OpenAquaponics Proposal

Bau Sauvage, Luna Richards, Elias Obrist

Last Revised: 29 April 2025

Abstract

Aquaponics is a food production system that combines raising fish with cultivating plants, allowing them both to mutually benefit. By sharing water, the plants can filter the fish's waste and use it as fertilizer to help them grow. While this is much more efficient and less wasteful than traditional aquaculture or hydroponics alone, the system can be difficult and time-intensive to build and manage. To reduce those barriers, we developed a prototype automated care system - equipped with an automated feeder, an automated lighting system, a camera, and a variety of sensors - that allows the user to monitor and control the condition of the aquaponic environment remotely. This prototype integrates easily accessible commodity hardware to provide automatic and adjustable control over the environment within the aquaponic system. Building a simple and modular aquaponics platform is a useful endeavor for several reasons. It can lower the barrier to entry for industries like indoor farming and potentially allow for smaller and/or scalable indoor farms to exist. Alternatively, it could be a useful project for enthusiasts who want to experiment with growing their own produce and herbs, or for children, elderly people, or anyone who cannot manage such a system on their own. Designing and building such a system with easily acquired off-the-shelf components only increases accessibility and aims to drive down costs for the end user.

Introduction

Food production through agriculture is a vital pursuit in the continuance of the human race; food is one of the most basic human necessities. There are several major agricultural challenges faced across the globe, threatening the major systems of food production, and, with them, life as we know it. The hard hitters include climate change, water shortages, labor shortages, soil erosion and degradation, the urbanization of agricultural land, and the high and continually rising cost of land. An automated aquaponics system is proposed to uniquely address these challenges by offering a contained fish and plant growth system that can be used indoors and requires no soil, less water, little space, and very little human management. While this system can be created and used on a small scale for decentralized personal food production, it can also be scaled horizontally and vertically for commercial production. There are three main elements of the automated aquaponic system detailed in this proposal:

1) A monitor system to track water level, pH, temperature, and total dissolved solids (TDS) using a variety of sensors connected to a Raspberry Pi Model 3B+ single-board computer, henceforth referred to as the Pi.

2) An automated care system running within the Pi's native Python programming environment to feed fish the fish, pump the water between beds, and provide lighting to plants.

3) An HTML5 web interface, hosted on the Pi, that will allow the user to adjust settings and access the sensor data and camera view to monitor the condition of the aquaponic environment remotely.

This proposal includes a background in aquaponic growth systems, the qualifications of the design team, a work schedule, and a proposal statement that details the functional and non-functional requirements, performance requirements, interface requirements, architectural design documents, and standards and tools used in development. It ends with a description of the expected results of this project.

Background

(Sources cited on final page)

Aquaponics is a combination of aquaculture and hydroponics. Aquaculture refers to the cultivation of aquatic life, and hydroponics refers to the soilless cultivation of plants. The principle of their combination is rather simple: a substantial portion of the nutrients that plants need can be made from fish waste, and the oxygen that fish need can be produced by plants (7). Most modern aquaponics systems operate by having a fish tank where the water gets pumped into a medium full of bacteria that can break down the toxic fish waste, such as ammonia, into nitrates (7). The water is then moved into a medium that the plants are growing in and subsequently sent back into the fish tank. The nitrates from the fish waste act as extremely effective fertilizers for the plants, and the process also cleans the water for the fish and oxygenates it. This is an artificial ecosystem with a miniature version of the nitrogen cycle. Several common designs implement this basic idea in the DIY space (1, 2, 3, 4, 5, 6), and there exist different industrial implementations of this concept.

This project will be an open-source monitoring and automation system for DIY aquaponics systems. It will monitor water quality and temperature, and will also include a camera for monitoring the fish and plants, and a web interface that allows the system to be controlled from a local network. There are existing monitoring systems for aquaponics that exist (1, 2, 3, 4, 5, 6); there are even open-source ones (3, 4, 5). The problem is that most of them seem to be geared towards industrial aquaponics systems, and a staggering amount of the monitoring for DIY and beginner systems is fairly manual and fairly technical. The goal of this project is to bridge the gap and provide a cheap monitoring and automation system for scalable aquaponics setups for beginners.

Qualifications:
BAU SAUVAGE

Bauwow5@gmail.com | (970) 644-1864 | Bozeman, MT

Skills Summary

Successful student through high school with a 4.24 cumulative GPA. Current student studying Computer Science at Montana State University maintaining a 3.9 GPA. Natural problem solver and effective communicator with strengths in customer relations, self-management, and attention to detail. Fast learner and highly adaptable. Strongly interested in mathematical theory, UI Design, and cyber security.

Education

Palisade High School, Palisade CO / August 2017-May 2021
4.24 GPA and extensive Honors, AP, and IB classes

Montana State University, Bozeman MT / August 2021-Present
Computer Science Bachelor's Degree

Psychology Minor

WUE Scholarship

Experience

Hoplite Industries, Bozeman, MT

Security Operations Center Analyst / October 2024 – Present
Honed skills in cyber security threat detection, network and appliance monitoring, graph creation and analysis, customer service, ticket system management, and handling of Controlled Unclassified Information while working with Hoplite's unique patented technologies.

Black Box Design, Bozeman, MT

AV Technician at the City of Bozeman / June 2023 – Present
Developed skills in presentation graphics (Pro Presenter 7), self-management, and professionalism while facilitating remote engagement at City of Bozeman public meetings.

Awards and Acknowledgements

MSU oSTEM Leadership/ elected position in a student club, Spring 2022 - Present

MSU Honor Roll / Fall 2021, Spring 2022, Fall 2022, Spring 2023, Fall 2023

Luna Richards

Cell: (847) 977-5933 | Email: dylanjr821@gmail.com | LinkedIn: https://linkedin.com/in/djr821

Overview: Recent computer science graduate, seeking full-time work in System Administration, Computer Science, or IT services with exceptional service and dedication.

B.S. Computer Science, Montana State University, Bozeman, MT, 2025

- Minor: Computer Engineering
- Experiences: Java, Python, C/C++, Linux system administration, cybersecurity, software engineering, computer architectures, UI design, HTML5/CSS web design, circuits, digital logic, FPGAs and VHDL, microcontrollers, robot vision, human-computer interaction, technical writing.
- Activities: Student Government, Senator (2023-25). Elected by fellow students to serve two terms. Coordinated student safety programs with administrators.

Certification: Microsoft Technology Associate, Windows OS Fundamentals, March 2020.

American Computer & Robotics Museum, Visitor Services Lead/IT Support Specialist, Bozeman, MT, September 2020 – Present

- Solutions-oriented management of IT infrastructure, social media, and staff scheduling.
- Increased responsibilities progressively through tenure. Training new hires and scheduling.
- Rebuilt and streamlined the museum's internal network infrastructure.
- Assisted with curation of exhibits and interactive experiences.
- Providing exceptional customer service in the museum and at special events.

College of Lake County, Computer Lab Assistant, Grayslake, IL, Summer 2021

- Provided level 1 desktop support for 100+ station computer lab and Pharos print server.

Freelance Computer Repair Specialist, October 2016 – Present

- Specialized in PC repair & troubleshooting. Sale of refurbished computers.
- Notable clients include Youth Conservation Corps, NFP and Grayslake Youth Center.

Gamers HQ, Gaming Experience Guide, Third Lake, IL, March 2019 – December 2019

- Provided level 1 desktop support for 30 computer stations. Set up large scale gaming events.

Eagle Scout, COOL Ministries, Waukegan, IL, November 2018 – October 2019

- Led a team to refurbish twelve laptop computers for a homeless transition program.

Volunteer, FreeGeek Chicago, Chicago, IL, March 2017 – March 2020

- Refurbished computers for sale to low-income households.

Elias Obrist

53440 Barber Trail

eliasobrist11@gmail.com

(323) 552 6621

Education

Montana State University, math and computer science double major 4th year, currently enrolled- cumulative GPA: 3.03

Santa Monica College, concurrent high school enrollment, math classes included Algebra 2, Precalc, Calc1, Calc2.

Work Experience and Projects

Large Language Models January 2025 - (present)

Running large language models locally on bare metal, and doing experiments with integrating them in python programs.

Bicycle Rental Summers of 2023 and 2024

Bike repair, helping customers, painting, and general repairs of property at Spoke House Bike Rental in Morain State Park PA.

Mining Cryptocurrency January 2021 – Dec 2021

Building a computer to mine Etherium, monitoring the hash rate, over clocking and undervolting GPUs, and managing a linux os.

Using Linux August 2020 – (present)

Using linux for day to day computer needs, recovering data from old computers, installing and customizing linux desktops.

Robotics Club, First Robotics Competition. August 2017 – May 2019

Westchester High School Robotics Club, Team leader, and Mechanic.

Skills

- java
- python
- C
- OpenScad
- Linux
- Bicycle Repair
- Linear Algebra
- Machine Learning

Work Schedule

Week of:	General	Luna	Bau	Elias
1/19/2025	Begin project work	Start ordering parts	Outline of web interface	Outline of automated care system
1/26/2025				
2/2/2025			rough draft of web interface	rough draft of automated care system
2/9/2025		have all sensor equipment: Raspberry Pi, sensor array, smart plugs, camera		
2/14/2025	have all prototype equipment: tank, feeder, lights, pump, grow bed, bell siphon, et. al.			
2/23/2025		all sensor readings coming into program successfully		
3/2/2025		Modify grow lamp		
3/9/2025		Construct grow bed	completed web interface	completed automated care system
3/16/2025	Spring break	Spring break	Spring Break	Spring Break
3/23/2025	Final assembly	Assembly	Integrate Web Interface	Integrate Automated Care System
3/30/2025	Debugging			
4/6/2025	Debugging			
4/13/2025	Completed Prototype	Finish Assembly	Complete Integration	Complete Integration
4/20/2025	Present at research celebration	Fine-tuning	fine-tuning	fine-tuning

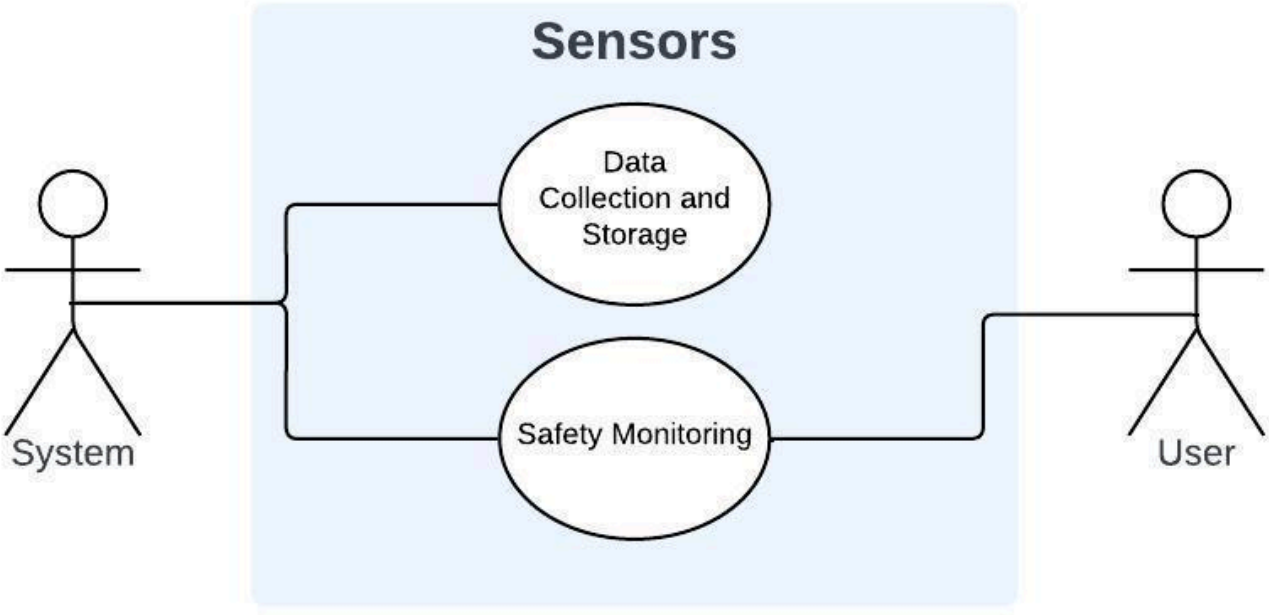
4/27/2025	upload final source code to GitHub under the GNU General Public License (GPLv3)	GitHub Documentation	Code clean-up	Final Portfolio

The primary responsibilities and milestones are laid out in the above table. Bau is primarily responsible for the web interface, Elias for the controller, and Luna for the physical systems and connections with the software. For more detailed weekly tasks, a SCRUM lifecycle approach is used. There are weekly stand-up meetings in which the work from the previous sprint is reviewed and the upcoming work for the next sprint planned and decided upon. The design is open to change as the implementation is fleshed out completely.

