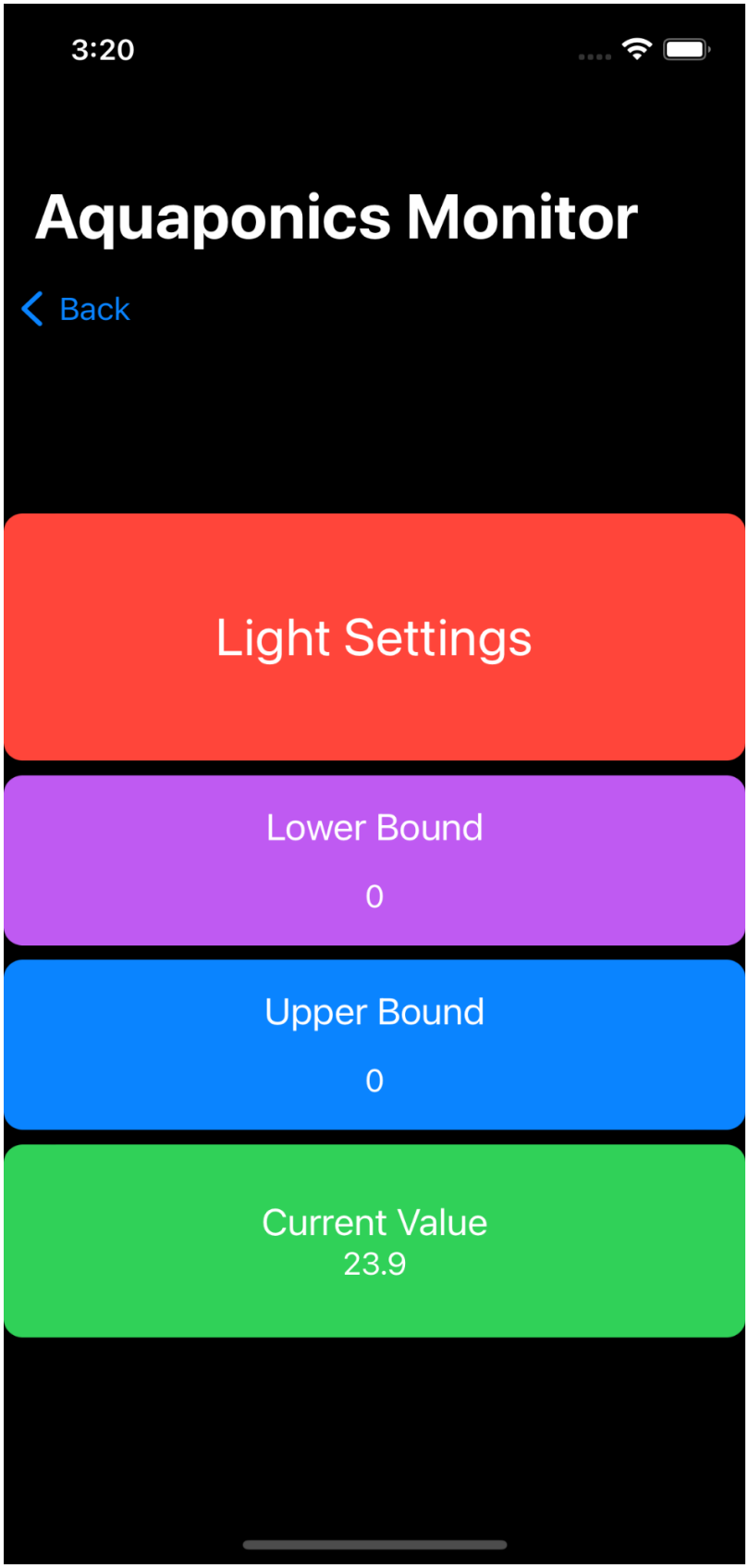

3:23
Sign In
Create Account
Email Address
Password
Arduino ID
Create Account

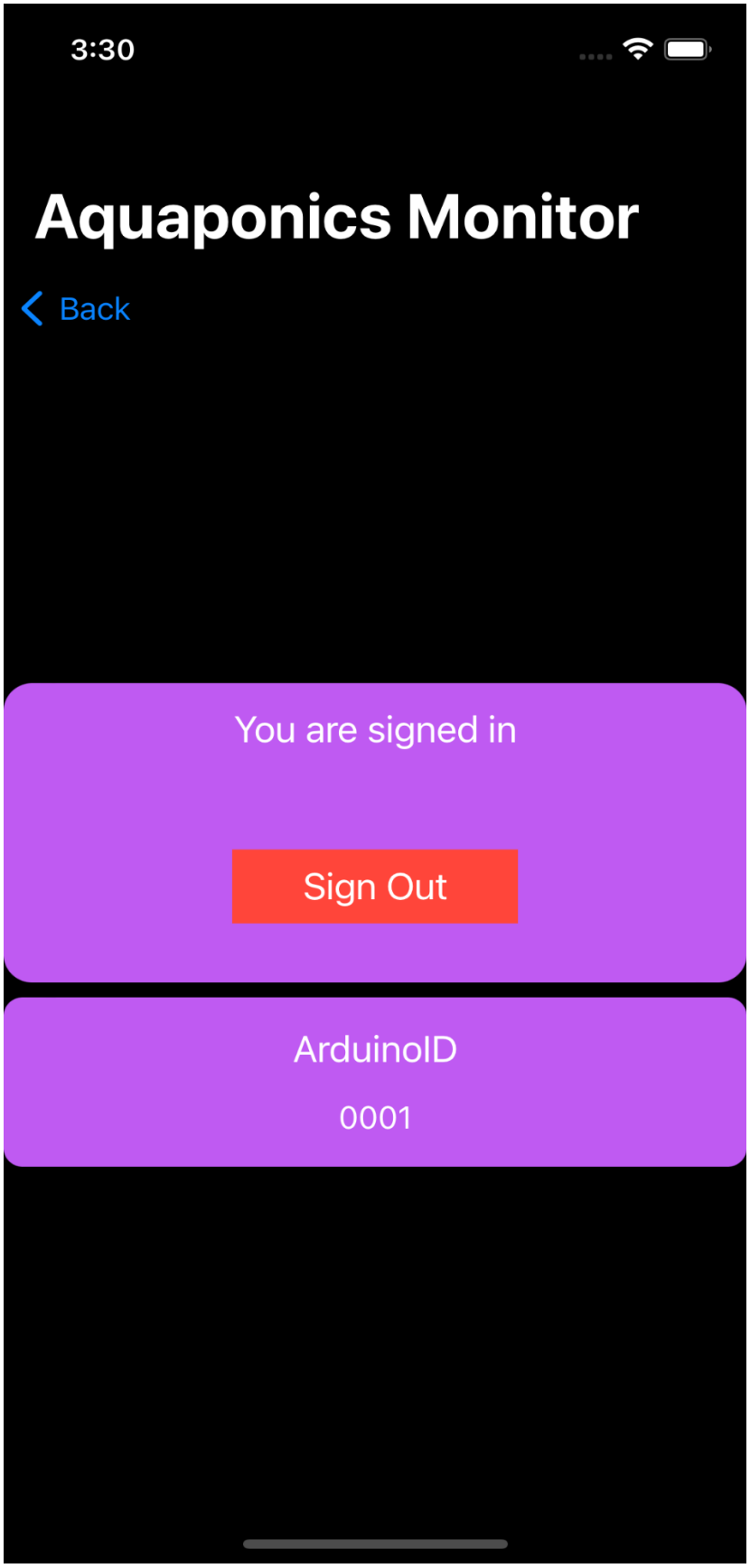

3:11
Aquaponics Monitor
Temperature
57.0
pH
7.6
Moisture
45.0
Liquid Level
100.0
Light
23.9
Historical Data
Light
Moisture
Temperature
pH
Liquid Level
90.0
Apr 18
Apr 19
Apr 20
Apr 21
Apr 22
Apr 23
Apr 24

