B.A.S.I.L. H.E.R.B.

Bio-Automated System with Intelligent Lighting Hydroponics for Enhanced Rapid Bloom

Group 5

Casey O'Donahoe - CE
Logan Voiselle - CE
Justin Boyd - EE
Justin Press - OSE

Sponsor: Michael Pape
Reviewers: Stephen Kuebler, Michael Pape, Enxia Zhang

Table of Contents

List of Figures

List of Tables

List of Equations

Chapter 1 - Executive Summary

BASIL HERB is designed to revolutionize the way indoor herbs are cultivated. Many consumers struggle to cultivate their own herbs due to time constraints and lack of knowledge of the optimal way to grow them. Our design aims to provide the necessary conditions to ensure that any consumer will be able to enjoy their own freshly grown herbs. This idea was cultivated and designed by four engineers from the University of Central Florida. Through the use of multiple sensors, photonic sciences, hydroponics, and software design, BASIL HERB will be a fully functional system designed to optimally produce herbs indoors.

Seeds of an herb will be able to be planted where the roots will grow into the hydroponics system. Once grown, the roots will be submerged in a water reservoir that contains nutrients being circulated by a pump which will distribute oxygen and disperse the nutrients evenly to all parts of the roots. Above where the herbs are growing will be a grow light that utilizes specific frequencies to optimally stimulate the plants in a simulated day/night cycle. Encased in the enclosure will be a microcontroller that is enabling a light sensor, pH sensor, color sensor, humidity and temperature sensor, and a water level sensor. The color sensor will be used to analyze the color of the leaves and report any deficiencies present. The rest of the sensors are responsible for tracking the different environmental factors present in the design. Through the use of a Wi-Fi enabled microcontroller, the data for all the sensors will be sent to a database that will display these values in a user-friendly design on our web application. The web app will be responsible for alerting the user once any one of the environmental factors is no longer optimal. Also, the user will be able to customize the timing of the daylight cycle.

This document is a record of project BASIL HERB’s complete design process. It begins with research of similar products to what we are designing and transitions into our goals/objectives for what this project will accomplish. Following this, is the research of a multitude of parts and sensors that were considered for the design. Then it dives into the specifics of the hardware and software design of this project. After that, we compare different LLMs and discuss our design constraints and regulations. Following that is a section on the systems fabrication, results from testing and integration, and the administrative content. Finally, we have the conclusion with the Appendix and References.

Chapter 2 - Project Description

The Bio-Automated System with Intelligent Lighting Hydroponics for Enhanced Rapid Bloom (BASIL H.E.R.B.) project is born out of a compelling motivation to simplify indoor herb cultivation, particularly for individuals with busy lifestyles. This chapter provides a comprehensive overview of the project's motivation, background, and environmental requirements for optimal basil growth. By exploring existing products and prior related work, we aim to identify key features and shortcomings that inform our design decisions.

The initial sections detail the ideal environmental conditions for growing basil, supported by a summary table for quick reference. We then delve into a comparative analysis of existing hydroponic systems, including AeroGarden, AquaSprouts Garden Bundle, Botanium, and Click and Grow Smart Garden 9. This evaluation highlights the strengths and limitations of these products, guiding our approach to developing a superior system.

Further, the chapter outlines the project’s goals, objectives, and key features, providing a clear roadmap for the design and development of our system. Detailed project specifications, accompanied by hardware and software block diagrams, offer insight into the technical foundations of our solution. By the end of this chapter, readers will have a solid understanding of the project's purpose, scope, and the innovative strategies we employ to achieve our vision.

2.1 Motivation

As college students, we often find ourselves juggling a hectic schedule filled with classes, assignments, projects, and extracurricular activities. Amidst this busy lifestyle, many of us still desire to cultivate our own organic food, recognizing the benefits it offers in terms of health, freshness, and environmental sustainability. However, the time and effort required to maintain a traditional garden can be overwhelming, leading to neglected plants and suboptimal growth. In fact, studies have shown that up to 50% of home-grown basil plants die due to inadequate care and attention. This challenge served as the primary motivation behind our project: to develop an intelligent control system that simplifies the process of growing organic basil, even for those with demanding schedules.

Our project's motivation was further solidified when our photonics team member was approached by Dr. Michael Pape, an entrepreneurship professor with a Ph.D. in chemistry who had previously sold a company to Pfizer. Dr. Pape introduced a new technique for optimizing plant growth using electromagnetic field generators, which aligned with the photonics member's existing ideas for a similar setup. This serendipitous encounter not only validated our project's potential but also secured funding and support from Dr. Pape, who will serve as a sponsor and reviewer throughout the development process.

2.2 Background

The demand for fresh, locally grown organic produce has been steadily increasing, with organic food sales in the U.S. reaching $50.1 billion in 2019 [1]. However, traditional gardening methods can be challenging, especially for college students with limited time and resources. In fact, a survey found that 31% of college students struggle to maintain a healthy diet due to time constraints [2].

Advances in photonics, electrical engineering, and computer science have opened up new possibilities for addressing these challenges. Smart greenhouses, equipped with intelligent control systems, have emerged as a promising solution, with the global smart greenhouse market expected to reach $2.1 billion by 2025 [3]. By applying our expertise in these fields, we aim to develop an intelligent greenhouse control system tailored specifically for herb cultivation, empowering college students and other busy individuals to successfully grow their own organic herbs. Through this project, we hope to contribute to the broader movement towards sustainable, locally grown food production.

2.2.1 Ideal Basil Environmental Conditions

Sweet basil (Ocimum basilicum), a member of the mint family (Lamiaceae), is native to southern Asia and the South Pacific islands. This warm-weather herb flourishes in locations that receive 6 to 8 hours of direct sunlight daily, though it can also be grown indoors under artificial lights set for 10 to 12 hours a day. Basil thrives in temperatures between 70°F and 85°F (21°C to 29°C) and is highly sensitive to cold and frost; even brief exposure to such conditions can be fatal.

For optimal growth, basil requires consistently moist, well-drained soil. Overhead watering should be avoided to prevent fungal growth and diseases. To ensure good air circulation and prevent overcrowding, space basil plants about 12 to 18 inches apart. The ideal soil pH ranges from 6.0 to 7.5. When growing basil indoors, use a liquid fertilizer at half the recommended strength every 4 to 6 weeks. For outdoor basil, fertilize every 3 to 4 weeks to support healthy growth.

Regular harvesting and pruning are crucial for promoting continued growth. To harvest, cut just above a pair of leaves; this stimulates new growth within a week. Keep in mind that basil is an annual plant, meaning it will eventually flower and die regardless of care. However, regular pruning can delay flowering and extend the plant's lifespan, ensuring a longer harvest period.

Table 2.1. Summary of Ideal Basil Environmental Conditions

Feature	**Description**
Botanical Name	Ocimum basilicum
Family	Lamiaceae
Origin	Southern Asia and South Pacific Islands
Light Requirements (Outdoor)	6-8 hours of direct sunlight daily
Light Requirements (Indoor)	10-12 hours of direct sunlight daily

Temperature Range	21°C - 29°C
Sensitivity	Highly sensitive to cold and frost
Substrate	Consistently moist and well-drained soil, or aerated nutrient rich water
Watering	Avoid overwatering and overhead watering
Spacing	31 – 46 cm apart
pH	6.0 – 7.5
Fertilization (Indoor)	Liquid fertilizer at half label dose every 4-6 weeks
Fertilization (Outdoor)	Liquid fertilizer at half label dose every 3-4 weeks
Harvesting	Cut just above pair of leaves
Lifespan	1 year (Annual plant)

2.3 Existing Product / Prior Related Work

2.3.1 AeroGarden: Harvest Elite

The AeroGarden Harvest elite is a popular choice for automated indoor plant care. This product claims that it will grow six plants five times faster without the need for soil. AeroGarden Harvest Elite includes the following features:

- 20W full spectrum LED grow light
- Up to 12” plant height
- Stainless steel enclosure 17.4” H x 10.5” W x 6.25” D
- Digital display and push button controls
- Automatic reminders for water and plant food
- Vacation mode

AeroGarden is designed so that plant pods can be inserted into six different spots. The pods have holes in the bottom where the roots of the plants are able to grow into the water reservoir beneath to absorb the nutrient enriched water. Over-arching the plant's pods is an LED panel which emits lights that are specific frequencies to stimulate photosynthesis. They have also incorporated a control panel with a user interface which allows the user to set reminders and control the LEDs.

Figure 2.1. AeroGarden

2.3.2 <u>AquaSprouts Garden Bundle</u>

AquaSprouts is a product kit that simplifies aquaponics in the home. Aquaponics is the combination of aquaculture (raising fish) and hydroponics (growing plants in water) into a symbiotic relationship where the fish waste provides nutrients to the plants while the plants filter and clean the water in a closed loop cycle. The AquaSprouts Garden Bundle includes the following features:

- Grow-bed: housing that fits any standard 10 gallon aquarium
- Light Bar: 2ft, 2640 lumen LED strip providing daylight white light
- Submersible LED light: Full range RGB LEDs for aesthetic purposes
- Pump and Timer: Submersible 160 gallon per hour pump and mechanical plug timer with 15 minute adjustable increments
- Grow Media: Allows for plant roots and fish waste to have a surface to combine

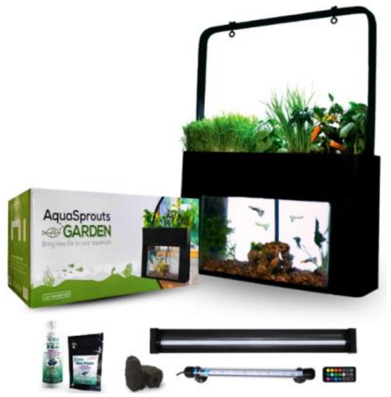


Figure 2.2. AquaSprouts

2.3.3 <u>Botanium</u>

Is a stylish but simple single plant hydroponics device that provides automated watering and nutrient delivery into a porous growing medium. Botanium is divided into a plant chamber with the porous growing medium and a nutrient rich water reservoir with a submerged pump.

Figure 2.3. Botanium

2.3.4 Click and Grow Smart Garden 9

The Click & Grow Smart Garden 9, launched in 2020, is an indoor gardening system designed to simplify the process of growing fresh herbs, vegetables, and flowers in urban living spaces. Utilizing advanced hydroponic technology, energy-efficient LED lighting, and a self-watering system, the Smart Garden 9 ensures optimal plant growth with minimal user intervention. The companion mobile app offers personalized gardening tips, reminders, and inspiration, empowering users to cultivate fresh, healthy produce year-round and promote a sustainable lifestyle.

Figure 2.4. Click and Grow Smart Garden 9

2.3.5 Evaluation of Products

After taking a look into these commercially available products we got some reassurance and new ideas. The products already in circulation were using some ideas we had already thought of implementing so this was a good sign that we were on the right track of thinking to begin with. One major takeaway we realized when researching these products was that none of them were using soil which we were going to use originally. This led us to start doing some research into hydroponics and how we could benefit from also implementing a similar system into our product.

2.3.5.1 LEDs

One of the primary concepts we envisioned at the outset of our B.A.S.I.L. H.E.R.B. project was the utilization of LEDs as the main light source, rather than depending on natural sunlight. This idea stemmed from the need to provide consistent and controllable lighting conditions, crucial for optimal plant growth. Our initial research further validated this approach, as we observed that most advanced automated growing systems heavily rely on LED technology in their designs. For instance, popular products like AeroGarden and Click and Grow Smart Garden incorporate sophisticated LED light hoods that emit specific wavelengths tailored to stimulate photosynthesis efficiently.

The use of LEDs in these systems offers several advantages, including the ability to fine-tune light spectra to match the plants' needs at various growth stages, from germination to flowering. These LEDs are designed to provide the blue and red light frequencies that are most effective for photosynthesis, ensuring healthy and robust plant development. By integrating a similar LED setup, our project aims to replicate these benefits, ensuring our plants receive the precise light spectrum necessary for optimal growth.