The SCRUM approach was chosen because it is fast and allows the work to be broken down into easily manageable weekly chunks. It is adaptive and allows any necessary changes to be easily incorporated into the system.

Functional Requirements

Feature 0 - Sensors



Functional Requirements:

- The system shall be able to read data from various sensors, such as:
    - Temperature sensors (for water).
    - Camera (for viewing the plants)
    - Water level sensors (for grow bed).
    - Water quality sensors (for pH and TDS).
- The system shall update sensor data at regular intervals (e.g., every minute).
- The system shall trigger warnings to the web interface if sensor readings fall outside predefined safe ranges.

User Stories:

1. As a user, I want to monitor the nitrogen levels and temperature so that I can ensure the system is in optimal conditions for both fish and plants.

2. As a user, I want to receive alerts when any sensor reading is out of range so that I can take corrective action.

Acceptance Criteria:

- The system accurately reads and logs sensor data.
- Alerts are sent when values are out of predefined thresholds
- Sensor data should be updated every minute in real time.

Use Case 1: Data Collection and Storage

Actor(s): System, User
Description: The system continuously reads data from various sensors and logs it in real-time. The data is updated every minute and stored for future analysis.

Preconditions:

- Sensors are installed and functioning properly.
- The system is powered on and connected to the local network.

Basic Flow:

1. The system collects data from the sensors.
2. Sensor readings are updated every minute and logged into the system's data file.
3. The system checks if the readings fall within predefined safe ranges for each sensor.
4. If all readings are within safe ranges, the system continues logging data.
5. If any reading falls outside the safe range, the system triggers a warning to be sent to the web interface.

Alternate Flow:

- If there is a failure with a sensor, the system logs an error and notifies the user to take corrective action.

Postconditions:

- Data is logged in the data file and is available for future retrieval.
- Alerts are sent if any sensor readings fall outside of acceptable thresholds.

Use Case 2: Safety monitoring

Actor(s): System, User
Description: The system monitors real-time sensor data and sends alerts to the user when a sensor reading falls outside the predefined safe threshold.

Preconditions:

- Sensors are functional.
- The user has configured acceptable thresholds for each data point.

Basic Flow:

1. The system continuously reads and monitors the data from sensors.
2. The system compares the sensor readings to the predefined threshold values.
3. If any sensor reading is out of range, the system sends a warning to the web interface.
4. The user views the alert on the web interface and can take corrective actions.

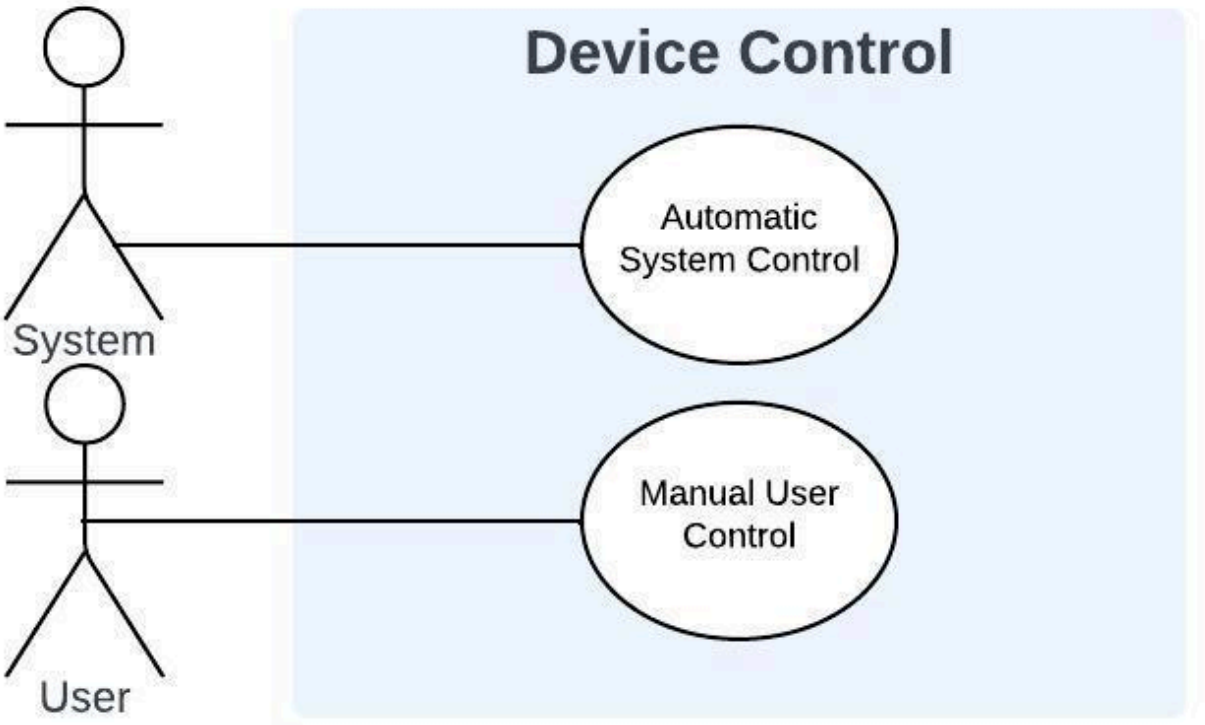
Alternate Flow:

- If the warning is acknowledged by the user, the system logs the corrective action taken.

Postconditions:

- Alerts are generated and sent when readings are outside safe thresholds.
- The system logs the alert for future reference.

Feature 1 - Device Control



Functional Requirements:

- The system shall control devices like pumps, heaters, feeders, and grow lights.
- Devices shall be activated or deactivated based on sensor readings (e.g., turn on aerators if dissolved oxygen is low).
- Manual control of devices shall be available via the web interface.
- The system shall support scheduling for devices.
- Device status (on/off) shall be displayed in real-time on the web interface.

User Stories:

1. As a user, I want to control the water pump automatically to ensure proper water circulation in the system.
2. As a user, I want the system to automatically adjust light settings based on time.

Acceptance Criteria:

- The system responds to sensor inputs in real time and controls devices accordingly.

- Lights can be manually controlled from the web interface.
- Devices can be scheduled (e.g., lights, water pump) to operate at specific times.
- The real-time status of all devices (on/off) is visible on the web interface.

Use Case 1: Automatic System Control

Actor(s): System, User
Description: The system automatically controls the pump based on real-time sensor data.

Preconditions:

- Devices are connected and functional.
- Sensor readings are being updated in real time.
- The system is configured to control devices based on sensor data.

Basic Flow:

1. The system monitors real-time sensor data.
2. If the water level sensor reads that the grow bed is full, it will shut off the water pump for a set amount of time to allow the grow bed to drain.
3. The system logs the device action.
4. The web interface displays the real-time status of whether or not the grow bed is draining or filling.
5. The water pump automatically turns back on after the set time interval.

Alternate Flow:

- If the pump does not shut off automatically, the user is notified.

Postconditions:

- Devices are automatically controlled based on sensor readings.
- Device status is reflected on the web interface.
- All actions are logged in the system for historical review.

Use Case 2: Manual User Control

Actor(s): User, web interface, system

Description: The user can manually control the lights via the web interface.

Preconditions:

- The user is logged into the web interface.
- The smart plug controlling the lights is connected and functioning.
- The system is responsive and operational.

Basic Flow:

1. The user accesses the web interface from any device.
2. The user navigates to the settings page of the web interface.
3. The user can manually turn the lights on or off from the interface.
4. The system immediately updates the device status on the web interface.

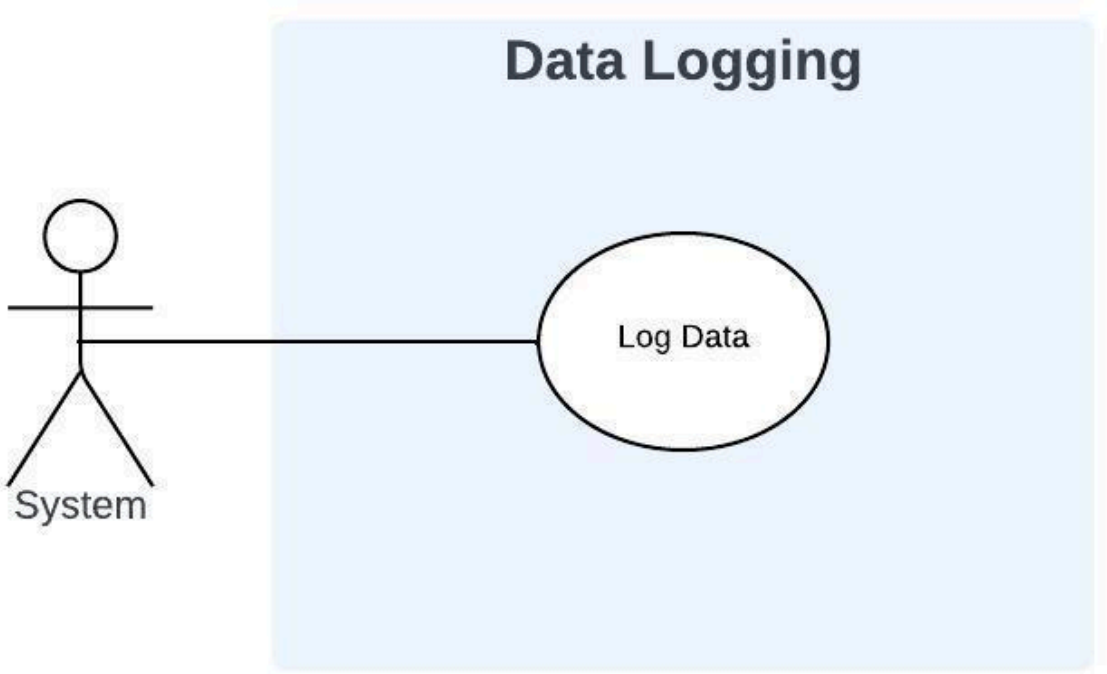
Alternate Flow:

- If the device is not responding, the system displays an error and suggests troubleshooting steps.

Postconditions:

- The device is manually controlled by the user.
- The device status is updated on the web interface.