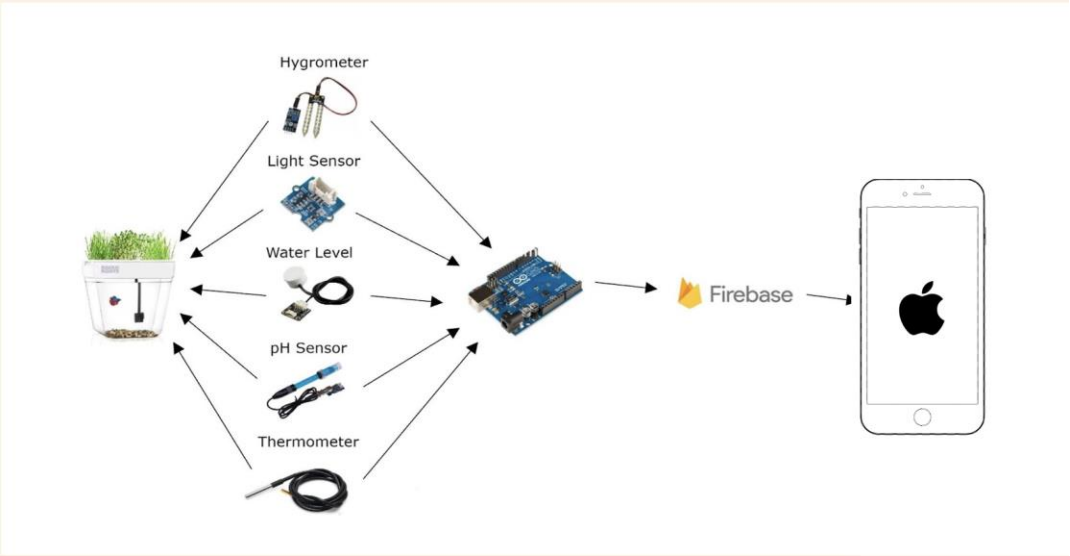

3:20
Aquaponics Monitor
Back
Temperature Settings
Lower Bound
0
Upper Bound
0
Current Value
57.0


3:20
Aquaponics Monitor
Back
pH Settings
Lower Bound
0
Upper Bound
0
Current Value
7.6


3:20
Aquaponics Monitor
Back
Moisture Settings
Lower Bound
0
Upper Bound
0
Current Value
45.0


3:20
Aquaponics Monitor
Back
Light Settings
Lower Bound
0
Upper Bound
0
Current Value
23.9


3:30
Aquaponics Monitor
Back
You are signed in
Sign Out
ArduinoID
0001



While the monitoring system and application achieve the primary goals of our project, there is still room for future work. For instance, in the mobile application, we could add the functionality for a user to register multiple distinct aquaponics systems to one account, which would save the user the hassle of creating multiple different accounts just to monitor their own systems. We could also allow the user to add a description of their aquaponic system in the app so the user can easily determine which system they are currently monitoring. Implementing notifications for our application required a subscription to Apple's Developer Program, so we also opted to not include notification support for now, but notifications could be implemented in the future. In regard to the Arduino portion of the project, changes could be made to the hardware setup to make it easier for a consumer to set up the monitoring system themselves. Rather than using a breadboard, the monitoring system could consist of wires soldered to boards that are contained within some sort of shell, which would protect the electronics of the system and make it more convenient to set up.

4.3 Risks

Risk	Risk Reduction
Wi-Fi connection to the Arduino is disrupted	Implement code within the Arduino WiFi Rev2 to restart the WiFi connection after it is lost. If it is still lost after the restart, send the user a

	notification that the Arduino has lost connection to the internet.
Electricity and water interacting (fire hazard)	Create drip loops for any wire coming out of the water. In this case, that would be the hygrometer, pH sensor, and water level sensor. This will prevent water from reaching any electricity connectivity points.
Plant pathogens/disease	Regularly check, clean, and sanitize the aquaponic system. Any water contaminants will directly affect fish and all of the plants.

4.4 Tasks

1. Create individual sites
2. Initially create team site
3. Build the rest of the team site so it meets all requirements listed in the site expectations
4. Begin implementing the Arduino-breadboard system and conducting initial research on aquaponics and Arduinos
    a) Download Arduino software to be used with the system and ensure it works as expected with Windows
    b) Perform additional research about how to properly connect the Arduino and the breadboard so they function as one system with the Arduino software (videos, diagrams)
    c) Conduct research on linking wires (that connect to each sensor) between the Arduino and breadboard. This includes in depth research on ports, input functions, and output functions for sensors which measure environmental factors.
5. Test liquid level sensor and hygrometer with simple test cases (before the sensors have been submerged in water or placed in the aquaponic system)
    a) If the sensors do not return expected values, check the connections for both inputs and outputs to ensure that nothing in the breadboard is short circuiting
    b) If the breadboard is *not* short circuiting, check the code in the software to make sure there are no issues that may cause outstanding errors in regards to the functionality of the system
    c) Ensure before we continue that we have all necessary components to make the aquaponics system function as expected - additional materials required: photoresistors, temperature sensors, breadboard, USB to USB-B wire for connections