In addition to spectrum control, our system will feature a timer mechanism to simulate natural daylight cycles. This approach mimics the natural environment, promoting healthier growth patterns and improving overall plant vitality. The timer will allow us to adjust light durations to align with the plants' circadian rhythms, ensuring they receive adequate periods of light and darkness. This aspect of our design is inspired by the successful implementation in AeroGarden and Click and Grow systems, which have demonstrated the effectiveness of using timed LED lighting to simulate natural day-night cycles.

2.3.5.2 Water Reservoir

We originally wanted to implement a self-watering system but due to the benefits of integrating hydroponics we decided to change our approach. BASIL will utilize a water reservoir which will be placed under the herbs where the roots will be submerged in nutrient dense water. This idea came from AeroGarden, AquaSprouts, and Botanium as they all use a form of hydroponics. Similar to these products we will be implementing a pump to incorporate oxygen and disperse the nutrients in the water so that it reaches the roots. We plan on improving on these designs and implementing our own form of hydroponics that will cater to herbs specifically.

2.3.5.3 Sensors

Our system will be equipped with a comprehensive suite of sensors, ensuring it stands on par with, and even surpasses, related products currently available in the market. Drawing inspiration from successful models like AeroGarden, Botanium, and Click and Grow, our design will incorporate several essential sensors to monitor and maintain optimal growing conditions for the plants. Like these systems, our project will feature a water sensor, critical for ensuring that the water reservoir is consistently filled to the appropriate capacity, preventing both under-

watering and over-watering scenarios. The water sensor will provide real-time feedback to the control system, triggering automatic refills when water levels drop below a specified threshold, thereby maintaining a stable growing environment.
In addition to the water sensor, our system will integrate a light sensor similar to those used in AeroGarden and Click and Grow. This sensor will continuously monitor the light intensity and duration that the plants receive. If the light levels fall below the optimal range, the system will automatically adjust the LEDs to provide the necessary illumination. This feature ensures that the plants receive adequate light for photosynthesis, promoting healthy growth and development. What sets our project apart from existing products is the inclusion of a broader array of sensors designed to monitor multiple environmental parameters, including a temperature sensor to track the ambient temperature around the plants and a pH sensor to monitor the acidity or alkalinity of the nutrient solution.
Moreover, our project will be equipped with a humidity sensor to measure the moisture levels in the air, ensuring that the relative humidity stays within the optimal range for basil. A color sensor will detect changes in leaf coloration, which can indicate nutrient deficiencies or plant stress, allowing for timely interventions. Additionally, a Gauss sensor will monitor electromagnetic field (EMF) levels, ensuring that the EMF generator operates safely and effectively, contributing to enhanced plant growth. All sensor data will be processed and displayed on our dedicated website, offering users real-time access to comprehensive data regarding their plants' growing conditions. Users will be able to monitor water levels, light intensity, temperature, pH, humidity, leaf color, and EMF levels from any location, with the website providing alerts and recommendations based on the sensor readings.

2.3.5.4 Electromagnetic Field (EMF) Generator

One of the most significant differentiators for our project, setting it apart from other products currently available on the market, is the planned integration of an Electromagnetic Field (EMF) Generator. While our primary objective is to ensure robust growth and yield of indoor plants, we have identified a stretch goal that aims to dramatically enhance the effectiveness of our system. This stretch goal involves leveraging specific EMF frequencies to optimize plant growth conditions in ways that have not been previously explored in commercial products.
The inclusion of an EMF generator is based on cutting-edge research that demonstrates how particular thresholds of electromagnetic frequencies can significantly increase biomass accumulation in plants. By incorporating these frequencies, our system aims to stimulate more vigorous root growth, which is especially advantageous for herb cultivation. A stronger root system not only enhances the plant's ability to uptake nutrients and water more efficiently but also improves overall plant health and resistance to environmental stressors.
Currently, no other products on the market employ an EMF generator to boost plant growth, making this a pioneering feature of our project. By pushing the boundaries of traditional indoor gardening systems, we aspire to provide our consumers with a consistently abundant supply of fresh herbs. The EMF generator

represents a significant innovation that could revolutionize indoor plant cultivation, offering our users unprecedented growth rates and yields. This ambitious stretch goal underscores our commitment to leveraging advanced technology to enhance the indoor gardening experience.

2.3.5.5 External Connectivity

In order to include a component of software engineering into our design we knew that implementing a smart system capable of extrapolating data from our sensors and sending notifications was the best solution. AeroGarden and Click and Grow Smart Garden both utilize a pairable phone application designed to give the user control over the LEDs, send notifications, and provide specific plant information. In our design we plan on utilizing a website to handle these tasks and include some control over the EMF generator.

2.3.5.6 Takeaways

Our project represents the culmination of all previous works in the field of automated hydroponic systems. By carefully analyzing and integrating the best features from existing models such as AeroGarden, Botanium, and Click and Grow, we aim to enhance and optimize these elements to work more efficiently within our design. This approach ensures that our system not only incorporates proven technologies but also improves upon them to deliver superior performance and user experience. By taking the best parts of these existing systems and refining them, we create a robust platform that leverages the strengths of prior innovations while addressing their limitations.

One of the key differentiators in our design is the implementation of an Electromagnetic Field (EMF) Generator, designed to stimulate plant growth and enhance biomass accumulation—a feature not commonly found in most current systems. Additionally, our design emphasizes the use of advanced external connectivity options, enabling users to control and monitor their herb cultivator remotely. This connectivity ensures that users can manage their plants' environment from anywhere, providing greater flexibility and convenience. Together, these components form a semi-autonomous herb cultivation system with significantly enhanced growing capabilities, offering optimal growth conditions with minimal manual intervention and full user oversight. This comprehensive approach results in a robust, user-friendly solution that stands at the forefront of modern hydroponic technology.

2.4 Project Outline

2.4.1 Goals

Basic Goals:

- Ensure optimal growing conditions for the basil plant by accurately monitoring environmental factors.
- Implement a hydroponics-based system that efficiently circulates oxygen and nutrients to the roots of the basil plant.
- Implement a web application for data analysis and user-friendly interaction
- Design a portable and safe hardware setup that can be comfortably placed inside a home

Advanced Goals:

- Stimulate plant growth by integrating optimal LED lighting systems.

Stretch Goals:

- Implement machine learning algorithms to detect plant health issues prior to leaves turning brown.
- Develop a mobile application for remote monitoring and control of the B.A.S.I.L. system.
- Incorporate an EMF generator to disperse the most optimal waves for further growth stimulation.

2.4.2 Objectives

Basic Objectives:

- Integrate sensors into the hardware setup with the following accuracy ranges:
  - Temperature sensor accurate within ±2°C
  - Humidity sensor accurate within ±5% RH
  - Light intensity sensor accurate within ±5% of the full-scale range
- Design and integrate a hydroponics system into the B.A.S.I.L. setup that includes:
  - A water reservoir with a capacity of at least ½ gallon to ensure an adequate supply of nutrient solution
  - A submersible water pump capable of circulating the nutrient solution at a rate of 1-2 liters per minute to maintain optimal oxygen levels and nutrient distribution
  - A network of pipes and tubing to direct the nutrient solution from the reservoir to the plant's roots and back, ensuring continuous circulation
- Develop a user-friendly web application that includes:
  - An intuitive user interface for displaying real-time sensor data and plant growth metrics
  - A backend system that efficiently processes and stores sensor data

    - Deployed on an AWS cloud instance
- Design and construct a hardware enclosure that meets the following criteria:
    - Dimensions not exceeding 24 inches in length, width, or height to ensure portability
    - Made from durable, food-safe, and water-resistant materials to ensure longevity and safety

Advanced Objectives:

- Integrate an adjustable LED lighting system into the hardware setup that includes:
    - LED lights with a spectrum optimized for basil growth, focusing on wavelengths between 450-480 nm (blue)
    - A control system that allows users to adjust light intensity and duration based on the plant's growth stage

Stretch Objectives:

- Develop a machine learning-based plant health detection system that includes:
    - Integration of a high-resolution camera into the B.A.S.I.L. system to capture images of the basil plant at regular intervals
    - Creation of a comprehensive dataset of basil plant images, including healthy plants and those exhibiting early signs of stress, such as leaf curling or discoloration
    - Development and training of a machine learning model using the collected dataset to accurately identify early symptoms of plant stress or disease
- Develop a feature-rich mobile application for the B.A.S.I.L. system that includes:
    - A user-friendly interface that displays real-time sensor data, plant growth metrics, and system status
    - Secure user authentication and authorization to ensure data privacy and system security
    - Remote control capabilities, allowing users to adjust lighting, watering, and EMF settings from their mobile devices
- Develop and integrate an EMF generator into the B.A.S.I.L. system that includes:
    - An EMF generator capable of producing frequencies between 0.1 Hz and 30 Hz, which are believed to have potential effects on plant growth

2.4.3 Features

- Sensors that obtain accurate readings of the light intensity, pH, temperature, and humidity
- A web application that shows data analysis on the measurements of the plant and have the information available in a user friendly environment
- An automated water circulation system

- An EMF generator to disperse the most optimal waves to further stimulate growth
- Portable design which can be placed comfortably inside a home
- LED lighting system which stimulates the plants to further increase growth
- Safe and user-friendly hardware design

2.4.4 Product Description

BASIL will be designed to make indoor plant care simpler and more efficient. With the integration of multiple sensors we will be able to accurately track a few environmental factors such as the pH levels, temperature, and humidity. The goal is to keep the herbs at their most optimal level for these readings to ensure the plant is always primed for growth. There will also be a nutrient rich water reservoir which will eliminate the need for soil and having to water the plant manually. Our design will additionally include LED grow lights which will be emitting the optimal spectrum to stimulate the plants growth further. All this data will be available on a web application so the user is able to track their plants health and receive real-time updates. BASIL will overall reduce the amount and time and effort normally required when it comes to growing herbs so even the busiest individuals will be able to enjoy fresh, organic herbs year-round.

2.5 Project Specifications

Table 2.2. Project Requirements and Specifications

System Dimensions	
Dimensions	< L: 600 mm x W: 600 mm x D: 600 mm
Weight	< 13 kg
Hydroponics System	
Water Pump	≥ 1 Liter per minute
Reservoir	≥ 4 Liters
Water Level Sensor	± 25 mm
Environmental Monitoring	
Temperature Sensor	±2 °C

Humidity Sensor	±5% RH
Light Sensor	0 to 100,000 lux
pH Sensor	±0.5 pH
Color Sensor	±5% Full-Scale Range
Growth Stimulation	
Artificial Sunlight	2,000 to 3,000 lumens
Adjustable EMF Emitter	0.1 - 30 Hz
Web Application Deployment	
Cloud Deployment	AWS Lightsail
Database Deployment	MySQL instance using AWS Lightsail
AI / ML Data Analysis	
Speed of response from API	< 5000 ms
Data Export	
PCB to Database	< 5000 ms
Web Application to PCB	< 5000 ms