Feature 2 - Data Logging



Functional Requirements:

- The system shall log all sensor data (temperature, TDS, pH) when the website is loaded.
- The system shall allow users to view historical data.
- The system shall support the export of data to CSV for offline analysis.

User Stories:

1. As a user, I want to view historical sensor data so that I can analyze trends and identify potential issues.
2. As a user, I want the system to analyze data trends and provide insights, such as whether water quality is improving or deteriorating over time.

Acceptance Criteria:

- Sensor data is logged and retrievable for at least 30 days.
- The system highlights sensor readings outside of the acceptable range (e.g., pH goes higher than what the user set it to).
- Data export functionality is available and works as expected.

Use Case 1: Log Data

Actors(s): System

Description: The system automatically collects and stores data in a data file.

Preconditions:

- The sensors are connected and functional.
- Sensors are receiving input.

Basic Flow:

- The system takes sensor readings every time the website is loaded.
- The system prints output to a log file in a csv.

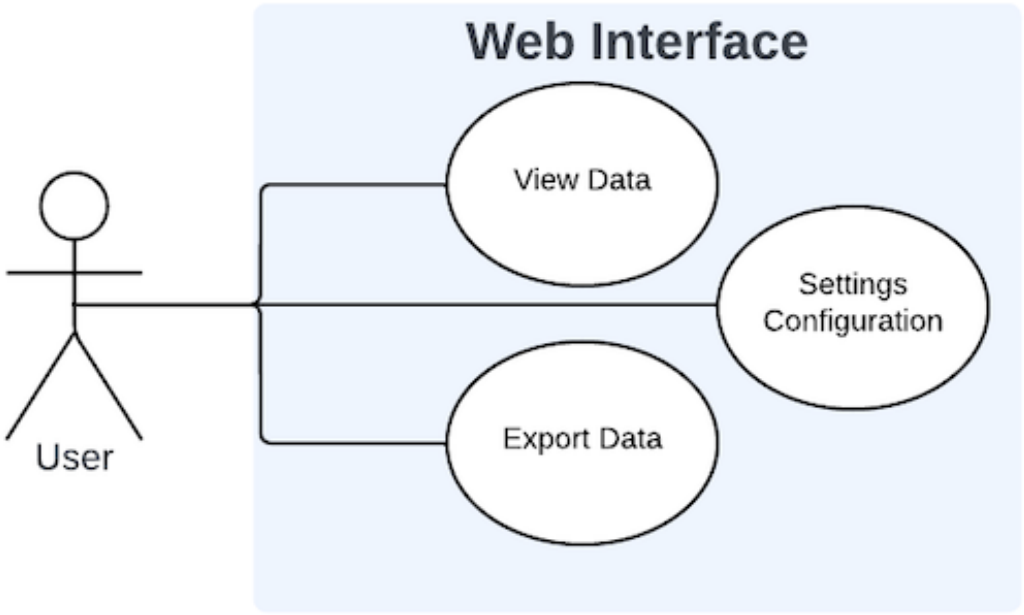
Alternate Flow:

- If a sensor malfunctions, log an error to the data file.

Postcondition:

- There is now a data file with sensor data in it.
- Sensors are ready for more data.

Feature 3 - Web Interface



Functional Requirements:

- The system shall have a web-based interface accessible from any device with a modern browser.
- The interface shall display real-time sensor data and device status.
- The interface shall allow users to manually control devices (i.e., toggle the lights).
- The interface shall allow users to configure thresholds for sensors.
- The interface shall display camera footage of the grow bed.
- The interface shall display warnings if dangerous readings are detected.

User Stories:

1. As a user, I want to access the aquaponics system dashboard from any device so that I can monitor the system remotely.
2. As a user, I want to set safe thresholds for sensors via the web interface to ensure that my system remains in an optimal state.
3. As a user, I want to configure safe thresholds for measurements from the web interface.

Acceptance Criteria:

- The web interface is fully functional across desktop, tablet, and mobile devices.
- Real-time sensor data is displayed on the dashboard.
- Devices can be controlled (on/off) from the interface.
- The interface allows users to configure sensor thresholds.

Use Case 1: View Data

Actor(s): User, web interface
Description: The user can view historical sensor data to analyze trends and identify potential issues with the system.

Preconditions:

- The system has been collecting sensor data for at least a predefined period.
- Data logging is active, and historical data is stored in the database.

Basic Flow:

1. The user logs into the web interface and navigates to the "Logs" section.
2. The system displays historical sensor data.
3. The user can view trends of pH, temperature, and TDS.

Postconditions:

- The user views the historical data and trends.

Use Case 2: Export Data

Actor(s): User, web interface, system
Description: The user can export historical sensor data to a CSV file for offline analysis or reporting.

Preconditions:

- Historical data is available for export.
- The user is logged into the web interface.

Basic Flow:

1. The user navigates to the "Logs" section of the web interface.
2. The user chooses to download the logs as a CSV.
3. The system compiles the data into a CSV file.
4. The user downloads the file for offline analysis or reporting.

Postconditions:

- The user successfully downloads the sensor data in CSV format.

Use Case 3: Configuration

Actor(s): User, web interface
Description: The user can configure the safe operating thresholds for each sensor through the web interface.

Preconditions:

- The user is logged into the web interface.
- The system has default threshold values set for each sensor.

Basic Flow:

1. The user navigates to the "Settings" section of the web interface.
2. The user can view and adjust the threshold values to the desired limits for any measured value in the system, as well as toggle the lights on or off.
3. The system saves the new thresholds and updates its monitoring accordingly.
4. The system immediately begins comparing sensor readings to the newly configured thresholds.

Alternate Flow:

- If the user attempts to set a threshold outside the allowed range, the system displays an error message and suggests a valid range.

Postconditions:

- The sensor thresholds are updated and stored in the system.
- Alerts are generated if sensor readings exceed the new thresholds.

Non-Functional Requirements

Performance:

- The system shall update sensor data and control devices in real time (within 1 second for device control).
- The web interface shall load within 3 seconds.

Scalability:

- The system should support additional sensors and devices as needed (e.g., additional tanks or grow beds).
- The database should be capable of storing large amounts of historical data (at least a month).

Reliability:

- The system shall be highly available, with a target uptime of 99.9%.
- The system should be able to function outdoors in moderate weather.

Usability:

- The web interface should be intuitive and user-friendly, requiring no special training to use effectively.
- The dashboard should provide clear visual feedback on the system’s current state.

Interface Requirements

The OpenAquaponics system utilizes numerous interfaces to facilitate its operation.

In terms of physical interfaces, the system utilizes mechanical, electrical, and electronic interfaces.

Mechanically, the system uses a pump to transfer water from the fish tank into the plant growth bed, and a bell siphon to transfer water back to the fish tank once the bed fills to a certain threshold. A feeding mechanism ensures the fish are fed adequately to ensure optimal production of the beneficial nitrates that assist with plant growth.

Electrically, the system runs off of standard 110-volt “household-style” electrical service to ensure maximum compatibility. The Raspberry Pi Model 3B+ single-board computer will receive constant power for maximum uptime. The grow lamp and water pump will be controlled by Tasmota smart plugs to ensure they are operating on a set schedule. The aforementioned smart plugs will be controlled by a script on the Raspberry Pi, utilizing the Python requests library for sending commands.

Electronically, the system will make extensive use of the Raspberry Pi’s GPIO pins and other interfaces. The camera that provides remote monitoring of the plants will be connected to one of the Pi’s USB ports. The sensor array that monitors water level, pH, temperature, and total dissolved solids will be connected as inputs to the GPIO pins.

In terms of functional interfaces, the system utilizes information transfer, human-computer, maintenance, and installation interfaces.

For information transfer, the system will utilize the Raspberry Pi’s onboard WiFi connection and a self-hosted HTML5 web interface with CSS to provide remote monitoring of the system. The sensor data will be read in via a Python script and

transmitted to the web interface via the one-wire protocol, digital input, and Serial Peripheral Interface (SPI) to provide live readings of the sensors in the system. The camera will also be viewable through the web interface using the Motion utility. The Tasmota smart plugs are connected to the WiFi as well, to ensure they can receive commands from the Pi. All source code will be published to GitHub and open-sourced via the GNU General Public License V3.

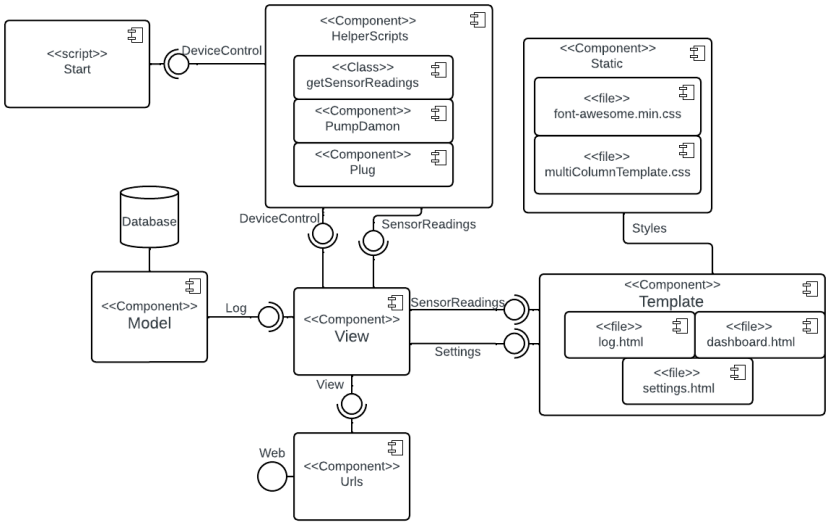
For human-computer interaction, the system will provide the above-mentioned HTML5 web interface to monitor sensors and the camera in real time.

For maintenance interactions, the system will provide documentation on tolerances of variables the sensors can read and will provide alerts in the web interface if any readings are outside of the tolerance.

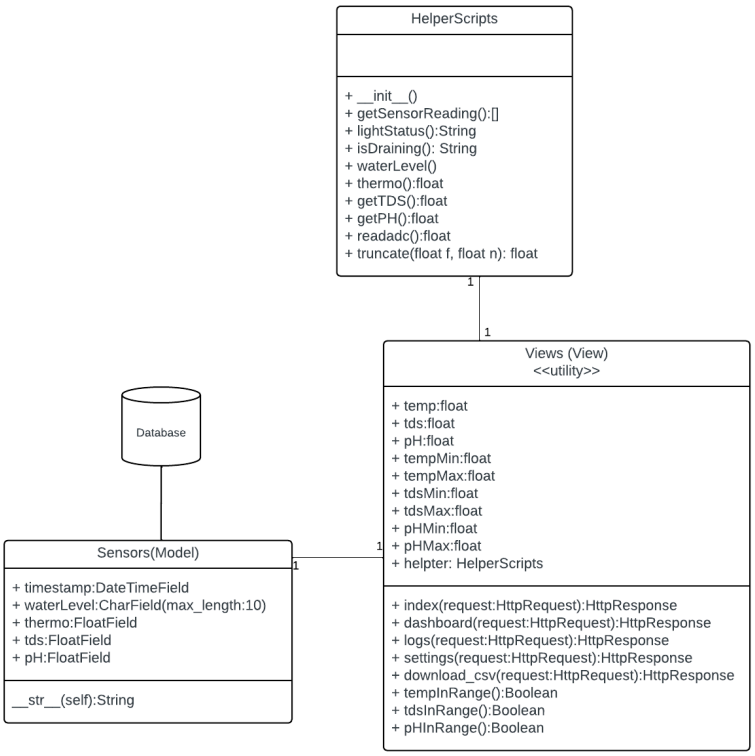
For installation, documentation will be provided for the parts required to build the system, how they are assembled, and how to install and configure the requisite software.