6. Connect Arduino to Google Firebase and get initial code running to make sure Firebase is retrieving values from the sensors
    a) Conduct research on Python libraries that work with the Arduino board, software and sensors
    b) Implement simple Python concepts that return values based on data collected
7. Connect light sensor to Arduino and test it with the software to make sure the light sensor is returning the correct values
8. Create accessible spreadsheet linked to team task list to make team site easier to navigate
9. Begin initial development of the Android application
10. Update future task list to be more in line with our current project trajectory
11. Work on connecting Google Firebase to the Android application
12. Research how to send data collected by sensors to Firebase
13. Obtain consistent values from pH sensor
14. Link Google Firebase to Android Studio
15. Create GitHub repository; Create branches for Android application
16. Begin researching Firebase code to be used in the application
17. Connect Google Firebase and Android studio
18. Link Android studio with GitHub
19. Code application so data is being received from the Arduino; make sure the data is being refreshed in a timely manner (every few seconds or so)
20. Map out goals for application UI and begin creating low fidelity mockups so there will be a framework in place when the interface is being developed
21. Make a file to store users' sign-in information and collected data
22. Create high-fidelity mockups with specified fields (input fields, tables, etc.) and ensure that the user experience is efficient and easy to understand
23. Obtain consistent values from temperature sensor
24. Start writing preliminary project report/slides
25. Begin researching how to develop a user interface in XML
26. Finish writing Preliminary Proposal report and slides; rehearse for presentation

*** At this point in the project, we decided to revise our plan for frontend (User Interface) code. Rather than developing in XML for Android, we decided to write the user interface of our application in Swift on XCode for iOS. We thought this would be more realistic for practical use given that a majority of people own iOS devices. Additionally, Swift is quicker and simpler to execute than XML.

27. Compile documentation with reference links to code for different page components contained within the UI

28. Develop a clear understanding of Swift/SwiftUI and its basic principles before beginning development in XCode
29. Develop a simple sign-in/create account page and authenticate user sign-in through Google Firebase
    (a) The create account page should allow for a user to create their own personal username and password to be stored in Google Firebase and used to sign in later.
    (b) If the username the user is attempting to create has already been stored in the database, then the user will not be able to utilize this username.
30. Implement a sign-out button that allows the user to sign out of their account; once signed out, user should be led back to the sign-in page.
    (a) Upon launching the app, if user is signed out or has not created an account, they will be prompted to sign-in or create an account on the sign-in page
31. Write code which will allow a user to set minimum and maximum values in the application for their accepted range regarding data measurements within the system
32. Create poster for poster session and ensure it includes all necessary details about changes made to the application (Poster Session: April 28, Poster Due: April 19)
33. Make necessary changes to the final proposal report in regards to the change of language and the different features being implemented
34. Store historical data for up to a week from the time it was collected
35. If a sensor's measurement is out of its user-specified range, indicate this to the user by changing the color of the bar associated with that sensor in the data table within the application
36. Test the consistency of the aquaponics system and application in unison with all features we have implemented so far
37. Make final changes to final project report and slides

4.5 Schedule

Tasks	Schedule
Tasks 1-2	1/20-1/23
Tasks 3-4	1/23-1/25
Task 5	1/25-2/01
Tasks 6-8	2/01-2/06
Tasks 9-11	2/06-2/09

Tasks 12-20	2/10-2/24
Tasks 21-25	2/25-3/10
Task 26	3/12-3/17
Tasks 27-29	4/1-4/10
Tasks 30-33	4/10-4/20
Tasks 34-37	4/21-4/24

4.6 Deliverables

- Design Document: The hardware consists of an Arduino Uno Wi-Fi Rev2 that works on the Arduino software, probes or sensors that measure water levels, plant moisture, pH levels, water temperature, and light that work with the Arduino Uno Wi-Fi Rev2, an Aquaponics tank that includes a pump, a server hosted on Google Firebase, and a mobile application developed for iOS using Swift.
- Arduino and Application Code: The Arduino software was developed in Arduino's IDE. The code sends the information received by the Arduino board to Google Firebase. Code for the iOS application is written in Swift and made to retrieve JSON data from Google Firebase.
- Aquaponics Tank: Plants are placed on top of the tank, and a fish is in the reservoir below the plants. The aquaponics tank includes a simple pump that pulls water from the tank upward to the plants and pushes oxygen created by the plants down to the fish. Sensors placed in (pH sensor, water level sensor, thermometer) and on (light sensor, hygrometer) the tank are connected to an Arduino WiFi Rev2.
- Google Firebase(server/database): Google Firebase is the server that receives and stores data as JSON objects sent from the Arduino. The server sends information from the Arduino to the mobile application. Google Firebase also hosts user information and allows for multiple users to sign up