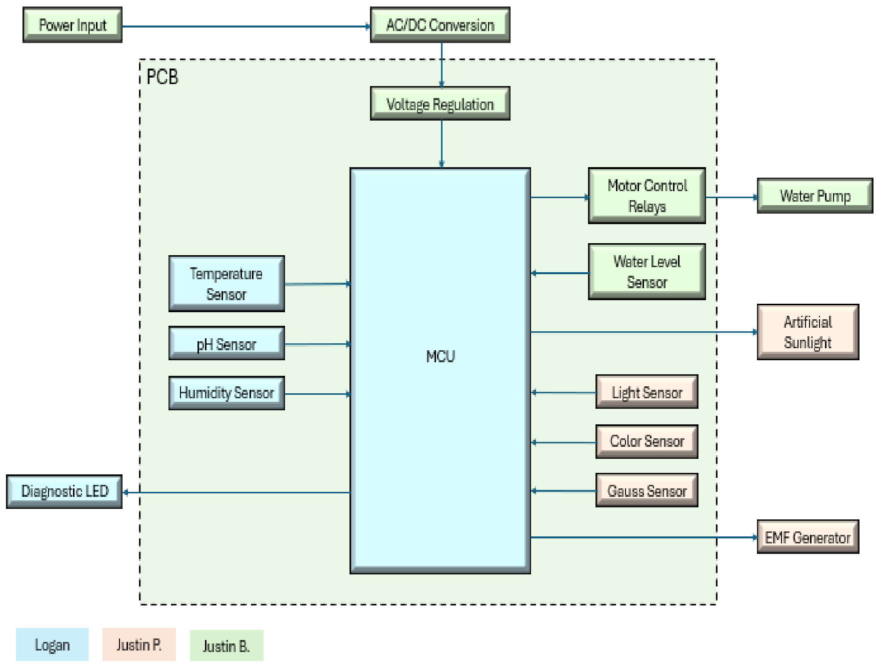


Figure 2.5. Hardware Block Diagram

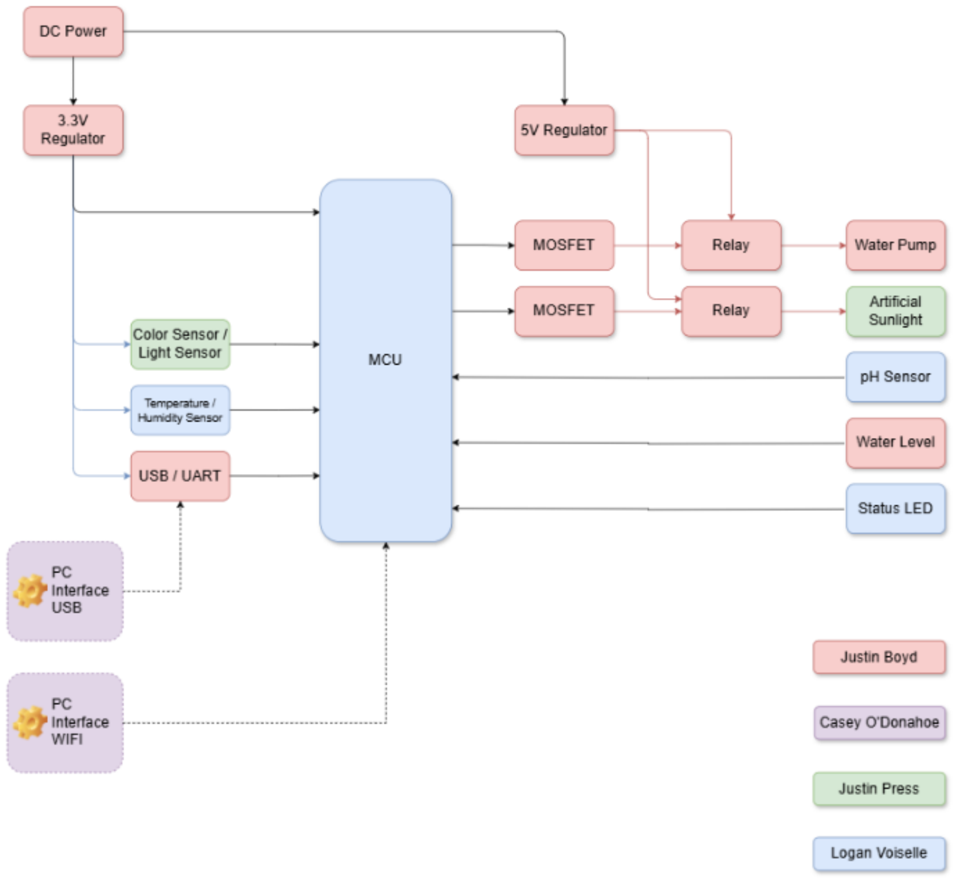


Figure 2.6. SD2 Updated Hardware Block Diagram

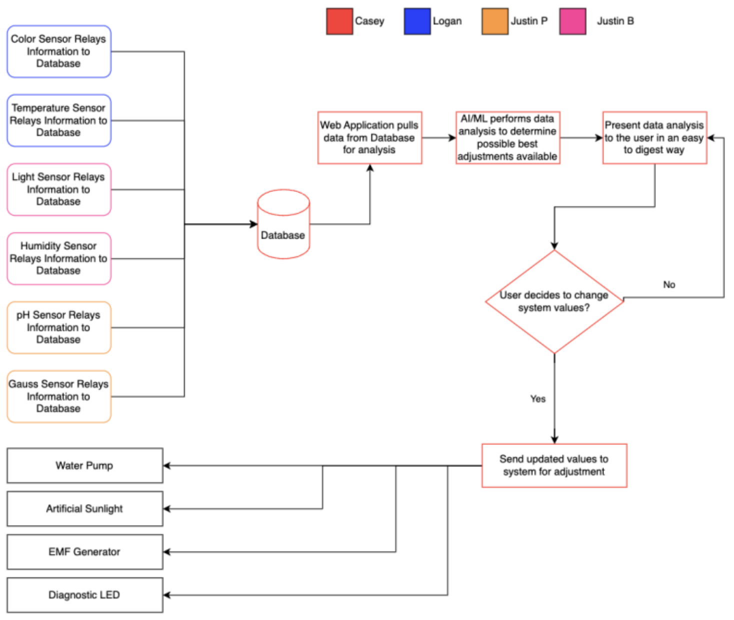

Casey
Logan
Justin P
Justin B
Color Sensor Relays Information to Database
Temperature Sensor Relays Information to Database
Light Sensor Relays Information to Database
Humidity Sensor Relays Information to Database
pH Sensor Relays Information to Database
Gauss Sensor Relays Information to Database
Database
Web Application pulls data from Database for analysis
AI/ML performs data analysis to determine possible best adjustments available
Present data analysis to the user in an easy to digest way
User decides to change system values?
No
Yes
Send updated values to system for adjustment
Water Pump
Artificial Sunlight
EMF Generator
Diagnostic LED

Figure 2.7. Software Block Diagram

The software block diagram above implements an approach that updates our database with new values periodically. This data is then leveraged in our web application and is then analyzed by AI to then recommend to the user what to do in their system. Then it is up to the user to actually update those values in the web application, if they decide to do so, those values are then sent back to the system to be updated. We wanted to create the web app to be a one stop shop for all of the users' needs for their system, this goes back on one of our main objectives of being convenient for users.

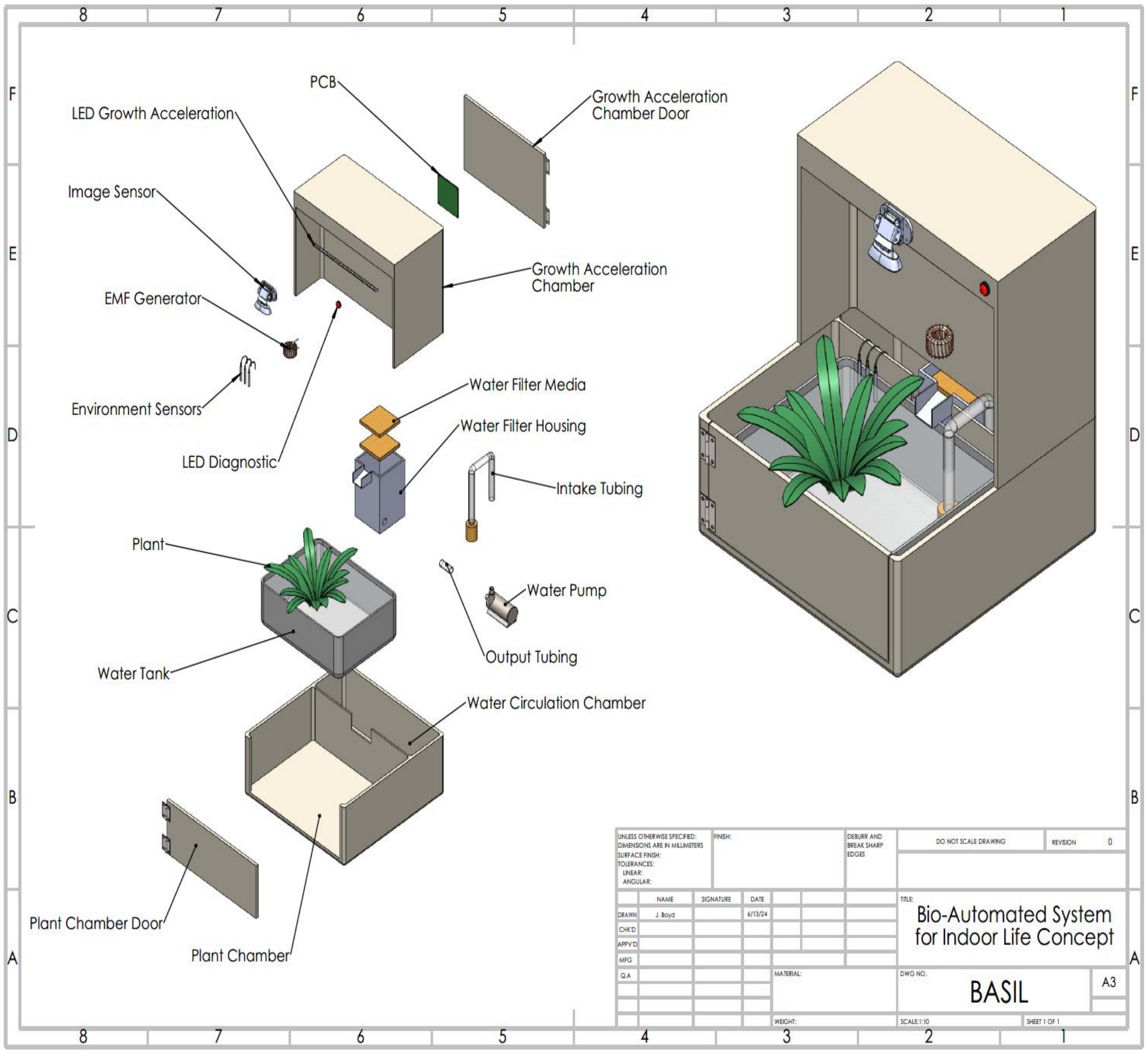


Figure 2.8. Prototype

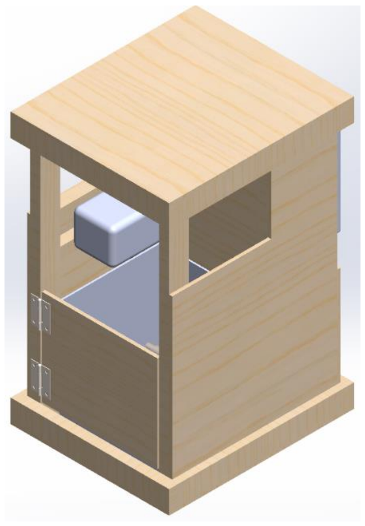
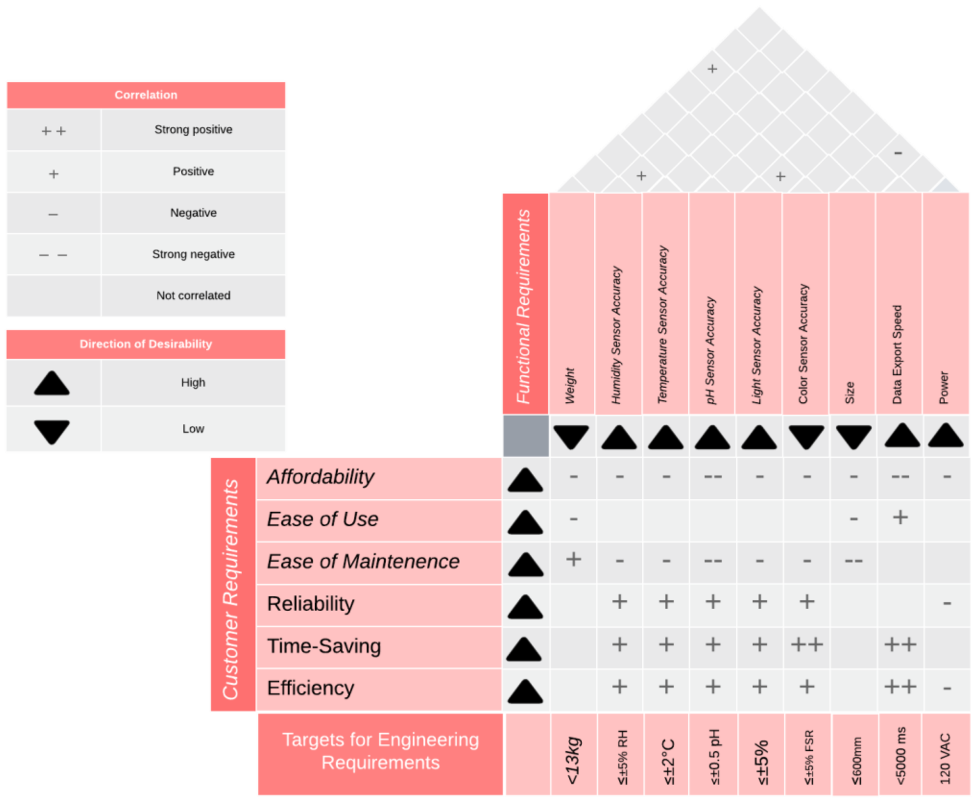