Architectural Design Documents

Component Diagram



Class Diagram



Design Pattern

The Model-View-Template design pattern (the Django version of the Model-View-Controller design pattern) was used to aid in the design and implementation. The model is the database model of the aquaponic system, containing the sensor readings, the view houses the automated care system and facilitates communication between the sensor readings, model, and template, and the template is the web interface that allows the user to view and control the system.

This pattern was chosen over possible alternatives because it is the common choice when establishing a web interface. It allows the distinct pieces of the project to be separated into different classes that can be created individually and easily reused.

Development Standards

This project aims to adopt certain development standards to produce a consistent, easy-to-use product for the end user.

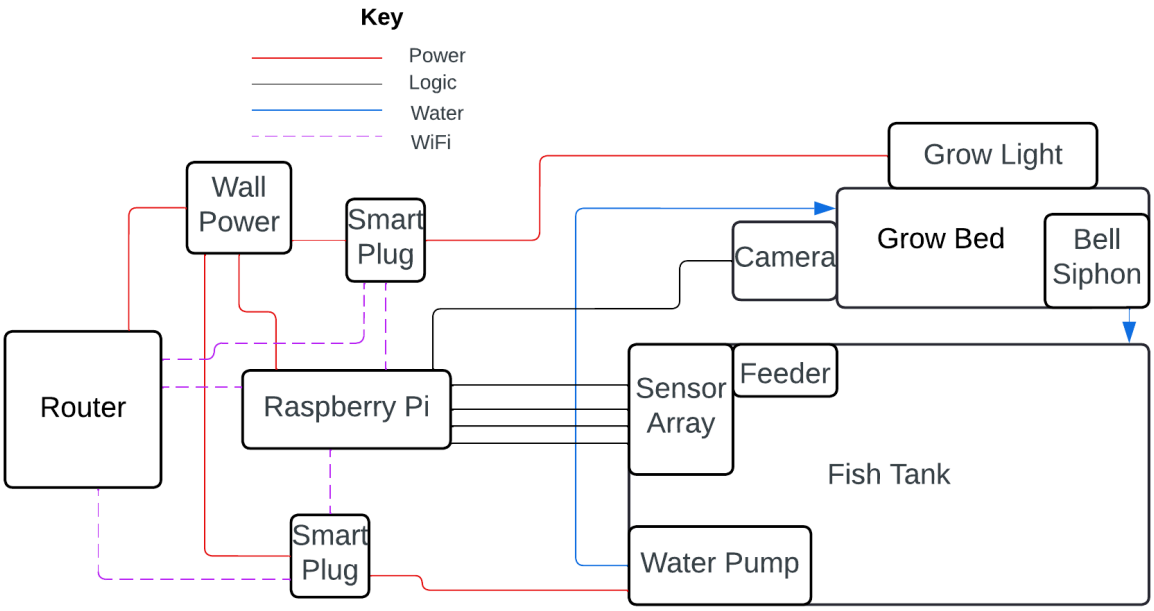
The codebase for this project will primarily be written in Python, with a script for handling sensor input in real-time, with secondary scripts running in the background for the remote-controlled devices (water pump and grow lamp) within the system. The web interface will utilize the Python web framework Django to host a collection of HTML5 web pages with corresponding CSS that provide a live camera feed of the system and live data from the sensors, data logs, and settings for acceptable sensor ranges. All code will follow similar layouts and best practices for indentation, spacing, naming, and other formatting.

Version control for this project will be managed via Git and hosted on GitHub, utilizing branching to add and test new features and components.

Project management will utilize the communication platform Discord, and will utilize a calendar built in Google Sheets to ensure project goals are met on time.

Open-sourcing of this project will be done under and in accordance with the GNU General Public License, V3, to ensure this project can be maintained and distributed freely long-term.

Expected Results



The physical system should match the above diagram, with the addition of any wanted filters or water stones to help maintain the health of the fish, plants, and tank. The Raspberry Pi should use the provided source code, a software system to pull sensor readings, control devices, and host a web interface to allow users to view sensor information and control the system. The web interface should be available to view on a browser while connected to the same local network as the Pi.

Overall, the system should function to aid a person in maintaining the health of plants and fish. Users will likely see improvements in plant yields compared to traditional growing techniques and manually-controlled aquaponics systems. The system should be easy to maintain and repair, attain replacement parts for, and to customize for the end-user's specific needs and hardware, which will all be laid out in the documentation on the project's GitHub repository. The code and schematics will be open-sourced via

the GNU GPLv3 and made available on GitHub to allow future development and improvements to the project by the broader community.

Parts List & Cost Breakdown

Part	Link	Cost (USD)
Raspberry Pi Model 3B+	https://www.adafruit.com/product/3775?src=raspberrypi	$35.00
CPU heatsink	https://www.adafruit.com/product/3082	$1.95
Pi Case (3D printed, use 40mm fan opening for best clearance w/ heatsink)	https://www.thingiverse.com/thing:3810243/files	Print yourself
Jumper wires (female/male)	https://www.adafruit.com/product/826	$3.95
10 Gallon Aquarium	https://www.petsmart.com/fish/tanks-aquariums-and-nets/aquariums/marineland-open-glass-aquarium-5356063.html?gclsrc=aw.ds&gad_source=1&gbraid=0AAAADiLNNnzyld623w-dmkx-lz_nq3Lc&gclid=Cj0KCQjwzrzABhD8ARIsANlSWNM30Z6B7xQ6GmdM-4gMZVzZWGcK7pRkica9kVsGfKPm2K5mpDFJ194aArv1EALw_wcB	$24.99
Water pump	https://www.amazon.com/AQUANEAT-Submersible-Fountain-Aquarium-Hydroponics/dp/B0BJJBK746?source=ps-sl-shoppingads-lpcontext&ref_=fplfs&smid=A1W7R5UJZCLN8X&gQT=1&th=1	$6.99
28Qt Storage bin (for grow bed)	https://www.walmart.com/ip/Sterilite-28-Qt-Storage-Box-Plastic-Adult-White/8870703312?wmlspartner=wlpa&selectedSellerId=0&selectedOfferId=FF183E88C9CA39B4A4B2D814CCC3980E&conditionGroupCode=1&wl13=2084&gclsrc=aw.ds&adid=22222222278870703312_117755028669_12420145346&wl0=&wl1=g&wl2=c&wl3=501107745824&wl4=pla-394283752452&wl5=1020859&wl6=&wl7=&wl8=&wl9=pla&wl10=8175035&wl11=local&wl12=8870703312&veh=sem_LIA&gclsrc=aw.ds&gad_source=1&gbraid=0AAAADmfBIot9tNnrQxP6tmTIIBsl-dbN&gclid=Cj0KCQjwzrzABhD8ARIsANlSWNOC3akGsBSdaKQOrIXaCstIlJ9j1KJ1bq2Ika4fENxc384jm8ZY4rsaAoTkEALw_wcB&sid=d4813fb1-2b6f-4cce-947b-66f733f36273	$5.98

Smart plugs w/ Tasmota firmware (x2)	https://cloudfree.shop/product/sonoff-s31/	$13.00 (x2)
Autofeeder	https://www.petsmart.com/fish/food-and-care/feeders/top-fin-fin-automatic-fish-feeder-5119522.html?gStoreCode=1033&gQT=1	$29.99
Bell siphon	https://www.amazon.com/Aquaponics-Smaller-Kitchen-Worldwide-Confidence/dp/B01AITYOTU?mcid=f4a3ff9efabb341cb88766cc756389e9&tag=hyprod-20&linkCode=df0&hvadid=693770257565&hvpos=&hvnetw=g&hvrand=6227970283409966986&hvpone=&hvptwo=&hvqmt=&hvdev=c&hvdvcmdl=&hvlocint=&hvlocphy=1020859&hvtargid=pla-1951178709028&psc=1	$24.00
Power strip - 2 pack	https://www.homedepot.com/p/6-Outlet-Power-Strip-with-4-ft-Cord-Right-Angle-Plug-2-Pack-YLPT-90B/303319020	$9.98
Grow lamp (some modification required to bypass timer circuitry)	https://www.homedepot.com/p/Bell-Howell-Bionic-Grow-5-Watt-Equivalent-Indoor-LED-Full-Spectrum-UV-Flexible-Plant-Grow-Light-in-Color-Changing-Lights-8717/322309060?source=shoppingads&locale=en-US&gQT=1	$29.99
Water temp sensor	https://www.adafruit.com/product/381	$9.95
pH sensor	https://www.mouser.com/ProductDetail/DFRobot/SEN0161?qs=Zcin8yvlhnPq1OJbBVjoIw%3D%3D&mgh=1&gQT=1	$29.50
TDS sensor	https://www.dfrobot.com/product-1662.html	$11.80
Water Level sensor	https://store-usa.arduino.cc/products/gravity-photoelectric-water-liquid-level-sensor-for-arduino?pr_prod_strat=e5_desc&pr_rec_id=25b78afbc&pr_rec_pid=6692751409359&pr_ref_pid=6643395592399&pr_seq=uniform	$6.90
Basic USB webcam	https://www.bestbuy.com/site/logitech-c270-720-webcam-with-noise-reducing-mics-black/9928354.p?skuId=9928354&gQT=1	$20.99
MCP3008 ADC	https://www.adafruit.com/product/856	$4.50
Small breadboard	https://www.adafruit.com/product/65	$3.95
Apple AirPort Extreme Router (used)	https://www.ebay.com/itm/226504500970?mkcid=16&mkevt=1&mkrid=711-127632-2357-0&ss	$17.95

	spo=Nm_asfSaQMe&sssrc=2047675&ssuid=3s_ymp3bq1c&widget_ver=artemis&media=COPY	
TOTAL (before taxes & shipping)		$274.37

Some assembly is required. Users will need to provide their own computer for interfacing with the Pi over SSH to start the software, drill with ⅝” hole saw for installing the bell siphon, and adhesives for mounting sensors and tubing. More details on configuring the system are available in the documentation on GitHub.