5.0 Key Personnel

William Mendoza – Mendoza is a senior Computer Science major in the Computer Science and Computer Engineering Department at the University of Arkansas. He has completed Software Engineering, Programming Paradigms, and is currently enrolled in Embedded Systems. He has had some experience in the past working with an Arduino board. Mendoza will ensure that the physical aquaponics/plant system interfaces with the Arduino correctly in order to connect it to the application.

Robert Yarbrough – Yarbrough is a Senior Computer Science major in the Computer Science Department at the University of Arkansas. He has completed Programming Paradigms, Software Engineering, Database Management, Computer Organization, and Programming Foundations I and II. Yarbrough has worked on personal Python projects that deal with Machine Learning and Database

Management. Yarbrough will ensure that the application will work properly with the Arduino device and the User Interface is implemented correctly so that the hardware and software work in unison.

Payton Smith – Smith is a senior Computer Science major in the Computer Science and Computer Engineering Department at the University of Arkansas. She has completed Programming Foundations I and II, Software Engineering, Programming Paradigms, and Database Management. Smith has experience interning as a User Experience Designer, focusing specifically on the creation of app wireframes and ensuring that workflows are efficient from beginning to end. Smith will make sure that the user experience in the app is seamless and meets the goals set at the beginning of the project.

Calder West - West is a senior Computer Science major at the University of Arkansas. He has completed Software Engineering, Database Management, Computer Organization, Programming in C, Programming Paradigms. He has experience working on websites and mobile applications. West will ensure that the app and website are implemented correctly and can connect to the aquaponics system.

Dr. Matthew Patitz - Dr. Matthew Patitz received his B.S. and M.S. in Computer Science from Iowa State University, then worked at a start-up software company, SupportSoft, in Redwood City, CA as a software engineer and team lead from 2000-2005. In 2005 he returned to Iowa State University to pursue his PhD in Computer Science, which he completed in 2010. From 2010 to 2012 he was an assistant professor of Computer Science at the University of Texas-Pan American. Since 2012, Matt has been in the Department of Computer Science and Computer Engineering at the University of Arkansas. He is currently an associate professor. His research interests are DNA computing, algorithmic self-assembly, and theoretical computer science in general. He has received research grants from the National Science Foundation, including a CAREER award in 2016, and published over 60 peer-reviewed conference and journal papers.

6.0 Facilities and Equipment

Description of all facilities and/or equipment required and/or utilized for the complete project.

-An Arduino UNO WiFi Rev 2 is required so that data is collected from each of our probes, and with the built-in WiFi capability, it can send data to our server.

-A prebuilt hydroponic system helps us ensure our sensors work correctly and that the data makes it from our monitoring system to the app.

-A power supply that powers the Arduino UNO WiFi Rev 2 and includes a long enough cable to reach the tank.

- A breadboard that connects to the Arduino UNO WiFi Rev2 and creates electronic circuits for our sensors.

-A plant light that provides light so that the plant can grow properly under the required conditions.

-A fish helps feed the plant's nutrients, which helps with testing the sensors in our aquaponics system.

-Plants are required for the aquaponics system and will act as a test case for the plant moisture sensor.

Sensors(compatible with Arduino)

-The temperature sensor is required to make sure that the water in the tank is warm or cold enough for the fish.

-The pH sensor is used to ensure that the water's pH levels are safe enough for the fish.

-The water level sensor ensures that the tank has enough water and there aren't any issues regarding leaks.

-The light sensor recognizes if there is enough light for the plant.

-The plant moisture sensor recognizes if the plant is getting enough water or not.

7.0 References

[1]https://www.instructables.com/Simple-Arduino-Controlled-Aquaponic-System/

[2] https://create.Arduino.cc/projecthub/neuron/smart-aquaponics-with-dashboard-4e8b23

[3]https://www.amazon.com/AquaSprouts-Garden/dp/B01B4ZRVR4/ref=pd_lpo_2?pd_rd_i=B01B4ZRVR4&psc=1

[4]Leithauser, David C. *How to Cheaply Monitor and Automate Your Aquaponic-Hydroponic Garden with Arduino/Genuino*. 2017.