Figure 2.9. SD2 Updated Prototype



Correlation	
++	Strong positive
+	Positive
–	Negative
– –	Strong negative
	Not correlated

Direction of Desirability	
▲	High
▼	Low

Customer Requirements		Weight	Humidity Sensor Accuracy	Temperature Sensor Accuracy	pH Sensor Accuracy	Light Sensor Accuracy	Color Sensor Accuracy	Size	Data Export Speed	Power
		▼	▲	▲	▲	▲	▼	▼	▲	▲
Affordability	▲	-	-	-	--	-	-	-	--	-
Ease of Use	▲	-						-	+	
Ease of Maintenence	▲	+	-	-	--	-	-	--		
Reliability	▲		+	+	+	+	+			-
Time-Saving	▲		+	+	+	+	++		++	
Efficiency	▲		+	+	+	+	+		++	-
Targets for Engineering Requirements		<13kg	≤±5% RH	≤±2°C	≤±0.5 pH	≤±5%	≤±5% FSR	≤600mm	<5000 ms	120 VAC

Figure 2.10. House of Quality

Chapter 3 - Research and Investigation

Going into this project our team as a whole had somewhat general knowledge when it came to the types of parts we were going to be testing and implementing through our background in our variety of courses. For our specific project though we needed to figure out what our best options would be when considering our project scope. We began by identifying what our needs for the project were and then started investigating in order to figure out what kind of parts were best suited to handle our project specifications.

Because of the nature of our project, we split up the research based on what our roles are for integrating the parts in SD2. This enabled each of us to become more familiar with the parts we will be handling and to fully understand the intricacies of what we will be implementing.

3.1 Microcontrollers

Selecting an appropriate microcontroller was the first crucial step in ensuring the design of a system that provides seamless operation and integration of various components. This microcontroller is essentially the brain of the operation, responsible for coordinating the functions of the sensors, LED system, and pump to ensure maximum efficiency. Its role is pivotal, as it must manage the data inputs from multiple sensors, process this information, and control the actuators to maintain the ideal conditions for plant growth. Given the complexity of monitoring and adjusting numerous environmental factors such as light, temperature, humidity, and pH levels, the microcontroller selected needs to be capable of handling multiple tasks simultaneously while maintaining high reliability and performance.

Since we planned on monitoring multiple environmental factors, the microcontroller needs to have sufficient processing power and memory to handle the concurrent operations without lag or error. This capability ensures that sensor data is processed in real-time, allowing for immediate adjustments to environmental controls. Additionally, because the system is designed to operate continuously to ensure the plant's health, the microcontroller must be able to provide real-time updates while remaining energy efficient. Energy efficiency is critical for long-term operation, minimizing power consumption to prolong the life of the system and reduce the need for frequent maintenance. Moreover, our project aims to extrapolate data to a database for further analysis and user access, which necessitates robust data processing and communication capabilities. Therefore, the microcontroller must support wireless communication protocols, such as Wi-Fi or Bluetooth, to transmit data effectively and integrate smoothly with our online monitoring platform. By meeting these requirements, the selected microcontroller will ensure that our hydroponic system operates reliably, efficiently, and sustainably, providing users with detailed insights and control over their plant cultivation environment.

3.1.1 Raspberry Pi 4

The Raspberry series has many advantages that can meet our requirements inside our project scope. The Raspberry Pi is a fully functional Linux-based computer system that is part of an embedded system that can be programmed for specific functions and a standard computer, which makes it suitable for software development and testing. Equipped with a 1.8 GHz quad-core CPU, it has enough power to perform different tasks, which is an advantage for our project because we want to track several factors of the environment. Adding 1 GB of RAM and 32 GB of storage removes any memory-related issues, a big turn-off to other microcontrollers.

The device also includes built-in 2.4 and 5 GHz 802.11ac wireless, Bluetooth, and Bluetooth Low Energy to provide optimum connectivity. It also supports multiple peripherals through USB 3.0 ports and a 40-pin GPIO header that accommodates UART, SPI, and I2C protocols. However, the Raspberry Pi has a 5 V power supply, with a minimum of 3 A, which is relatively high compared to the microcontroller boards. It also lacks built-in ADC and DAC, which might be necessary for specific sensor applications.

3.1.2 Raspberry Pi Zero

The Raspberry Pi Zero is slightly different from the usual Raspberry Pi and comes with a more simplistic design and fewer features but it is much more affordable. This smaller version retains all the significant advantages of the Raspberry Pi, including software development, since the device works under the Linux PC OS. It is equipped with a 1 GHz single-core CPU and 512 MB of RAM, which is low compared to its other version but sufficient for our project.

It also has a built-in 2.4 GHz band wireless LAN and Bluetooth, facilitating connectivity with an onboard antenna. It lacks more USB ports and options for connectivity, and such as a basic model, it lacks the integration of an ADC and DAC. This device also requires less power, unlike the earlier one which needs 3A, the smaller version only requires a single A which helps us stay more energy efficient.

Table 3.1. Part 1 Microcontroller Comparisons

	Raspberry Pi 4	Raspberry Pi Zero	Arduino MKR 1000	Arduino MKR WAN 1300
Processor	1.8GHz quad-core	1GHz single-core	1GHz single-core	48 MHz
RAM	1GB-8GB	512 MB	32 KB	32 KB
Wi-Fi	Yes	No	Yes	No
Bluetooth	Yes	No	Yes	No
GPIO Pins	40	40	8	8
ADC	No	No	Yes	Yes
DAC	No	No	Yes	Yes
Power Consumption	3.5-5W	0.4-0.6W	0.231-1.056W	0.231-1.056W

3.1.3 Arduino MKR 1000

The Arduino MKR 1000 series balances low power consumption while also carrying wi-fi capabilities. The intended operating voltage is 5V with a minimal current draw, which makes it suitable for low-voltage devices. The board backs up rechargeable batteries with automatic charging, which could be ideal for our project. Connectivity features include 2.4 GHz 802.11 bgn Wi-Fi, Bluetooth, and Bluetooth Low Energy, enabling solid wireless communication.

The interfaces include UART, SPI, I2C, PWM, ADC, and DAC, which can connect the system's different sensors and actuators within the MKR 1000. It has 256 KB of flash memory, 32 KB of RAM, and a 48 MHz processor, which should suffice for this project. The Arduino environment is rather popular due to its intuitive tools and many resources available, so the development process would be easier in comparison to different models. However, this module costs approximately $40, which is high compared to other microcontroller options. Furthermore, its relatively low operating temperature and low processing power features could be a problem when executing our project.

3.1.4 Arduino MKR WAN 1300

For the Arduino MKR WAN 1300, networking protocols are implemented as LoRa and LoRaWAN and it is suited for power-conscious remote sensors because of low power consumption and long distances. The board offers ultra-low-power modes that can function on battery power with as small as 104 µA. This makes it ideal for long-term use in remote areas, or in our case where we want the power consumption to remain low.

The MKR WAN 1300 also has several interfaces, similar to the MKR 1000 board, which ranges from UART, SPI, I2C, PWM, ADC, and DAC. It boasts extensive security features, including encryption algorithms and Wi-Fi Protected Access (WPA3) support. It makes the wireless protocol secure and safe from unauthorized access by any person or device. However, compared to other microcontrollers, they are one of the most expensive on the market at 46$. Although its price is higher than others, its overall low power capabilities and long range communication may benefit our project.

3.1.5 ESP32 series

3.1.5.1 ESP32-WROOM-32

ESP32-WROOM-32 is one of the most popular models of development modules around ESP32 and is characterized by high performance and connectivity capabilities. It also encompasses dual-mode Bluetooth and 2.4 GHz Wi-Fi, making it convenient for forming mesh networks. Aboard the MH-53E is a dual-core processor of the brand ARM926TE, which operates at a maximum speed of 240

MHz but can be reduced for power conservation to 80 MHz It includes 4 MB flash memory and 520 KB SRAM to deliver adequate data storage and processing ability for various operations. It has 26 multipurpose input/output pins for interfaces like UART, SPI, I2C, PWM, ADC, DAC.. This device can be purchased for less than $10, which is by far the cheapest out of the options thus far and still provides plenty of beneficial features.

3.1.5.2 ESP32-WROVER

ESP32-WROVER is designed for applications needing higher memory and processing power than the ESP32-WROOM board. There is a 4MB flash for storing programs and a 4MB PSRAM, which is helpful when the device needs more memory space. The CPU is dual-core based and operates at a frequency of 240MHZ which is more than enough power for our project. Like WROOM-32, it has Bluetooth wifi connectivity integrated into the board. It also hosts a variety of interfaces such as UART, SPI, I2C, PWM, ADC, DAC. Even if the price is higher than the WROOM-32 model, it is still affordable. The only negative aspect that one is likely to have as a result of having extra memory is that the power consumption will also rise, which could pose a potential problem.

3.1.5.3 ESP32-S2

The ESP32-S2 has a deep sleep current that can be as low as 5 µA, enabling it to be highly energy-efficient. Features like hardware encryption and booting allow the server to be used in secure operations. It is more affordable compared to ESP32-WROOM-32 and WROVER versions. However, it does not have Bluetooth and has a single core which means it has less processing power than the newer dual-core models. Apart from the cost and the low power consumption, this microcontroller has multiple disadvantages that make it less favorable than the other models.

Table 3.2. Part 2 Microcontroller Comparisons

	ESP32-WROOM-32	ESP32-WROVER	ESP32-S2
Processor	Dual-core 240 MHz	Dual-core 240 MHz	Single-core 240 MHz
RAM	520 KB SRAM	520 KB SRAM + 4 MB PSRAM	320 KB SRAM
Wi-Fi	Yes	Yes	Yes

Bluetooth	Yes	Yes	No
GPIO Pins	26	26	32
ADC	Yes	Yes	Yes
DAC	Yes	Yes	No
Power Consumption	0.528-1.056W	0.528-1.056W	0.429-1.056W

3.2 Temperature Sensors

As stated in our motivation section, maintaining the plant's health and ensuring it thrives in optimal growing conditions is our primary goal. One of the most critical environmental factors that must be meticulously monitored is temperature. Temperature significantly influences various physiological processes in plants, including photosynthesis, respiration, and transpiration. Therefore, ensuring that the plants are kept within their ideal temperature range is crucial for promoting healthy growth and maximizing yield. A stable temperature environment helps prevent stress and diseases that can hinder plant development.

The temperature sensor thus plays an indispensable role in our project. It is not merely a component but a cornerstone of our environmental monitoring system, providing real-time data that allows for precise control over the growing conditions. Given its importance, selecting the right sensor is a pivotal decision that impacts the overall success of the project. The sensor we choose must meet several key criteria: it must be highly accurate to ensure precise temperature readings, which are essential for maintaining the delicate balance required for optimal plant growth. Additionally, the sensor must consume low energy to ensure the system remains efficient and sustainable over long periods. Finally, it must be easy to integrate with the rest of the system, ensuring seamless communication with other components and the central control unit. This integration facilitates the efficient processing of data and the timely adjustment of environmental conditions, ultimately contributing to the health and productivity of the plants.

3.2.1 DHT22

The DHT22 is a low-cost, easy-to-use temperature and humidity sensor. It will measure temperature from -40 to 80°C with an error margin of ±0.5°C. The DHT22 is digital, and its data output is available through a single wire for easy interfacing with a microcontroller. However, it has a relatively low sampling rate with a maximum interval of two seconds for each sample, which isn’t the biggest problem since we don’t need to have it constantly updating into our database. This device is larger than other sensors, which could potentially cause a problem with space in our design.