Appendix - Source Code

All code and documentation is available in our GitHub repository at https://github.com/bauwow5/OpenAquaponics/. Code written by us is as follows:

models.py

```
from django.db import models
from django.utils import timezone
class Sensors(models.Model):
    timestamp = models.DateTimeField()
    waterLevel = models.CharField(max_length=10)
    thermo = models.FloatField(default=0)
    tds = models.FloatField(default=0)
    pH = models.FloatField(default=0)

    def __str__(self):
        now = self.timestamp
        date_time = now.strftime("%Y-%m-%d %H:%M:%S") # Year-Month-Day Hours:Minutes:Seconds
        return date_time
```

urls.py

```
from django.urls import path
from . import views
urlpatterns = [
    path("", views.index, name="index"),
    path("settings/", views.settings, name="settings"),
    path("logs/", views.logs, name="logs"),
]
```

start.sh:

```
python3 pumpDamon.py &
pumpProcess=$!
sudo motion on
motionProcess=$!
cd ../../
sudo python3 manage.py runserver 10.0.1.6:8000 &
```

```
serverProcess=$!
echo $pumpProcess
echo $motionProcess
echo $serverProcess
```

getSensorReadings.py:

```
import spidev
import sys
import RPi.GPIO as GPIO
import Adafruit_DHT
import requests
import json

spi = spidev.SpiDev()
spi.open(0, 0)
spi.max_speed_hz = 1000000

class HelperScripts:

    def __init__(self):
        print("init")

    def getSensorReading(self):
        sensor_array =[]
        sensor_array.append(self.isDraining())
        sensor_array.append(self.thermo())
        sensor_array.append(self.getTDS())
        sensor_array.append(self.getPH())
        print(sensor_array)
        return sensor_array

    def lightStatus(self):
        request =requests.get('http://10.0.1.2/cm?cmnd=Power')
        jsonRequest =request.json()
        status = jsonRequest.get('POWER')
        return status
```

```
def isDraining(self):
    try:
        request =requests.get('http://10.0.1.3/cm?cmnd=Power')
        jsonRequest =request.json()
        status = jsonRequest.get('POWER')
        if status =="ON":
            return "Filling"
        else:
            return "Draining"
    except:
        return 0
def waterLevel(self):
    GPIO.setmode(GPIO.BCM)
    GPIO.setup(17,GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
    waterLine = (GPIO.input(17) ==GPIO.HIGH)
    return waterLine

#28-0000103f8d52/temperature"
def thermo(self):
    try:
        temperature = open("/sys/bus/w1/devices/28-0000103f8d52/temperature","r")
        temp = temperature.read().replace("\n","")
        temperature.close()
#conversion to degrees celsius from 1/1000's of degrees celsius
        temperature = int(temp)/1000
        return temperature
    except:
        return 0

def getPH(self):
    try:
        Value = self.readadc(0)
        voltage = (Value * 5.0) / 1023  # Corrected for 5V MCP3008
        phvalue = 7 - (voltage - 1.85) * 3.5  # Adjusted neutral voltage
        phvalue = self.truncate(phvalue,2)
```

```
            return phvalue
        except:
            return 0


 def readadc(self,adcnum):
        if adcnum < 0 or adcnum > 7:
            return -1
        r = spi.xfer2([1, (8 + adcnum) << 4, 0])
        adcout = ((r[1] & 3) << 8) + r[2]
        return adcout
    # Read pH sensor on Channel 0
    def getTDS(self):
    # Read TDS sensor on Channel 1
        try:
            tds_adc = self.readadc(1)
            tds_voltage = (tds_adc * 5.0) / 1023  # Convert ADC to voltage
    # Convert voltage to TDS (from DFRobot's formula)
            tds_value = (133.42 * tds_voltage**3 - 255.86 * tds_voltage**2 + 857.39 * tds_voltage) * 0.5
    #truncates the value
            tds_value =self.truncate(tds_value,2)
            return tds_value
        except:
            return 0
#this is a helper method to truncate floating point numbers
    def truncate(self,f, n):
        s = '%.12f' % f
        i, p, d = s.partition('.')
        return '.'.join([i, (d+'0'*n)[:n]])
```

plug.py:

```
import requests
import time
def growLamp():
    requests.get('http://10.0.1.2/cm?cmnd=Power%20TOGGLE') # Grow Lamp
def waterPump():
    requests.get('http://10.0.1.3/cm?cmnd=Power%20TOGGLE') # Water Pump
```

pumpDamon.py:

```
import requests
import time
from getSensorReadings import HelperScripts
from plug import  waterPump, growLamp

def main():
    helper = HelperScripts()
    while True:
        if(helper.waterLevel()):
            time.sleep(15)
            waterPump()
            time.sleep(180)
            waterPump()
main()
```

views.py:

```
from django.db.models import F
from django.shortcuts import render
from .models import Sensors
import sys
from django.utils import timezone
import csv
from django.http import HttpResponse
sys.path.insert(0,'/home/oa-user/openAquaponics/oa/helperScripts/')
import getSensorReadings
from getSensorReadings import HelperScripts
from plug import growLamp
  #default instance variables
temp=0
tds=0
pH=0
   #default mins and maxs for formatting/alerts
tempMin=0
tempMax=0
tdsMin=0
tdsMax=0
pHMin=0
```

```
pHMax=0
helper = getSensorReadings.HelperScripts()
    #helper methods
def tempInRange():
    if tempMin <temp and temp<tempMax:
        return True
    else:
        return False

def tdsInRange():
    if float(tdsMin) <float(tds) and float(tds)<float(tdsMax):
        return True
    else:
        return False

def pHInRange():
    if float(pHMin) <float(pH) and float(pH)<float(pHMax):
        return True
    else:
        return False
    #views
def index(request):
        #pass a variable to the html like this vvv
    global temp, tds, pH
    r = helper.getSensorReading()
    readings = Sensors(waterLevel=r[0], thermo=r[1], tds=r[2], pH=r[3],     timestamp=timezone.now())
    readings.save()
    readings.waterLevel = r[0]
    readings.thermo = r[1]
    temp = r[1]
    readings.tds = r[2]
    tds = r[2]
    readings.pH = r[3]
    pH = r[3]
    readings.timestamp =timezone.now()
        #need most recent sensor reading here
```

```
    tempIR = tempInRange()
    tdsIR = tdsInRange()
    pHIR = pHInRange()
    #have to import from Helper Scripts
    light_status = helper.lightStatus()
    context = {"sensor_reading":readings, "tempIR":tempIR, "tdsIR":tdsIR, "pHIR":pHIR,
"lightStatus":light_status}
    return render(request, "oa/index.html", context)

def settings(request):
        # Default values
    light_status = False
    global tempMin, tempMax, tdsMin, tdsMax, pHMin, pHMax
    if request.method == 'POST':
            # Handle light toggle
        if request.POST.get('light_status') == 'on':
            print("Light ON")
            light_status = True
            growLamp()  # Activate grow lamp
        else:
            print("Light OFF")
            light_status = False
            # Get numeric form values (safe cast with fallback to None or 0)
        try:
            tempMin = float(request.POST.get('tempMin', 0))
            tempMax = float(request.POST.get('tempMax', 0))
            tdsMin = float(request.POST.get('tdsMin', 0))
            tdsMax = float(request.POST.get('tdsMax', 0))
            pHMin = float(request.POST.get('pHMin', 0))
            pHMax = float(request.POST.get('pHMax', 0))
        except ValueError:
            print("Invalid input encountered.")
        print("Temperature Min/Max:", tempMin, tempMax)
        print("TDS Min/Max:", tdsMin, tdsMax)
        print("pH Min/Max:", pHMin, pHMax)
    context = {
```

```
            "light_status": light_status,
            "tempMin": tempMin,
            "tempMax": tempMax,
            "tdsMin": tdsMin,
            "tdsMax": tdsMax,
            "pHMin": pHMin,
            "pHMax": pHMax
    }
    return render(request, "oa/settings.html", context)

def download_csv(request):
    response =HttpResponse(content_type='text/csv')
    writer = csv.writer(response)
    response.write(u'\ufeff'.encode('utf8'))
    writer.writerow(["Time & Date","Current Status","Temperature","TDS","PH"])
    for row in Sensors.objects.all():
        List = [row.timestamp,row.waterLevel,row.thermo,row.tds,row.pH]
        writer.writerow(List)
    return response

def logs(request):
    if request.GET.get('download') == 'csv':
            # Create the HttpResponse object with CSV headers.
        responce = download_csv(request)
        return responce
    # regular logs rendering
    #need most recent sensor reading here
    times = Sensors.objects.order_by('-id').values_list("timestamp",flat=True)
    waterlevels = Sensors.objects.order_by('-id').values_list("waterLevel",flat=True)
    temps = Sensors.objects.order_by('-id').values_list("thermo",flat=True)
    tds = Sensors.objects.order_by('-id').values_list("tds",flat=True)
    pHs = Sensors.objects.order_by('-id').values_list("pH",flat=True)
    #sensor_reading = get_object_or_404(Sensors)
    context =
{"timestamp":times,"waterlevels":waterlevels,"temps":temps,"tds":tds,"pHs":pHs,"tempInRange":tempInRa
nge(),"tdsInRange":tdsInRange(),"phInRange":pHInRange()}
```

```
    return render(request, "oa/logs.html", context)
```

index.html:

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>Dashboard - OpenAquaponics</title>
{% load static %}
<link rel="stylesheet" href="{% static 'oa/multiColumnTemplate.css' %}">
</head>
<body>
<div class="container">
  <header>
    <a id="top"></a>
    <div class="primary_header">
      <h1 class="title">OpenAquaponics</h1>
    </div>
    <nav class="secondary_header" id="menu">
      <ul>
        <li><a href="{% url 'index' %}">DASHBOARD</a></li>
        <li><a href="{% url 'logs' %}">LOGS</a></li>
        <li><a href="{% url 'settings' %}">SETTINGS</a></li>
      </ul>
    </nav>
  </header>
  <section>
    <h2 class="noDisplay">Hello</h2>
    <article class="left_article">
      {% if tempIR and tdsIR and pHIR %}
      <h3 class=dashboard_overview_good style="color: #8AC860;">Everything is going swimmingly
well!</h3>
      {% else %}
      <h3 class=dashboard_overview_bad style="color: #DE1D20;">Something is fishy!</h3>
```

```
    {% endif %}
   </article>
   <!--<aside class="right_article"><iframe width="100%" height="100" src="http://10.0.1.4:8081/" alt=""
class="placeholder"></iframe><!-- </aside>-->
   <img id="image" src="http://10.0.1.6:8081/0/stream"></img>
  </section>
  <div class="row">
   <div class="columns">
              <p style=" font-size: 30px;
   color: #8AC860;
   text-align: center;"> {{sensor_reading.waterLevel }} </p>
    <h4>Water Status</h4>
   </div>
   <div class="columns">
      {% if tempIR %}
         <p style="
   font-size: 30px;
   color: #8AC860;
   text-align: center;"> {{sensor_reading.thermo}} </p>
      {% else %}
         <p style="
   font-size: 30px;
   color: #DE1D20;
   text-align: center;"> {{sensor_reading.thermo}} </p>
      {% endif %}
    <h4>Temperature (&deg;C)</h4>
   </div>
   <div class="columns">
      {% if tdsIR %}
         <p style="
   font-size: 30px;
   color: #8AC860;
   text-align: center;"> {{sensor_reading.tds}} </p>
      {% else %}
         <p style="
   font-size: 30px;
```

```
    color: #DE1D20;
    text-align: center;"> {{sensor_reading.tds}} </p>
        {% endif %}
      <h4>Total Disolved Solids (ppm)</h4>
    </div>
    <div class="columns">
                {% if pHIR %}
            <p style="
    font-size: 30px;
    color: #8AC860;
    text-align: center;"> {{sensor_reading.pH}} </p>
        {% else %}
            <p style="
    font-size: 30px;
    color: #DE1D20;
    text-align: center;"> {{sensor_reading.pH}} </p>
        {% endif %}
      <h4>pH</h4>
    </div>
  </div>
  <div class="row blockDisplay">
    <div class="column_half left_half">
      {% if lightStatus == "ON" %}
      <h2 class="column_title" style="background-color:green;height:100px;margin:auto;">Lights -
                        On</h2>
      {% else %}
      <h2 class="column_title" style="background-color:red;padding:50px;">Lights -
        Off</h2>
      {% endif %}
    </div>
  </div>
  <footer class="secondary_header footer">
    <div class="copyright"><a href="#top">Back to top</a></div>
  </footer>
</div>
</body>
```

```
</html>
```

logs.html:

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>Logs - OpenAquaponics</title>
{% load static %}
<link rel="stylesheet" href="{% static 'oa/multiColumnTemplate.css' %}">
<link href="multiColumnTemplate.css" rel="stylesheet" type="text/css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
</head>
<body>
</style>
<div class="container">
  <header>
    <a id="top"></a>
    <div class="primary_header">
      <h1 class="title">OpenAquaponics</h1>
    </div>
    <nav class="secondary_header" id="menu">
      <ul>
        <li><a href="{% url 'index' %}">DASHBOARD</a></li>
        <li><a href="{% url 'logs' %}">LOGS</a></li>
        <li><a href="{% url 'settings' %}">SETTINGS</a></li>
      </ul>
    </nav>
  </header>
  <section>
    <h2 class="noDisplay">Hello</h2>
    <article class="article">
      <h3>Sensor Log</h3>
            <a href="?download=csv" class="btn"><i class="fa fa-download"></i> Download as csv</a>
```

```
        <div class="row">
        <div class="columns1">
        <h4>Timestamp</h4>

        {%for i in timestamp%}
        <p class = "reading">{{i}}</p>
        {%endfor%}
        </div>
<div class="columns1">
  <h4>Water Level</h4>
        {%for i in waterlevels%}
        <p class="reading">{{i}}</p>
        {%endfor%}
</div>
<div class="columns1">
  <h4>Temperature (&deg;C) </h4>
        {%for i in temps%}
        {%if tempInRange%}
                <p class="reading">{{i}}</p>
        {%else%}
                <p class="reading"> {{i}}</p>
        {%endif%}

        {%endfor%}
</div>
<div class="columns1">
  <h4>Total Disolved Solids (ppm)</h4>
        {%for i in tds%}
        {%if tdsInRange%}
                <p class="reading">{{i}}</p>
        {%else%}
                <p class="reading"> {{i}}</p>
        {%endif%}

  {%endfor%}
</div>
```

```
    <div class="columns1">
      <h4>pH</h4>
            {%for i in pHs%}
            {%if phInRange%}
            <p class="reading">{{i}}</p>
            {%else%}
            <p class="reading"> {{i}}</p>
            {%endif%}
            {%endfor%}
    </div>
            </div>
    </article>
  </section>
  <footer class="secondary_header footer">
    <div class="copyright"><a href="#top">Back to top</a></div>
  </footer>
</div>
</body>
</html>
```

settings.html:

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>Settings - OpenAquaponics</title>
{% load static %}
<link rel="stylesheet" href="{% static 'oa/multiColumnTemplate.css' %}">
<link href="multiColumnTemplate.css" rel="stylesheet" type="text/css">
<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
</head>
<body>
<div class="container">
```

```
 <header>
  <a id="top"></a>
  <div class="primary_header">
   <h1 class="title">OpenAquaponics</h1>
  </div>
  <nav class="secondary_header" id="menu">
   <ul>
    <li><a href="{% url 'index' %}">DASHBOARD</a></li>
    <li><a href="{% url 'logs' %}">LOGS</a></li>
    <li><a href="{% url 'settings' %}">SETTINGS</a></li>
   </ul>
  </nav>
 </header>
 <section>
  <h2 class="noDisplay">Settings</h2>
  <article class="article">
   <h3>Settings</h3>
   <form method="post">
    {% csrf_token %}
    <div class="row">
     <div class="columns3">
      <strong>Lights</strong>
      <label class="switch">
       <input type="checkbox" name="light_status" value="on" {% if light_status %}checked{% endif
%}>
       <span class="slider round"></span>
      </label>
     </div>
    </div>
    <div class="row">
     <div class="columns2">
      <h4>Temperature (&deg;C)</h4>
      <label for="minTemp">Minimum:</label>
      <input type="number" id="minTemp" name="tempMin" value="{{ tempMin }}" min="-99"
max="200"><br>
      <label for="maxTemp">Maximum:</label>
```

```
          <input type="number" id="maxTemp" name="tempMax" value="{{ tempMax }}"  min="-99"
max="200">
        </div>
        <div class="columns2">
          <h4>Total Dissolved Solids (ppm)</h4>
          <label for="minTDS">Minimum:</label>
          <input type="number" id="minTDS" name="tdsMin" value="{{ tdsMin }}" min="0"
max="9000000"><br>
          <label for="maxTDS">Maximum:</label>
          <input type="number" id="maxTDS" name="tdsMax" value="{{ tdsMax }}"  min="0"
max="9000000">
        </div>
        <div class="columns2">
          <h4>pH</h4>
          <label for="minPH">Minimum:</label>
          <input type="number" id="minPH" name="pHMin" value="{{ pHMin }}" min="0" max="14"
step="0.1"><br>
          <label for="maxPH">Maximum:</label>
          <input type="number" id="maxPH" name="pHMax" value="{{ pHMax }}" min="0" max="14"
step="0.1">
        </div>
      </div>
      <article class="article1">
        <button class="btn" type="submit">Save Changes</button>
      </article>
    </form>
  </article>
 </section>
 <footer class="secondary_header footer">
  <div class="copyright"><a href="#top">Back to top</a></div>
 </footer>
</div>
</body>
</html>
```

font-awesome.min.css

```
@charset "UTF-8";
```

```
.reading_good{
	color: #8AC860;
}
.reading_bad{
	color: #DE1D20;
}
.reading{
	color: black;
}
.btn {
  background-color: #52bad5;
  border: none;
  color: white;
  padding: 12px 30px;
  cursor: pointer;
  font-family: "Source Sans Pro";
}

/* Darker background on mouse-over */
.btn:hover {
  background-color: #2EA0BE;
}

.container {
	background-color: #FFFFFF;
	width: 90%;
	margin-left: auto;
	margin-right: auto;
	border-bottom-width: 0px;
	padding-left: 0px;
	padding-top: 0px;
	padding-right: 0px;
	padding-bottom: 0px;
}
.row {
	width: 100%;
```

```
        margin-top: 0px;
        margin-right: 0px;
        margin-bottom: 0px;
        margin-left: 0px;
        padding-top: 0px;
        padding-right: 0px;
        padding-bottom: 0px;
        padding-left: 0px;
        display: inline-block
}
.row.blockDisplay {
        display: block;
}
.column_half {
        width: 100%;
        float: left;
        margin-top: 0px;
}
.columns {
        width: 25%;
        float: left;
        font-family: "Source Sans Pro";
        color: #A5A5A5;
        line-height: 24px;
        padding-top: 10px;
        padding-bottom: 10px;
        text-align: justify;
        margin-top: 15px;
        margin-bottom: 15px;
        padding-left: 0px;
        padding-right: 0px;
        margin-left: 0px;
        margin-right: 0px;
}

.columns1 {
```

```
        width: 20%;
        float: left;
        font-family: "Source Sans Pro";
        color: #A5A5A5;
        line-height: 24px;
        padding-top: 0px;
        padding-bottom: 0px;
        text-align: justify;
        margin-top: 0px;
        margin-bottom: 0px;
        padding-left: 0px;
        padding-right: 0px;
        margin-left: 0px;
        margin-right: 0px;
}

.columns2 {
        width: 33.3%;
        float: left;
        font-family: "Source Sans Pro";
        color: #A5A5A5;
        line-height: 24px;
        padding-top: 0px;
        padding-bottom: 0px;
        text-align: justify;
        margin-top: 0px;
        margin-bottom: 0px;
        padding-left: 0px;
        padding-right: 0px;
        margin-left: 0px;
        margin-right: 0px;
}

.columns3 {
        width: 100%;
        font-family: "Source Sans Pro";
```

```
        color: #717070;
        font-size: 17px;
        line-height: 34px;
        padding-top: 0px;
        padding-bottom: 0px;
        text-align: center;
        margin-top: 0px;
        margin-bottom: 0px;
        padding-left: 0px;
        padding-right: 0px;
        margin-left: 0px;
        margin-right: 0px;
}

.row .columns p {
        padding-left: 10%;
        padding-right: 10%;
}
.container .columns h4 {
        text-align: center;
        color: #01B2D1;
}

.container .columns1 h4 {
        text-align: center;
        color: #01B2D1;
}
.container .columns1_bad h4 {
        text-align: center;
        color: #01B2D1;
}

.container .columns1 {
        text-align: center;
        color: #8AC860;
}
```

```
.container .columns2 {
	text-align: center;
	color: #717070;
}

.container .columns1_bad{
	text-align: center;
	color: #DE1D20;
}

.timestamp {
	color: #717070;
}

.primary_header {
	width: 100%;

	backgroud-image:
url("https://cdn.pixabay.com/animation/2023/10/03/06/47/06-47-59-83_512.gif");
	background-color: #52bad5;

	padding-top: 10px;
	padding-bottom: 10px;
	clear: left;
	border-bottom: 2px solid #2C9AB7;
}
.secondary_header {
	width: 100%;
	padding-top: 50px;
	padding-bottom: 50px;
	background-color: #B3B3B3;
	clear: left;
}

.dashboard_overview{
```

```
        color: #DE1D20;
}ew
.container .secondary_header ul {
        margin-top: 0%;
        margin-right: auto;
        margin-bottom: 0px;
        margin-left: auto;
        padding-top: 0px;
        padding-right: 0px;
        padding-bottom: 15px;
        padding-left: 0px;
        width: 100%;
}
.secondary_header ul li {
        list-style: none;
        float: left;
        margin-right: auto;
        margin-top: 0px;
        font-family: "Source Sans Pro";
        font-weight: normal;
        color: #FFFFFF;
        letter-spacing: 1px;
        margin-left: auto;
        text-align: center;
        width: 33%;
        transition: all 0.3s linear;
}
.secondary_header ul li:hover {
        color: #717070;
        cursor: pointer;
}

.article {
        background-color: #FFFFFF;
        width: 100%;
        float: left;
```

```
        font-family: "Source Sans Pro";
        color: #343434;
        padding-bottom: 15px;
        text-align: center;
}

.article1 {
        background-color: #FFFFFF;
        width: 100%;
        float: left;
        font-family: "Source Sans Pro";
        color: #343434;
        padding-top: 15px;
        padding-bottom: 15px;
        text-align: center;
}

.left_article {
        background-color: #FFFFFF;
        width: 100%;
        float: left;
        font-family: "Source Sans Pro";
        color: #343434;
        padding-bottom: 15px;
        text-align: center;
}
.noDisplay {
        display: none;
}
.container .left_article h3 {
        padding-left: 5%;
        padding-right: 5%;
        margin-top: 5%;
        color: #DE1D20;
        font-weight: bold;
        text-transform: uppercase;
```

```
}

.container h3{
	color: #717070;
}
.container .left_article p {
	padding-left: 5%;
	padding-right: 5%;
	text-align: justify;
	line-height: 24px;
	margin-top: 30px;
	margin-bottom: 15px;
	color: #B3B3B3;
}
.right_article {
	width: 100%;
	float: left;
	background-color: #F6F6F6;
}
.container .right_article ul {
	margin-top: 0px;
	margin-right: 0px;
	margin-bottom: 0px;
	margin-left: 0px;
	padding-top: 0px;
	padding-right: 0px;
	padding-bottom: 0px;
	padding-left: 0px;
}
.right_article ul li {
	font-family: "Source Sans Pro";
	list-style: none;
	text-align: center;
	background-color: #B3B3B3;
	width: 90%;
	margin-left: auto;
```

```
        margin-right: auto;
        margin-top: 10px;
        margin-bottom: 10px;
        padding-top: 15px;
        padding-bottom: 15px;
        color: #FFFFFF;
        font-weight: bold;
        border-radius: 0px;
        transition: all 0.3s linear;
        border-left: 5px solid #717070;
}
.right_article ul li:hover {
        background-color: #717070;
        cursor: pointer;
}
.footer {
        background-color: #717070;
}
.title {
        font-weight: bold;
        font-style: normal;
        font-family: "Source Sans Pro";
        text-align: center;
        color: #FFFFFF;
        letter-spacing: 2px;
}
.placeholder {
        /* [disabled]max-width: 400px;
*/
        /* [disabled]max-height: 200px;
*/
        width: 100%;
        padding-top: 30px;
        /* [disabled]padding-left: 19px;
*/
        padding-bottom: 30px;
```