3.2.2 DHT11

This temperature and humidity sensor is a downgraded version of the previously mentioned DHT22. It provides temperature readings from 0 to 50°C with an error margin of ±2°C. This sensor utilizes a single wire interface which makes the connection simpler and easier to integrate. It has lower accuracy in comparison to all other sensors but it is still accurate enough for our project scope. Overall, its affordability and smaller size compared to its counterpart makes it a competitive option for our needs.

3.2.3 DS18B20

The DS18B20 is a digital temperature sensor that can operate in the temperature range of -55 to 125°C with an error margin of ±0.5°C It has a one-wire interface where several devices can be connected in parallel to a single conductive data line, which is a great benefit for this project since we intend on having multiple sensors involved. The DS18B20 is also available in various waterproof formats and is capable of functioning in humid places, which is great for where we are located. This sensor has a programmable data resolution, which can be set to 9, 10, 11, or 12 bits, allowing us to balance precision and conversion time. The DS18B20 could be powerful and flexible enough to be used for our project.

3.2.4 TMP36

The TMP36 provides an analog voltage level according to the temperature level. It is operable in the -40 to 125°C range with an error margin of up to ±1°C. The TMP36 is not very complex, and the signal can be easily connected with the microcontroller with ADC. This allows for it to have very fast response times while still keeping the power consumption low. However, because it is an analog sensor it is prone to noise, which must be filtered and may add additional complications to the project.

3.2.5 BME280

The BME280 focuses on having high accuracy in temperature, humidity, and pressure sensing. Its operating temperature is -40C to 85C, and the measuring accuracy varies around ± 1C. It can also have a two-wire serial interface, such as I2C, or a four-wire serial interface, such as SPI. Although this is one of the more expensive options it could eliminate the need for multiple sensors and still provide accurate readings.

Table 3.3. Temperature Sensor Comparisons

	DHT22	DS18B20	TMP36	BME280	MCP9808	DHT11

Temperature Range	-40 to 80°C	-55 to 125°C	-40 to 125°C	-40 to 85°C	-40 to 125°C	0 to 50°C
Accuracy	±0.5°C	±0.5°C	±1°C	±1°C	±0.25°C	±2°C
Interface	Digital (Single-Wire)	Digital (One-Wire)	Analog	I2C/SPI	I2C	Digital (Single-Wire)
Power Consumption (average)	150µA	5µA	50µA	3.6µA	200µA	150µA
Sampling Rate	0.5 Hz	Programmable	Continuous	Continuous	Continuous	0.5 Hz

3.2.6 MCP9808

The MCP9808 is a very accurate digital temperature sensor for system temperatures between -40 and 125°C with an accuracy of ±0.25°C. This makes it the most accurate sensor out of all the options. It uses I2C to connect, and the user can set specific alerts for certain over/under temperature conditions. This sensor is a great option for our project due to its high accuracy and fair price.

3.3 Humidity Sensors

Correct humidity levels are necessary to ensure an optimal growing environment within our project. Too little could cause problems with hydration and stress levels, while too much can promote fungal growth and diseases. The sensor we choose needs to be accurate and reliable in order to prevent these circumstances from occurring.

3.3.1 SHT31

Sensirion's SHT31 is a humidity and temperature sensor. It determines humidity level from 0 to 100% with an accuracy of ±2%. The SHT31 delivers the data through an I2C interface, which makes it easy to integrate with any microcontroller. It is the most compact sensor by comparison which could be beneficial in our design.

3.3.2 BME280

The BME280 as mentioned before, is a sensor that has the capability of measuring humidity, temperature, and barometric pressure. The humidity is measured from 0 to 100% with an accuracy of ±3%. BME280 supports I2C or SPI data transfer modes, which is unique out of the ones researched and could make extrapolating the data easier. Its high precision and multi-functionality make this instrument the front-runner for our project.

3.3.3 HIH6130

The HIH6130 is one of the most stable and accurate digital humidity and temperature sensors produced by Honeywell International Inc. It is able to measure the humidity from 0 to 100 %, with an accuracy of ± 1.8% which is one of the best compared to the rest of these options. It utilizes I2C Digital output and is easily programmable to be connected to a variety of microcontrollers. Overall, it’s a great fit for our project and even though it’s the most expensive it is worth considering.

3.3.4 Si7021

The Si7021 by Silicon Labs is low power and can also detect humidity and temperature. Its sensor range is from 0 to 100% with an accuracy of up to ± 2%. Similar to every other humidity sensor, the Si7021 comes with digital outputs via I2C. This sensor is the second cheapest of all the options and requires the least current while still carrying similar peripherals which gives it value to be considered.

3.3.5 DHT11

As previously mentioned, this sensor combines measuring temperature and humidity. It provides humidity readings from 20 to 80% with an error margin of ±5%. Unlike all the other humidity sensors this uses a single-wire interface instead of I2C or SPI, which makes integration simple. The DHT11 made by Aosong Electronics may not look like the best option on paper but it still does everything we need while being the most affordable option out of these choices.

3.3.6 Key Takeaways

While conducting this research, it became apparent that many of the best humidity sensors on the market happen to double as a temperature sensor. This is due to the simple fact that temperature can affect the readings you get for humidity. Also, many sensors that were only able to sense the humidity had slightly higher margins of error thus making the choice to bundle temperature and humidity together a logical choice. This will save money in our budget which we could spend testing out multiple of these sensors.

Table 3.4. Humidity Sensor Comparisons

	SHT31	BME280	HIH6130	Si7021	DHT11
Accuracy	±2%	±3%	±1.8%	±2%	±5%
Power Consumption	150μA	3.6μA	1mA	2μA	150μA
Sampling Rate	2 Hz	1 Hz	1 Hz	0.5 Hz	1 Hz
Interface	I2C	I2C/SPI	I2C	I2C	Single-wire digital
Response Time	< 8 seconds	< 1 second	< 5 seconds	< 10 seconds	< 6 seconds
Operating Voltage	2.4V - 5.5V	1.8V - 3.6V	2.3V - 5.5V	1.9V - 3.6V	3V - 5.5V

3.4 PH Sensors

The pH sensor is one of the most crucial components in our hydroponic system design, playing a vital role in maintaining the health and growth of the plants. For the Deep Water Culture (DWC) hydroponics system being integrated, precise monitoring and regulation of pH levels are absolutely essential. This sensor ensures that the nutrient solution maintains an optimal pH range, allowing plants to effectively absorb the necessary nutrients. Any deviation from this range can hinder nutrient uptake, leading to nutrient deficiencies and potentially causing the plant to fail. Accurate and reliable pH readings are therefore indispensable to the overall success of the system.

Selecting the right pH sensor involves careful consideration of several factors, with reliability being a top priority. Many pH sensors on the market require frequent calibration to maintain accuracy, which can be a significant drawback for a system designed to be as self-sufficient as possible. In our project, we aim to minimize the need for manual intervention, ensuring that the hydroponic system can operate autonomously for extended periods. Therefore, we are focused on choosing a pH sensor that not only provides precise and consistent readings but also has a low maintenance requirement. This will help achieve our goal of creating a robust, user-friendly system that supports optimal plant growth with minimal effort from the user. The chosen sensor must demonstrate durability and stability in its readings over time, ensuring that the nutrient solution remains within the ideal pH range for the plants to thrive.

3.4.1 Atlas Scientific pH Probe

The pH Probe from Atlas Scientific is known for being one of the most precise and accurate among its competition. It's able to measure the pH from 0-14 and its precision within ±0.002pH. It can be used with many different microcontrollers through ADC. It is also very durable as it can be submerged in water for a long time and exhibits fantastic stability. These features are useful for our project specifically as we would be using it in the hydroponics system. However, it is relatively costly compared to other pH probes and requires proper calibration fairly often for optimum performance.

3.4.2 Gravity Analog pH Sensor

This sensor from DFRobot is one of the more popular pH sensors on the market. It is easily able to be integrated with any microcontroller that has ADC capabilities. It can measure pH in a range from 0-14 with an accuracy of ±0.1 pH. While its general performance is quite suitable for daily use, it lacks the precision, stability, and durability of professional-grade sensors and might need frequent recalibration. Even with these disadvantages, it is still worth considering due to its low cost and ease of use.

3.4.3 SEN0161 Analog pH Meter

This pH meter is made by the same company as the gravity analog and can also monitor the ph within the same range and accuracy. It also similarly can be used by any microcontroller with ADC capabilities. It separates itself by featuring a durable glass electrode and BNC connector, which enhances its reliability and longevity. In comparison to the other pH sensors, it's about average but has a good balance between its performance, the cost, and higher durability which makes it a viable option for our project.

3.4.4 Vernier pH Sensor

Similar to the previous sensors, it can measure pH from 1-14 and utilizes analog output. It boasts the second most accurate pH reading, sensing within ±0.01 pH. This sensor is most similar to the atlas in that it prioritizes more accurate readings and reliability. However, it is the most expensive out of all the options and in comparison to the atlas which is the second, it doesn't perform better enough to justify the extra budget.

Table 3.5. pH Sensor Comparisons

	Atlas Scientific pH Probe	Gravity Analog pH Sensor	SEN0161 Analog pH Meter	Vernier pH Sensor
Accuracy	±0.002 pH	±0.1 pH	±0.1 pH	±0.1 pH

Response Time	< 1 second	< 30 seconds	< 30 seconds	< 1 second
Maintenance Requirements	Weekly	Bi-Weekly	Monthly	Quarterly
Durability	18 months	12-18 months	12-24 months	24-36 months

3.5 Light Sensors

The light sensor will play a key role in our design by ensuring the plant receives the most optimal light frequencies necessary for healthy growth and development. This sensor is integral to our system as it continuously monitors the light conditions and adjusts the LED output to match the specific needs of the plant at various stages of its life cycle. For the plant to effectively undergo photosynthesis and ultimately flower, it is essential that the correct light frequencies are emitted from our LEDs. The light sensor provides the feedback needed to maintain these precise conditions.

The importance of the light sensor extends beyond merely providing adequate illumination. It ensures that the light intensity and spectrum are precisely controlled, preventing scenarios where the plant might receive too much or too little light. Excessive light can cause photoinhibition, damaging the plant cells and hindering growth, while insufficient light can lead to poor photosynthesis, stunted growth, and reduced flowering. By continuously adjusting the light output based on real-time data from the sensor, our system maintains an optimal light environment, thereby maximizing the plant's health and productivity. This makes the light sensor an indispensable component of our design, crucial for achieving the desired outcomes in plant growth and ensuring the overall success of our hydroponic system.

3.5.1 TSL2561

TSL2561 sensor is used for measuring infrared and whole-spectrum light with various ranges to determine the total intensity of a given light source. It can be used in different lighting situations ranging from 1 to 40,000 Lux for short-range detection. In addition, the sensor is easy to integrate since it uses I2C. It's only downside is that it requires careful positioning as to not oversaturate the sensor.

3.5.2 BH1750

The BH1750 is a dual-interface ambient light sensor that measures the light intensity from 1 to 65535 Lux. This is one of the most popular choices for light sensors because of its high precision and stability across different conditions. This sensor is also very compact which makes it easy to install within the growing area to calculate the amount of light reflecting off the plants. The BH1750 is a solid option for our project, it's only downside is that it struggles in low light concentrated areas which should not be a problem in our situation.

3.5.3 VEML7700

The VEML7700 is a 16-bit ambient light sensor designed for high accuracy and can measure light intensity ranging between 0 and 120000 Lux. In the VEML7700, the user can change configurable settings, including programmable gain and integration times which gives us more control of the threshold we want to measure. With the extra options does come a downside though as this will be harder to integrate as it takes extra parameters to utilize the enhanced features.

3.5.4 OPT3001

The OPT3001 is a digital ambient light sensor with a measurement range from 0.01 to 83,000 Lux. It uses I2C for communication and offers raw Lux values that can be read directly, which makes integration and analysis easier. Another feature to be noted is that the OPT3001 has the lowest power consumption compared to the other sensors. Its performance and system integration make it a solid choice for our project.