```
        height: 100%;
}
.left_half {
        background-color:#8AC860
}
.container .column_half.left_half h2 {
        color: #FFFFFF;
        font-family: "Source Sans Pro";
        text-align: center;
}
.right_half {
        background-color: #01B2D1;
        color: #FFFFFF;
        font-family: "Source Sans Pro";
        text-align: center;
        font-weight: bold;
}
.column_title {
        padding-top: 25px;
        padding-bottom: 25px;
}
.copyright {
        text-align: center;
        background-color: #717070;
        color: #FFFFFF;
        text-transform: uppercase;
        font-weight: lighter;
        letter-spacing: 2px;
        border-top-width: 2px;
        font-family: "Source Sans Pro";
}
body {
        margin-top: 0px;
        margin-right: 0px;
        margin-bottom: 0px;
        margin-left: 0px;
```

```
}

@media (max-width: 320px) {
.secondary_header ul li {
	float: none;
	margin-top: 28px;
	margin-left: 0px;
	width: 100%;
}
.container .secondary_header ul {
	margin-top: 0px;
	margin-right: 0px;
	margin-bottom: 0px;
	margin-left: 0px;
	padding-top: 0px;
	padding-right: 0px;
	padding-bottom: 0px;
	padding-left: 0px;
	height: auto;
	width: 100%;
	text-align: center;
}
.secondary_header {
	margin-top: 0px;
	margin-right: 0px;
	margin-bottom: 0px;
	margin-left: 0px;
	padding-top: 1px;
	padding-bottom: 40px;
}
.left_article {
	width: 100%;
	height: auto;
}
```

```
.left_article h3{

}
.right_article {
	width: 100%;
	height: auto;
}
.placeholder {
	width: 100%;
	margin-top: 22PX;
	margin-right: 0px;
	margin-bottom: 22PX;
	margin-left: 0px;
	padding-top: 0px;
	padding-right: 0px;
	padding-bottom: 0px;
	padding-left: 0px;
	max-width: 400px;
	max-height: 200px;
	height: auto;
}
.columns {
	width: 100%;
	margin-top: 0px;
	margin-right: 0px;
	margin-bottom: 0px;
	margin-left: 0px;
	padding-top: 0PX;
	padding-right: 0PX;
	padding-bottom: 0PX;
	padding-left: 0PX;
}
.columns p {
	padding-left: 10px;
	padding-right: 10px;
}
```

```
.column_half.left_half {
	width: 100%;
}
.column_half.right_half {
	width: 100%;
}
.copyright {
	padding-top: 25px;
	padding-bottom: 0px;
	margin-bottom: 0px;
}
.container .left_article h3 {
	margin-top: 30px;
}
.social .social_icon img {
	width: 80%;
}
.container .secondary_header {
}
}

@media (min-width: 321px) and (max-width: 768px) {
.secondary_header ul li {
	float: none;
	margin-top: 28px;
	margin-left: 0px;
	width: 100%;
}
.container .secondary_header ul {
	margin-top: 0px;
	margin-right: 0px;
	margin-bottom: 0px;
	margin-left: 0px;
	padding-top: 0px;
	padding-right: 0px;
	padding-bottom: 0px;
```

```
        padding-left: 0px;
        height: auto;
        width: 100%;
        text-align: center;
}
.secondary_header {
        margin-top: 0px;
        margin-right: 0px;
        margin-bottom: 0px;
        margin-left: 0px;
        padding-top: 1px;
        padding-bottom: 40px;
}
.left_article {
        width: 100%;
        height: auto;
}
.right_article {
        width: 100%;
        height: auto;
        padding-bottom: 25px;
}
.placeholder {
        margin-top: 0px;
        margin-right: 0px;
        margin-bottom: 0px;
        margin-left: 0px;
        padding-top: 0px;
        padding-right: 0px;
        padding-bottom: 0px;
        padding-left: 0px;
        width: 100%;
        max-width: 100%;
        height: auto;
        max-height: 100%;
}
```

```
.columns {
	width: 100%;
	margin-top: 6px;
	margin-right: 0px;
	margin-bottom: 6px;
	margin-left: 0px;
	padding-top: 0px;
	padding-right: 0px;
	padding-bottom: 0px;
	padding-left: 0px;
}
.columns p {
	padding-left: 14px;
	padding-right: 14px;
}
.column_half.left_half {
	width: 100%;
}
.column_half.right_half {
	width: 100%;
}
}

@media (min-width: 769px) and (max-width: 1000px) {
.secondary_header {
	overflow: auto;
	padding-top: 30px;
	padding-bottom: 30px;
}
.secondary_header ul li {
	margin-top: 10px;
	margin-right: 7%;
	margin-bottom: 10px;
	margin-left: 7%;
}
.left_article {
```

```
        height: auto;
}
.right_article {
        height: auto;
        padding-bottom: 27px;
}
.placeholder {
        width: 100%;
        margin-left: 0px;
        margin-right: 0px;
        padding-left: 0px;
        padding-right: 0px;
}
.columns {
        width: 50%;
        float: left;
        padding-left: 0px;
        padding-top: 0px;
        padding-right: 0px;
        padding-bottom: 0px;
}
.container .columns p {
        padding-left: 25px;
        padding-right: 25px;
}
}

@media (min-width: 1001px) {
}
.thumbnail {
        width: 100px;
        border-radius: 200px;
        height: 100px;
        margin-left: auto;
}
.thumbnail_align {
```

```
        font-size: 30px;
        color: #8AC860;
        text-align: center;
}

.thumbnail_align_bad {
        font-size: 30px;
        color: #DE1D20;
        text-align: center;
}

.social {
        text-align: center;
        margin-right: 0px;
        margin-bottom: 0px;
        margin-left: 0px;
        width: 100%;
        background-color: #414141;
        clear: both;
        overflow: auto;
}
.social_icon {
        width: 25%;
        text-align: center;
        float: left;
        transition: all 0.3s linear;
        line-height: 0px;
        padding-top: 7px;
}
.container .social .social_icon:hover {
        cursor: pointer;
        opacity: 0.5;
}
#menu {
}
```

```
/* The switch - the box around the slider */
.switch {
  position: relative;
  display: inline-block;
  width: 60px;
  height: 34px;
}

/* Hide default HTML checkbox */
.switch input {
  opacity: 0;
  width: 0;
  height: 0;
}

/* The slider */
.slider {
  position: absolute;
  cursor: pointer;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  background-color: #ccc;
  -webkit-transition: .4s;
  transition: .4s;
}

.slider:before {
  position: absolute;
  content: "";
  height: 26px;
  width: 26px;
  left: 4px;
  bottom: 4px;
  background-color: white;
```

```
  -webkit-transition: .4s;
  transition: .4s;
}

input:checked + .slider {
  background-color: #52bad5;
}

input:focus + .slider {
  box-shadow: 0 0 1px #2196F3;
}

input:checked + .slider:before {
  -webkit-transform: translateX(26px);
  -ms-transform: translateX(26px);
  transform: translateX(26px);
}

/* Rounded sliders */
.slider.round {
  border-radius: 34px;
}

.slider.round:before {
  border-radius: 50%;
}
```

multiColumnTemplate.css

```
@charset "UTF-8";
img {
  display: block;
  margin: auto;
  background-color: blue;
}
.reading_good{
```

```
        /*color: #8AC860;*/

        color:green;
}
.reading_bad{
        /*color: #DE1D20;*/
        color: red;
}
.reading{
        color: black;
/*      color: #A5A5A5;*/
}
.btn {
  background-color: #52bad5;
  border: none;
  color: white;
  padding: 12px 30px;
  cursor: pointer;
  font-family: "Source Sans Pro";
}

/* Darker background on mouse-over */
.btn:hover {
  background-color: #2EA0BE;
}

.container {
        background-color: #FFFFFF;
        width: 90%;
        margin-left: auto;
        margin-right: auto;
        border-bottom-width: 0px;
        padding-left: 0px;
        padding-top: 0px;
        padding-right: 0px;
        padding-bottom: 0px;
```

```
}
.row {
	width: 100%;
	margin-top: 0px;
	margin-right: 0px;
	margin-bottom: 0px;
	margin-left: 0px;
	padding-top: 0px;
	padding-right: 0px;
	padding-bottom: 0px;
	padding-left: 0px;
	display: inline-block
}
.row.blockDisplay {
	display: block;
}
.column_half {
	width: 100%;
	float: left;
	margin-top: 0px;
}
.columns {
	width: 25%;
	float: left;
	font-family: "Source Sans Pro";
	color: #A5A5A5;
	line-height: 24px;
	padding-top: 10px;
	padding-bottom: 10px;
	text-align: justify;
	margin-top: 15px;
	margin-bottom: 15px;
	padding-left: 0px;
	padding-right: 0px;
	margin-left: 0px;
	margin-right: 0px;
```

```
}

.columns1 {
	width: 20%;
	float: left;
	font-family: "Source Sans Pro";
	color: #A5A5A5;
	line-height: 24px;
	padding-top: 0px;
	padding-bottom: 0px;
	text-align: justify;
	margin-top: 0px;
	margin-bottom: 0px;
	padding-left: 0px;
	padding-right: 0px;
	margin-left: 0px;
	margin-right: 0px;
}

.columns2 {
	width: 33.3%;
	float: left;
	font-family: "Source Sans Pro";
	color: #A5A5A5;
	line-height: 24px;
	padding-top: 0px;
	padding-bottom: 0px;
	text-align: justify;
	margin-top: 0px;
	margin-bottom: 0px;
	padding-left: 0px;
	padding-right: 0px;
	margin-left: 0px;
	margin-right: 0px;
}
```

```
.columns3 {
	width: 100%;
	font-family: "Source Sans Pro";
	color: #717070;
	font-size: 17px;
	line-height: 34px;
	padding-top: 0px;
	padding-bottom: 0px;
	text-align: center;
	margin-top: 0px;
	margin-bottom: 0px;
	padding-left: 0px;
	padding-right: 0px;
	margin-left: 0px;
	margin-right: 0px;
}

.row .columns p {
	padding-left: 10%;
	padding-right: 10%;
}
.container .columns h4 {
	text-align: center;
	color: #01B2D1;
}

.container .columns1 h4 {
	text-align: center;
	color: #01B2D1;
}

.container .columns1_bad h4 {
	text-align: center;
	color: #01B2D1;
}
```

```
.container .columns1 {
	text-align: center;
	color: #8AC860;
}

.container .columns2 {
	text-align: center;
	color: #717070;
}

.container .columns1_bad{
	text-align: center;
	color: #DE1D20;
}

.timestamp {
	color: #717070;
}

.primary_header {
	width: 100%;

	backgroud-image:
url("https://cdn.pixabay.com/animation/2023/10/03/06/47/06-47-59-83_512.gif");
	background-color: #52bad5;

	padding-top: 10px;
	padding-bottom: 10px;
	clear: left;
	border-bottom: 2px solid #2C9AB7;
}
.secondary_header {
	width: 100%;
	padding-top: 50px;
	padding-bottom: 50px;
	background-color: #B3B3B3;
```

```
        clear: left;
}

.dashboard_overview{
        color: #DE1D20;
}

.dashboard_overview_good{
    color: green;
}

.dashboard_overview_bad{
    color: #DE1D20;
}

.container .secondary_header ul {
        margin-top: 0%;
        margin-right: auto;
        margin-bottom: 0px;
        margin-left: auto;
        padding-top: 0px;
        padding-right: 0px;
        padding-bottom: 15px;
        padding-left: 0px;
        width: 100%;
}
.secondary_header ul li {
        list-style: none;
        float: left;
        margin-right: auto;
        margin-top: 0px;
        font-family: "Source Sans Pro";
        font-weight: normal;
        color: #FFFFFF;
        letter-spacing: 1px;
        margin-left: auto;
```

```
        text-align: center;
        width: 33%;
        transition: all 0.3s linear;
}
.secondary_header ul li:hover {
        color: #717070;
        cursor: pointer;
}

.article {
        background-color: #FFFFFF;
        width: 100%;
        float: left;
        font-family: "Source Sans Pro";
        color: #343434;
        padding-bottom: 15px;
        text-align: center;
}

.article1 {
        background-color: #FFFFFF;
        width: 100%;
        float: left;
        font-family: "Source Sans Pro";
        color: #343434;
        padding-top: 15px;
        padding-bottom: 15px;
        text-align: center;
}

.left_article {
        background-color: #FFFFFF;
        width: 100%;
        float: left;
        font-family: "Source Sans Pro";
        color: #343434;
```

```
        padding-bottom: 15px;
        text-align: center;
}
.noDisplay {
        display: none;
}
.container .left_article h3 {
        padding-left: 5%;
        padding-right: 5%;
        margin-top: 5%;
        color: #DE1D20;
        font-weight: bold;
        text-transform: uppercase;
}

.container h3{
        color: #717070;
}
.container .left_article p {
        padding-left: 5%;
        padding-right: 5%;
        text-align: justify;
        line-height: 24px;
        margin-top: 30px;
        margin-bottom: 15px;
        color: #B3B3B3;
}
.right_article {
        width: 100%;
        float: left;
        background-color: #F6F6F6;
}
.container .right_article ul {
        margin-top: 0px;
        margin-right: 0px;
        margin-bottom: 0px;
```

```
        margin-left: 0px;
        padding-top: 0px;
        padding-right: 0px;
        padding-bottom: 0px;
        padding-left: 0px;
}
.right_article ul li {
        font-family: "Source Sans Pro";
        list-style: none;
        text-align: center;
        background-color: #B3B3B3;
        width: 90%;
        margin-left: auto;
        margin-right: auto;
        margin-top: 10px;
        margin-bottom: 10px;
        padding-top: 15px;
        padding-bottom: 15px;
        color: #FFFFFF;
        font-weight: bold;
        border-radius: 0px;
        transition: all 0.3s linear;
        border-left: 5px solid #717070;
}
.right_article ul li:hover {
        background-color: #717070;
        cursor: pointer;
}
.footer {
        background-color: #717070;
}
.title {
        font-weight: bold;
        font-style: normal;
        font-family: "Source Sans Pro";
        text-align: center;
```

```
        color: #FFFFFF;
        letter-spacing: 2px;
}
.placeholder {
        /* [disabled]max-width: 400px;
*/
        /* [disabled]max-height: 200px;
*/
        width: 100%;
        padding-top: 30px;
        /* [disabled]padding-left: 19px;
*/
        padding-bottom: 30px;
        height: 100%;
}

.container .column_half.left_half h2 {
        color: #FFFFFF;
        font-family: "Source Sans Pro";
        text-align: center;
}
.right_half {
        background-color: #01B2D1;
        color: #FFFFFF;
        font-family: "Source Sans Pro";
        text-align: center;
        font-weight: bold;
}
column_title {
        padding-top: 25px;
        padding-bottom: 25px;
}
.copyright {
        text-align: center;
        background-color: #717070;
        color: #FFFFFF;
```

```
        text-transform: uppercase;
        font-weight: lighter;
        letter-spacing: 2px;
        border-top-width: 2px;
        font-family: "Source Sans Pro";
}
body {
        margin-top: 0px;
        margin-right: 0px;
        margin-bottom: 0px;
        margin-left: 0px;
}

@media (max-width: 320px) {
.secondary_header ul li {
        float: none;
        margin-top: 28px;
        margin-left: 0px;
        width: 100%;
}
.container .secondary_header ul {
        margin-top: 0px;
        margin-right: 0px;
        margin-bottom: 0px;
        margin-left: 0px;
        padding-top: 0px;
        padding-right: 0px;
        padding-bottom: 0px;
        padding-left: 0px;
        height: auto;
        width: 100%;
        text-align: center;
}
.secondary_header {
        margin-top: 0px;
        margin-right: 0px;
```

```
        margin-bottom: 0px;
        margin-left: 0px;
        padding-top: 1px;
        padding-bottom: 40px;
}
.left_article {
        width: 100%;
        height: auto;
}


.left_article h3{

}
.right_article {
        width: 100%;
        height: auto;
}
.placeholder {
        width: 100%;
        margin-top: 22PX;
        margin-right: 0px;
        margin-bottom: 22PX;
        margin-left: 0px;
        padding-top: 0px;
        padding-right: 0px;
        padding-bottom: 0px;
        padding-left: 0px;
        max-width: 400px;
        max-height: 200px;
        height: auto;
}
.columns {
        width: 100%;
        margin-top: 0px;
        margin-right: 0px;
```

```
        margin-bottom: 0px;
        margin-left: 0px;
        padding-top: 0PX;
        padding-right: 0PX;
        padding-bottom: 0PX;
        padding-left: 0PX;
}
.columns p {
        padding-left: 10px;
        padding-right: 10px;
}
.column_half.left_half {
        width: 100%;
}
.column_half.right_half {
        width: 100%;
}
.copyright {
        padding-top: 25px;
        padding-bottom: 0px;
        margin-bottom: 0px;
}
.container .left_article h3 {
        margin-top: 30px;
}
.social .social_icon img {
        width: 80%;
}
.container .secondary_header {
}
}

@media (min-width: 321px) and (max-width: 768px) {
.secondary_header ul li {
        float: none;
        margin-top: 28px;
```

```
        margin-left: 0px;
        width: 100%;
}
.container .secondary_header ul {
        margin-top: 0px;
        margin-right: 0px;
        margin-bottom: 0px;
        margin-left: 0px;
        padding-top: 0px;
        padding-right: 0px;
        padding-bottom: 0px;
        padding-left: 0px;
        height: auto;
        width: 100%;
        text-align: center;
}
.secondary_header {
        margin-top: 0px;
        margin-right: 0px;
        margin-bottom: 0px;
        margin-left: 0px;
        padding-top: 1px;
        padding-bottom: 40px;
}
.left_article {
        width: 100%;
        height: auto;
}
.right_article {
        width: 100%;
        height: auto;
        padding-bottom: 25px;
}
.placeholder {
        margin-top: 0px;
        margin-right: 0px;
```

```
        margin-bottom: 0px;
        margin-left: 0px;
        padding-top: 0px;
        padding-right: 0px;
        padding-bottom: 0px;
        padding-left: 0px;
        width: 100%;
        max-width: 100%;
        height: auto;
        max-height: 100%;
}
.columns {
        width: 100%;
        margin-top: 6px;
        margin-right: 0px;
        margin-bottom: 6px;
        margin-left: 0px;
        padding-top: 0px;
        padding-right: 0px;
        padding-bottom: 0px;
        padding-left: 0px;
}
.columns p {
        padding-left: 14px;
        padding-right: 14px;
}
.column_half.left_half {
        width: 100%;
}
.column_half.right_half {
        width: 100%;
}
}

@media (min-width: 769px) and (max-width: 1000px) {
.secondary_header {
```

```
        overflow: auto;
        padding-top: 30px;
        padding-bottom: 30px;
}
.secondary_header ul li {
        margin-top: 10px;
        margin-right: 7%;
        margin-bottom: 10px;
        margin-left: 7%;
}
.left_article {
        height: auto;
}
.right_article {
        height: auto;
        padding-bottom: 27px;
}
.placeholder {
        width: 100%;
        margin-left: 0px;
        margin-right: 0px;
        padding-left: 0px;
        padding-right: 0px;
}
.columns {
        width: 50%;
        float: left;
        padding-left: 0px;
        padding-top: 0px;
        padding-right: 0px;
        padding-bottom: 0px;
}
.container .columns p {
        padding-left: 25px;
        padding-right: 25px;
}
```

```
@media (min-width: 1001px) {
}
.thumbnail {
        width: 100px;
        border-radius: 200px;
        height: 100px;
        margin-left: auto;
}
.thumbnail_align {
        font-size: 30px;
        color: #8AC860;
        text-align: center;
}

.thumbnail_align_good {
    font-size: 30px;
    color: #8AC860;
    text-align: center;

}

.thumbnail_align_bad {
        font-size: 30px;
        color: #DE1D20;
        text-align: center;
}

.social {
        text-align: center;
        margin-right: 0px;
        margin-bottom: 0px;
        margin-left: 0px;
        width: 100%;
        background-color: #414141;
```

```
        clear: both;
        overflow: auto;
}

.social_icon {
        width: 25%;
        text-align: center;
        float: left;
        transition: all 0.3s linear;
        line-height: 0px;
        padding-top: 7px;
}
.container .social .social_icon:hover {
        cursor: pointer;
        opacity: 0.5;
}
#menu {
}

/* The switch - the box around the slider */
.switch {
  position: relative;
  display: inline-block;
  width: 60px;
  height: 34px;
}

/* Hide default HTML checkbox */
.switch input {
  opacity: 0;
  width: 0;
  height: 0;
}

/* The slider */
.slider {
```

```
  position: absolute;
  cursor: pointer;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  background-color: #ccc;
  -webkit-transition: .4s;
  transition: .4s;
}

.slider:before {
  position: absolute;
  content: "";
  height: 26px;
  width: 26px;
  left: 4px;
  bottom: 4px;
  background-color: white;
  -webkit-transition: .4s;
  transition: .4s;
}

input:checked + .slider {
  background-color: #52bad5;
}

input:focus + .slider {
  box-shadow: 0 0 1px #2196F3;
}

input:checked + .slider:before {
  -webkit-transform: translateX(26px);
  -ms-transform: translateX(26px);
  transform: translateX(26px);
}
```

```
/* Rounded sliders */
.slider.round {
  border-radius: 34px;
}

.slider.round:before {
  border-radius: 50%;
}
```

References/Additional Reading

1) Fabio, A. (2015, June 6). Hacklet 50 – Hydroponic Projects. Hackaday. https://hackaday.com/2015/06/06/hacklet-50-hydroponic-projects/

2) Bremer, R. (2014, August 17). Aquaponic System Uses Arduino For Consistent Performance. Hackaday. https://hackaday.com/2014/08/16/aquaponic-system-uses-arduino-for-consistent-performance/

3) rjsears. (2022, April 11). rjsears/GardenPi. GitHub. https://github.com/rjsears/GardenPi

4) aHagouel. (2020). GitHub - aHagouel/IOT-Aquaponics-in-NYC. GitHub. https://github.com/aHagouel/IOT-Aquaponics-in-NYC

5) Agro-iot. (2020). GitHub - Agro-iot/iot2tangle.ayni. GitHub. https://github.com/Agro-iot/iot2tangle.ayni

6) Rob. " Aquaponics Design - 3 Easiest System Builds for the Backyard." *YouTube*, YouTube, www.youtube.com/watch?v=0QQA5BpWKec. Accessed 4 Dec. 2024.

7) Wikipedia Contributors. (2019, November 19). Aquaponics. Wikipedia; Wikimedia Foundation. https://en.wikipedia.org/wiki/Aquaponics