3.5.5 APDS-9301

The APDS-9301 is another sensor that works as a light-to-digital converter with light intensity and a digital output format that can be written via I2C. It encompasses visible and infrared light, making it a full spectrum analysis of the light source and can measure within the range of 3 - 65535 Lux. Overall, it's average but does everything our project would need so it is a worthy option.

Table 3.6. Light Sensor Comparisons

	TSL2561	BH1750	VEML7700	OPT3001	APDS-9301
Light Range	0.1 to 40,000 Lux	1 to 65,535 Lux	0 to 120,000 Lux	0.01 to 83,000 Lux	3 to 65,535 Lux
Spectral Range	400 - 1100 nm	320 - 1050 nm	320 - 1050 nm	300 - 1000 nm	400 - 700 nm
Operating Voltage	2.7V - 3.6V	2.4V - 3.6V	2.5V - 3.6V	1.6V - 3.6V	2.4V - 3.6V
Supply	0.6 mA	0.12 mA	0.001 mA	0.003 mA	0.16 mA

Current					
Size	3.1 x 6.4 x 2.6 mm	3.9 x 2.4 x 1.3 mm	2 x 2 x 0.75 mm	2 x 2 x 0.65 mm	3.94 x 2.36 x 1.1 mm

3.6 Water Pumps

A water pump is essential for the system to maintain a continuous flow of water, ensure adequate aeration, and push water through the filtration media. The consistent movement of water is vital for nutrient distribution and oxygenation, both of which are crucial for healthy plant growth. There is a wide array of water pump types available on the market, each with its own unique characteristics and advantages. Selecting the right pump requires a thorough understanding of these characteristics to ensure that the pump meets the specific needs of the hydroponic system.

One of the key distinctions between different types of water pumps is the mechanism by which they move water. This primarily divides pumps into two categories: kinetic pumps and positive displacement pumps. Kinetic pumps, such as centrifugal pumps, operate by imparting velocity to the water, converting this kinetic energy into pressure to move the water. These pumps are typically used for high-flow, low-pressure applications. On the other hand, positive displacement pumps, such as diaphragm or piston pumps, work by trapping a fixed amount of water and then displacing it, making them suitable for high-pressure, low-flow applications. Understanding these differences is crucial in selecting the appropriate pump, as the efficiency and effectiveness of the hydroponic system heavily depend on the pump’s ability to maintain the desired water flow and pressure.

3.6.1 Centrifugal Pumps

Centrifugal pumps convert the energy of a rotating impeller to create kinetic energy in the fluid. These pumps are ideal for low-viscosity fluids that require a high flow rate. Centrifugal pumps are not ideal for high-viscosity fluids or scenarios where precise volumes of fluid are to be moved due to a sensitivity in the flow rate due to changes in pressure. Also, centrifugal pumps are typically not able to self-prime which is the process of removing air from the system for the pump to interact with the fluid it intends to interact with.

3.6.1.1 Standard Centrifugal Pumps

A standard centrifugal pump works by directly interacting with the fluid by rotating an impeller which creates a rotational kinetic energy accelerating the fluid outward from the center of rotation and thus increasing in pressure. The fluid of high pressure is then output through a discharge pipe, while new fluid is flowing through an intake pipe.

Table 3.7. Comparison of Centrifugal Pumps

Feature	Centrifugal Pump	Submersible Pump
Flow rate	High	Moderate
Power Consumption	High	Moderate
Noise Level	High	Low
Reliability	High	Moderate
Self-priming	No	Less Frequent Priming Needed
Pressure Capability	Moderate	Moderate
Maintenance	Low	Moderate
Cost	Medium	Low

3.6.1.2 Submersible Pumps

A type of centrifugal pump that is designed to operate fully submerged in the fluid it intends to move. This provides a few additional advantages in that the noise level is dampened when compared with an external water pump. Also, because the pump is located in the liquid it intends to move, the need to prime the pump would be less frequent. However, submersible pumps may require additional maintenance as they are installed in the fluid or a more harsh environment and they typically have less pumping power than an equivalent external centrifugal pump.

3.6.2 Positive Displacement Pumps

A positive displacement pump moves fluid by trapping a fixed volume and displacing it through various mechanisms such as pistons, diaphragms, gears, screws, or vanes. Positive displacement pumps are ideal for scenarios where a precise or measurable amount of fluid is to be moved. Positive displacement pumps are ideal for high-viscosity fluids and the flow rate is not impacted by high pressure. These pumps also have the ability to self-prime. Additionally, because the moving parts of the motor do not need to come in direct contact with the fluid, these pumps can be used in conditions where food or health considerations are required. However, positive displacement pumps are higher in complexity, which affects both cost and reliability, and have a lower efficiency when compared with centrifugal pumps.

Table 3.8. Comparison of Positive Displacement Pumps

Feature	Reciprocating Pump	Rotary Pump
Flow rate	Low to High	Moderate to High
Power Consumption	High	Low to Medium
Noise Level	High	Low to Medium
Reliability	Moderate	Moderate
Self-priming	Yes	Yes
Pressure Capability	High	Moderate to High
Flow Pulsation	High	Low
Maintenance	High	Moderate
Cost	High	Moderate

3.6.2.1 Reciprocating Pumps

This pump category uses a piston or diaphragm to move fluid. These can be pneumatically or electrically operated and work by alternating the open and close state of the trapped fluid in chambers. To create suction, a piston or diaphragm opens an inlet valve and closes the outlet valve. Oppositely, to create the discharge the piston or diaphragm compresses and opens the outlet valve.

3.6.2.2 Rotary Pumps

This pump category utilizes gears, screws, or vanes to move fluid. In comparison, reciprocating pumps utilize a back and forth or reciprocating motion to move fluids while a rotary pump is utilizing rotational energy to move gears, screws, or vanes to push and pull fluid through the motor housing.

Table 3.9. Key Differences Between Centrifugal Pumps and Positive Displacement Pumps

Feature	Centrifugal Pumps	Positive Displacement Pumps
Energy Transfer Method	Convert kinetic energy to pressure	Displaces a fixed volume per cycle
Flow Rate	Variable with system pressure	Constant, regardless of pressure

Flow Characteristics	Continuous, can vary	Pulsating, generally constant
Pressure Capability	Lower pressure	Higher pressure
Applications	High flow, low pressure	High pressure, precise flow control

3.7 Water Level Sensors

The hydroponic system will be suspended in water, serving as the primary source of nutrients essential for sustaining plant life. Proper monitoring of the water reservoir is crucial to ensure it is neither empty nor below a pre-determined lower limit, as both conditions can adversely affect plant health and the operation of the water pump. When water pumps intake air instead of water, they can experience cavitation, loss of prime, and potentially premature part failure, leading to costly repairs and system downtime.

Therefore, it is vital to implement an effective method for monitoring the water level within the system. Understanding the various types of water level sensors available is key to selecting the most suitable option for the system's specific demands. Water level sensors can be categorized into two main types: point level measurement sensors and continuous level sensors. Point level sensors are designed to detect specific levels within the reservoir, triggering alerts when water reaches these critical points. In contrast, continuous level sensors provide ongoing monitoring of the water level, offering real-time data and enabling precise control over the water supply to maintain optimal conditions for plant growth and pump operation. This distinction is crucial in ensuring the system remains efficient and reliable, safeguarding both plant health and equipment longevity.

3.7.1 Point Level Sensors

Point level sensors are used to mark a single discrete point of liquid height. This point would be predetermined by the design and typically the predetermined point is used to either monitor an upper (high alarm) or lower limit (low alarm) of liquid height. This would allow the system to monitor a reservoir overfill or reservoir low condition and allow the user or system to act to remedy the situation before it becomes detrimental to system stability. This sensor would be monitored by a discrete digital input.

3.7.2 Continuous Level Sensors

Continuous level sensors are used to monitor the fluid level within a range rather than a specific point. Typical continuous level sensors would connect to an analog input that is scaled by the output of the sensor from 4-20 mA or 0-10 V to a range of the percentage of the reservoir fill state.

3.7.3 Types of Water Level Sensors

3.7.3.1 Float Switch

This point level sensor has a mechanical switch that is opened or closed via a device that floats on the surface of the measured fluid. Typically, the switch is activated mechanically or magnetically. The float switch requires contact with the surface of the liquid.

3.7.3.2 Optical Sensor

Typically, a point level sensor that contains an infrared LED that emits light that is directed at the liquid surface. The sensor can detect light via a phototransistor that measures the intensity of the returned light. A transparent window allows the light beam to pass through the sensor body into the container and when the light reaches the boundary between the window and the liquid it refracts differently on water than it does in air. If the water level is empty or low, the beam of light will not be refracted by any liquid and therefore the phototransistor will detect a strong light signal. Conversely, if the water level is high the signal will be weaker as the light will refract at a higher value in water than in air.

3.7.3.3 Non-Contact Ultrasonic Sensors

A type of continuous level sensor that emits a high-frequency sound wave and measures the travel time of the sound wave from the sensor to the liquids surface to determine the distance or height of the liquid by knowing the speed of the sound. If this sensor type is used, a clear path between the sensor and the liquid surface must be maintained.

3.7.3.4 Contact Ultrasonic Sensors

A type of point level sensor that operates similarly to the non-contact ultrasonic sensor except that a probe from the sensor is in contact with the surface of the liquid. An advantage of this sensor type is that it has the high accuracy of the non-contact variant but does not require a direct line of sight to be maintained between the sensor and the liquid surface.

Table 3.10. Key Differences Between Water Level Sensor Types

Feature	**Float Switch**	**Non-Contact UltraSonic**	**Contact UltraSonic**	**Capacitance**	**Optical**

Category	Point Level	Continuous	Point Level	Either	Point Level
Measurement Method	Mechanical float triggers switch	Ultrasonic sound waves echo	Ultrasonic sound waves via direct contact	Electrical capacitance changes due to liquid level	Detects changes in light due to liquid level
Liquid Contact	Yes	No	Yes	Either	Either
Accuracy	Moderate	High	High	Moderate to High	Moderate
Complexity	Low	Moderate	Moderate	Moderate	Moderate
Output Type	Digital	Analog or Digital	Analog or Digital	Analog or Digital	Analog or Digital
Maintenance	Low	Low	Low	Moderate	Low
Advantage	Simple, reliable, cost	No contact, accuracy	Accuracy, harsh environment	Works with various liquids	Simple, compact, cost
Disadvantage	Point level, liquid limitation	Clear path required, foam distortion	Contact required, liquid limitation	Tank wall or sensor coating can interfere	Turbulent liquids, affected by light
Cost	Low	Moderate	Moderate	Moderate to High	Low

3.7.3.5 Capacitance Level Sensors

This sensor can be either a point level sensor or a continuous level sensor. The sensor has a probe that extends into the liquid and the probe and the reservoir wall act as the plates of a capacitor while the liquid is the dielectric medium with a known or set dielectric constant value. The sensor applies a radio frequency signal to the capacitance circuit to sense the liquid.

3.8 Motor Control Relays

A relay functions as an electrically operated switch, playing a crucial role in controlling circuits by opening and closing contacts in another circuit. This switching mechanism can take two forms: normally open (NO) and normally closed (NC). In the normally open configuration, the switch remains open, interrupting the circuit until it is energized, at which point the switch closes, allowing current to flow. Conversely, in the normally closed configuration, the switch stays closed, permitting current to flow until it is energized, which then opens the switch and breaks the circuit.

The operation of the relay is managed through an output from the microcontroller unit (MCU), which sends the necessary electrical signal to trigger the switch. There are several types of relays commonly employed in various applications, each with distinct characteristics suited to specific needs. Electromagnetic relays use an electromagnet to mechanically operate the switch and are valued for their reliability and simplicity. Solid-state relays, on the other hand, utilize semiconductor devices to perform the switching, offering faster response times and longer lifespans without mechanical wear. Reed relays feature a pair of ferromagnetic reed switches enclosed in a glass tube, which are actuated by an external magnetic field, providing a compact and highly sensitive solution. Latching relays, distinct in their ability to maintain their position after being actuated, do not require continuous power to stay in either the open or closed state, making them energy efficient for applications where maintaining a specific state without continuous power is advantageous.

3.8.1 Electromagnetic Relays

These relays work on the principle of electromagnetic induction with either an AC or DC source. There is a coil that induces a magnetic field when energized, and this magnetic field either attracts or repels an armature (movable contact) to close or open contacts in the same manner as a switch. A few advantages of electromagnetic relays is that they are low cost, can withstand large inrush currents, and they are not susceptible to external electromagnetic interference. However, there are a few disadvantages in that they are typically slower than other variants, larger in size, and have a shorter lifespan as the armature is a mechanical wear part. Electromagnetic relays can be further subdivided into latching and non-latching variants.

3.8.1.1 Latching

Latching relays can remain in the last position when de-energized and will remain in this state until there is a command to invert the prior input. An example application of a latching circuit could be an emergency stop circuit where the user would want the system to remain de-energized until the system is clear and ready to be reset and back in operation.

3.8.1.2 Non-Latching

The non-latching relay has a spring or magnet that maintains its initial position unless the coil is energized. These can be thought of as momentary switches such as used in a doorbell or arcade button.

3.8.2 Solid State Relays

This relay variant has a very similar goal as the electromagnetic relay but achieves its goal without moving mechanical parts. The key components are an LED and a photosensitive MOSFET. When current flows through the LED, it lights up and is detected by the MOSFET triggering either a TRIAC (Triode for AC current) or SCR (Silicon Controlled Rectifier) that acts as the armature. There are several advantages of solid state relays such as fast switching speed, silent switching, small package size, and increased reliability by removing the mechanical wear part. The drawbacks are that the contact resistance is high, usually above 100 ohms, generating more heat, and these relays come at a higher cost.

Table 3.11. Key Differences Between Motor Control Relay Types

Feature	**Electromagnetic (Non-Latching)**	**Electromagnetic (Latching)**	**Solid State**	**Reed**
Operation	Electromagnet moves armature	Electromagnet moves armature and sets state	LED and MOSFET controlled	Magnetic field controls reeds in sealed tube
Latching	No	Yes	No	No
Control Signal	Continuous current to maintain state	One pulse to set or reset	Low power control signal	Low power control signal
Control Compatibility	Compatible with MCUs	Compatible with MCUs	Requires compatible driver circuit	Requires compatible driver circuit
Contact Type	Mechanical armature	Mechanical armature	Electronic	Mechanical reeds
Output Current Capacity	High	High	Very High	Moderate
Switching Speed	Moderate	Moderate	Very Fast	Very Fast
On Resistance	Low to Moderate	Low to Moderate	Very Low	Low

Off-State Leakage Current	Low	Low	Very Low	Very Low
Typical Applications	Point level, liquid limitation	Clear path required, foam distortion	Contact required, liquid limitation	Tank wall or sensor coating can interfere
Cost	Low to Moderate	Moderate	Moderate to High	Moderate

3.8.3 Reed Relays

A reed relay consists of magnetic strips (known as reed) contained within an inert gas filled tube. There is a magnetic field applied to a coil around the tube that makes the reeds in the tube move via the induced magnetic field to create a switching action. The advantages of reed relays are that they are low power consumption and package size, great for resistance to environmental factors as the inert gas isolates the moving reeds from any corrosive action, and the switching speed is 10 times faster than a typical electromechanical relay.

3.9 Optical Design

This section dives into the BASIL HERB system's optical design, utilizing advanced photonic components to create an ideal indoor herb-growing environment. We'll explore the functionalities of photoresistors, color sensors, and LED systems, along with their theoretical foundations and practical applications within the system. Comparisons of chosen and calculated specifications showcase design feasibility. To illustrate, we present two demonstration setups: one using a photoresistor and RGB LED for basic light control, and another integrating color sensors and LED strips for plant health-based light adjustments.

3.9.1 Introduction

The optical design of our system plays a crucial role in ensuring optimal growth conditions for indoor herb cultivation. By integrating advanced photonic components, such as a photoresistor, a color sensor, and an adaptive LED lighting system, we can closely monitor and adjust the environmental parameters that significantly impact plant health and growth. This section delves into the theoretical underpinnings and practical applications of these components, providing a comprehensive overview of their functions and interactions within the system.

Light is a fundamental factor in the process of photosynthesis, where plants convert light energy into chemical energy. Different wavelengths of light have varying effects on plant growth, with blue light (450-480 nm) promoting vegetative growth and red light (620-750 nm) encouraging flowering and fruiting.

Understanding and harnessing these effects is key to developing an efficient and effective indoor growth system. Our LED lighting system is designed to deliver the optimal light spectrum needed for each growth stage, ensuring that plants receive the appropriate wavelengths to maximize photosynthesis and overall health.

In addition to providing the necessary light spectrum, it is essential to continuously monitor light intensity and plant health. The photoresistor in our system detects ambient light levels, ensuring that the plants receive adequate light at all times. This sensor is crucial for maintaining consistent light conditions, even when external factors change. The color sensor monitors the color of the leaves, which is a vital indicator of the plant's health and any potential stress factors. For instance, yellowing leaves can indicate nitrogen deficiency, while purpling leaves might suggest phosphorus deficiency. By detecting these color variations, the system can alert users to potential nutrient deficiencies or other stress factors, enabling timely interventions to maintain optimal plant health.

The LED system, capable of dimming based on feedback from both the photoresistor and the color sensor, ensures that the plants receive the optimal light intensity and spectrum tailored to their current needs. Through the integration of these photonic components, our system creates a controlled environment that adapts to the dynamic needs of indoor herb cultivation. This adaptive lighting not only promotes healthier plant growth but also makes it possible for even the busiest individuals to enjoy fresh, organic herbs year-round, with minimal effort and intervention required.

3.9.2 Theory

Principles of Photonics in Plant Growth

Photonics, the science of light, plays a vital role in plant biology, particularly in the process of photosynthesis. Photosynthesis is the mechanism by which plants convert light energy into chemical energy, using light-absorbing pigments such as chlorophyll. The efficiency of photosynthesis is highly dependent on the light spectrum, with specific wavelengths of light promoting different aspects of plant growth.

Blue Light (450-480 nm): Blue light is essential for vegetative growth. It influences the development of strong stems, healthy leaves, and overall plant structure. Blue light activates photoreceptors like cryptochromes and phototropins, which regulate various growth processes.

Red Light (620-750 nm): Red light is crucial for flowering and fruiting. It affects the plant's photoperiodism, the physiological reaction to the length of day or night and promotes flowering in long-day plants. Phytochromes, the photoreceptors responsive to red light, play a significant role in these processes.

The integration of these specific wavelengths into our system ensures that plants receive the optimal light conditions required for their growth stages.

Interaction Between Light Intensity and Electromagnetic Fields

In addition to the spectrum of light, the intensity of light and the presence of electromagnetic fields (EMFs) play critical roles in influencing plant growth. Light intensity directly affects the rate of photosynthesis, with higher intensities generally promoting more vigorous growth, provided they are within the plant's tolerance limits. Adequate light intensity ensures that plants receive enough energy to drive essential processes such as photosynthesis, respiration, and transpiration, which are vital for healthy growth and development.

Moreover, the role of EMFs in plant growth is an area of growing interest and research. Studies have demonstrated that EMFs can significantly stimulate root growth and biomass accumulation in plants. This stimulation enhances the plants' ability to absorb nutrients and water more efficiently, leading to healthier and more robust growth. The presence of EMFs has been shown to activate certain biochemical pathways and cellular activities that promote stronger root systems and increased nutrient uptake, which are crucial for sustaining high growth rates and improving overall plant health.

Our system incorporates an advanced EMF generator to harness these benefits, creating a synergistic effect when used in tandem with our LED lighting system. The LED system provides the necessary light spectrum and intensity for optimal photosynthesis, while the EMF generator enhances the plant's physiological functions, particularly in the root zone. By integrating these technologies, our system creates a highly conducive environment for plant growth, optimizing conditions to support vigorous and healthy development. This combination ensures that plants not only receive the appropriate light for energy but also benefit from enhanced nutrient and water absorption capabilities, leading to superior growth outcomes.

Detection of Nutrient Deficiencies via Leaf Color

The color of plant leaves is a critical indicator of their health and can signal various nutrient deficiencies:

- Yellow Leaves (570-590 nm): Often indicate a nitrogen deficiency.
- Purple Leaves (380-450 nm): Can signal a phosphorus deficiency.
- Brown Edges (590-620 nm): May suggest potassium deficiency.

Our system’s color sensor detects these variations in leaf color, allowing for timely interventions to address nutrient deficiencies and other stress factors.

Optical/Photonic Equations

1. Equation for Light Intensity and Resistance in Photoresistor:

$$\boldsymbol{V_{out} = V_{in} \times \frac{R_{photo}}{R_{photo} \times R_{fixed}}}$$

- V_{out} = Output Voltage
- V_{in} = Input Voltage
- R_{photo} = Resistance of photoresistor
- R_{fixed} = Resistance of combined resistors

Equation 3.1. Light Intensity and Resistance in Photoresistor

This equation describes how the resistance of the photoresistor changes with varying light intensity. As light intensity increases, the resistance decreases, allowing the system to detect ambient light levels accurately.

2. Spectral Response Equation for Color Sensor:

$$S(\lambda) = \frac{P(\lambda)}{E(\lambda)}$$

- $S(\lambda)$ = Sensor sensitivity at wavelength λ
- $P(\lambda)$ = Power of the detected light at wavelength λ
- $E(\lambda)$ = Irradiance at wavelength λ

Equation 3.2. Spectral Response Equation for Color Sensor

This equation defines the sensitivity of the color sensor across different wavelengths. The sensor's ability to detect variations in light intensity and color helps monitor the health of the plant by assessing the color of its leaves.

3. Equation for LED Light Output:

$$P_{LED} = I_{LED} \times V_{LED}$$

- P_{LED}: Power output of the LED
- I_{LED}: Current through the LED
- V_{LED}: Voltage across the LED

Equation 3.3. LED Light Output

This equation calculates the power output of the LED system, which is crucial for determining the light intensity provided to the plants. The LED light output is adjusted based on feedback from the color sensor and photoresistor to ensure optimal lighting conditions.

By understanding and applying these principles and equations, our system can effectively monitor and adjust the light environment, ensuring that plants receive the precise conditions needed for healthy growth. This theoretical foundation supports the practical implementation of the optical design components, ultimately contributing to the system's overall efficiency and effectiveness.

3.9.3 Components and Functionality

3.9.3.1 Photoresistor

- **Function:** The photoresistor in our system is designed to detect ambient light levels to determine whether supplemental lighting is required. It ensures that plants receive adequate light at all times, whether from natural sources or the integrated LED system.
- **Theory of Operation:** The photoresistor operates on the principle that its resistance decreases with increasing light intensity. This relationship allows the system to measure light intensity accurately.
- **Placement and Integration:** The photoresistor is strategically placed within the system to capture the ambient light levels surrounding the plants. This placement ensures accurate detection and helps in adjusting the artificial lighting as needed.
- **Simulation Software:** Zemax OpticStudio is used to simulate and analyze the light intensity detected by the photoresistor. This ensures accurate monitoring of ambient light levels.
- **Equation:** $\boldsymbol{V_{out} = V_{in} \times \frac{R_{photo}}{R_{photo} \times R_{fixed}}}$

  This voltage divider equation describes how the output voltage varies with the photoresistor's resistance, which in turn depends on the ambient light intensity.

- **Prototype Demo:**
  - **Setup:** A photoresistor connected to an Arduino board.
  - **Operation:** When the photoresistor is covered, indicating low light, the Arduino lights up an LED.

3.9.3.2 Color Sensor

- **Function:** The color sensor detects the color of the leaves, providing critical data on plant health. Variations in leaf color can indicate issues such as nutrient deficiencies, overexposure to light, or other stress factors.
- **Theory of Operation:** The color sensor measures reflected light across different wavelengths, allowing it to determine the color and intensity of light reflected by the leaves. This data is used to assess the plant’s health and make necessary adjustments to the light and EMF settings.
- **Key Parameters:** The color sensor's spectral sensitivity covers the visible light range (380-780 nm), ensuring high accuracy in detecting variations in leaf color. It provides essential data that helps in monitoring and optimizing plant health.
- **Detection of Nutrient Deficiencies via Leaf Color:**

        - Yellow Leaves (570-590 nm): Often indicate a nitrogen deficiency.
        - Purple Leaves (380-450 nm): Can signal a phosphorus deficiency.
        - Brown Edges (590-620 nm): May suggest potassium deficiency.
- **Simulation Software:** Zemax OpticStudio used to simulate the detection of RGB values from plant leaves. This ensures accurate monitoring of plant health based on leaf color.
- **Equation:** $S(\lambda) = \frac{P(\lambda)}{E(\lambda)}$

  This equation defines the sensitivity of the color sensor across different wavelengths. The sensor's ability to detect variations in light intensity and color helps monitor the health of the plant by assessing the color of its leaves.

- **Prototype Demo:**
    - Setup: A color sensor connected to an Arduino (with the LED strip).
    - Operation: The color sensor detects green or brown paper and adjusts the LED strip color to blue for green (healthy) or red for brown (unhealthy).

3.9.3.3 LED System

- **Function:** The LED system provides artificial lighting to the plants, which can be dimmed based on the feedback from the photoresistor and color sensor. It ensures that plants receive the optimal light spectrum and intensity required for their growth stages.
- **Theory of Operation:** The LEDs emit light in specific wavelengths, particularly in the blue (450-480 nm) and red (620-750 nm) spectra, which are essential for vegetative growth and flowering. The system can adjust the light intensity and duration based on the real-time data from the sensors.
- **Control Mechanism:** The LED system is controlled through an adaptive mechanism that dims or brightens the lights based on the color of the plant leaves and the intensity of the EM field generator. This dynamic adjustment helps in maintaining optimal growth conditions by providing the right amount and type of light at all times.
- **Simulation Software:** Zemax OpticStudio is used to simulate light distribution and intensity of the RGB LED strip. This software ensures the light is evenly distributed and at the correct intensity for optimal basil growth.
- **Equation:** $P_{LED} = I_{LED} \times V_{LED}$

  This equation calculates the power output of the LED system, which is crucial for determining the light intensity provided to the plants. The LED light output is adjusted based on feedback from the color sensor and photoresistor to ensure optimal lighting conditions.

- **Prototype Demo:**
    - **Setup:** An LED strip connected to an Arduino (with the color sensor as stated previously).

- **Operation:** The color sensor detects green or brown paper and adjusts the LED strip color to blue for green (healthy) or red for brown (unhealthy).

3.9.3.4 Integration with EMF Generator

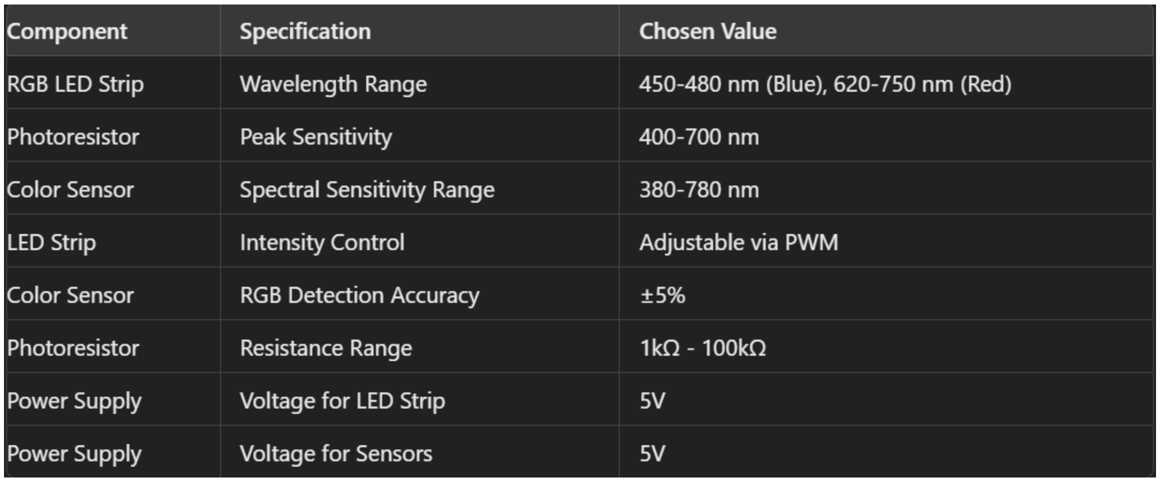
- **Function:** The EMF generator works in tandem with the LED system to enhance plant growth. By emitting electromagnetic fields, it stimulates root growth and biomass accumulation, further promoting healthy plant development.
- **Theory of Operation:** The EMF generator produces frequencies between 0.1 Hz and 30 Hz, which have been shown to positively affect plant growth. These frequencies enhance nutrient uptake and overall plant vitality.
- **Interaction with LED System:** The intensity of the LED lighting is adjusted based on the output from the EMF generator and the feedback from the color sensor. This integration ensures that the plants receive the optimal combination of light and EMF stimulation, leading to improved growth rates and health.

2.9.3.5 System Overview

- **Real-time Monitoring and Control:** Our system continuously monitors the light intensity and leaf color using the photoresistor and color sensor. The data collected is used to adjust the LED lighting and EMF generator settings in real-time.
- **User Interface:** All sensor data and system adjustments are accessible through a user-friendly web application. This interface allows users to monitor plant health, receive alerts, and manually adjust settings if needed.
- **Automation and Efficiency:** The integration of these components ensures that our system operates efficiently with minimal user intervention. The automated adjustments based on real-time data create an optimal environment for plant growth, making it easy for users to maintain healthy, thriving plants.

3.9.4: Engineering Requirement Specifications and Performance

Table 3.12. Tabulated ERS for Optical Components



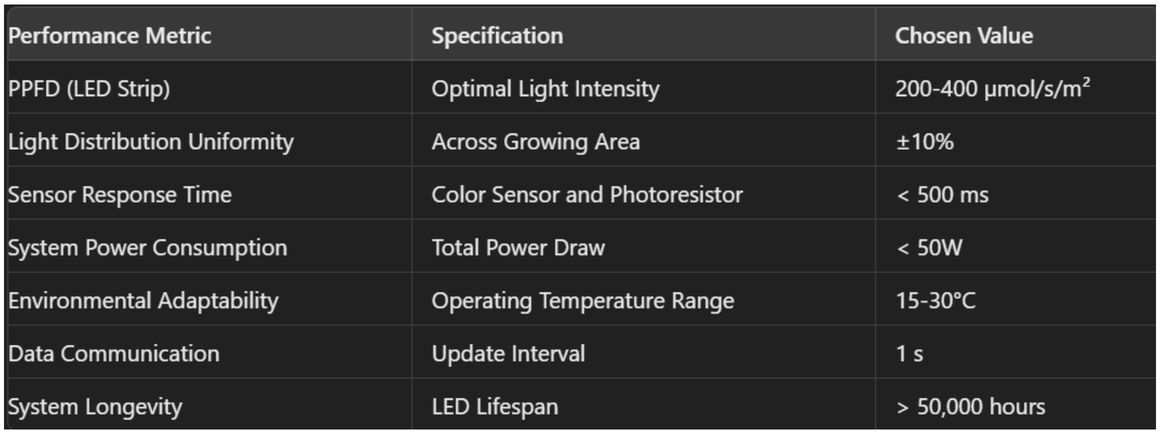
Component	Specification	Chosen Value
RGB LED Strip	Wavelength Range	450-480 nm (Blue), 620-750 nm (Red)
Photoresistor	Peak Sensitivity	400-700 nm
Color Sensor	Spectral Sensitivity Range	380-780 nm
LED Strip	Intensity Control	Adjustable via PWM
Color Sensor	RGB Detection Accuracy	±5%
Photoresistor	Resistance Range	1kΩ - 100kΩ
Power Supply	Voltage for LED Strip	5V
Power Supply	Voltage for Sensors	5V

This table lists the chosen engineering requirement specifications (ERS) for each optical component. The values were selected based on industry standards, research findings, and reputable sources to ensure optimal performance and compatibility within the system.

For instance, the RGB LED strip requires a wavelength range of 450-480 nm for blue light and 620-750 nm for red light, which are critical for vegetative growth and flowering, respectively. These specifications are supported by studies indicating the effectiveness of these wavelengths in promoting plant growth [30].
The power supply for both the LED strip and sensors is standardized at 5 volts to maintain consistency and simplify power management. The color sensor's spectral sensitivity range of 380-780 nm covers the visible spectrum, allowing it to detect various leaf colors associated with nutrient deficiencies:

- Yellow Leaves (570-590 nm): Indicate nitrogen deficiency [31].
- Purple Leaves (380-450 nm): Indicate phosphorus deficiency [32].
- Brown Edges (590-620 nm): Suggest potassium deficiency [33].

Table 3.13. Tabulated ERS for System Performance

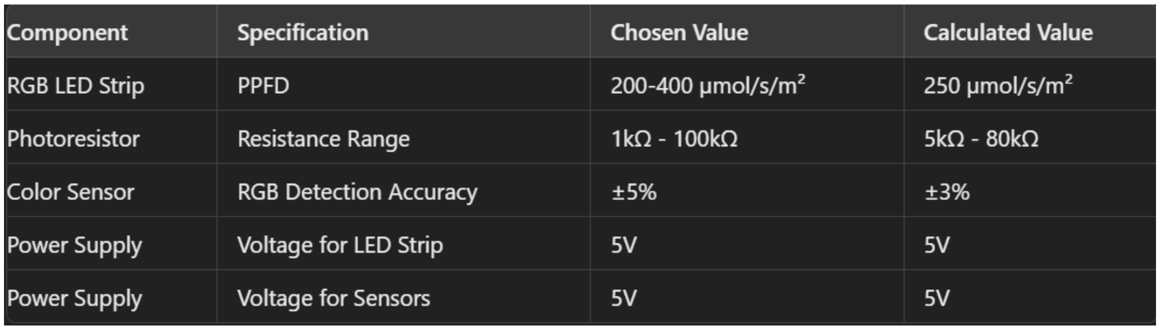


Performance Metric	Specification	Chosen Value
PPFD (LED Strip)	Optimal Light Intensity	200-400 μmol/s/m²
Light Distribution Uniformity	Across Growing Area	±10%
Sensor Response Time	Color Sensor and Photoresistor	< 500 ms
System Power Consumption	Total Power Draw	< 50W
Environmental Adaptability	Operating Temperature Range	15-30°C
Data Communication	Update Interval	1 s
System Longevity	LED Lifespan	> 50,000 hours

Explanation: This table outlines the performance specifications for the overall optical system. These metrics ensure that the system can provide the optimal light intensity and distribution, respond quickly to changes, operate efficiently within power constraints, and maintain longevity and reliability in various environmental conditions.

For example, the PPFD (Photosynthetic Photon Flux Density) for the LED strip is specified to be between 200-400 μmol/s/m², which is essential for optimal plant growth. This range is based on horticultural research and industry standards for indoor plant lighting [34]. The system's total power consumption is kept under 50W to ensure energy efficiency, aligning with best practices for sustainable energy use in automated plant systems [35].

Table 3.14. Tabulated Comparison of Specifications



Component	Specification	Chosen Value	Calculated Value
RGB LED Strip	PPFD	200-400 μmol/s/m²	250 μmol/s/m²
Photoresistor	Resistance Range	1kΩ - 100kΩ	5kΩ - 80kΩ
Color Sensor	RGB Detection Accuracy	±5%	±3%
Power Supply	Voltage for LED Strip	5V	5V
Power Supply	Voltage for Sensors	5V	5V

Explanation: This table compares the chosen specifications with the calculated values for the optical components. The comparison shows that the calculated values meet or exceed the chosen specifications, confirming that the design is feasible and the components will perform as required.

The chosen specification for the Photosynthetic Photon Flux Density (PPFD) of the RGB LED strip is between 200-400 μmol/s/m². The calculated PPFD for our

system is approximately 250 µmol/s/m², which falls well within the desired range, ensuring effective plant growth conditions.
The photoresistor's resistance range is designed to detect ambient light levels accurately. The chosen range of 1kΩ - 100kΩ allows for precise measurement of light intensity. Our calculations show that the actual resistance range under typical operating conditions is 5kΩ - 80kΩ, which confirms that the photoresistor will function effectively within the specified parameters.
For the color sensor, we need to detect various leaf colors that indicate plant health and potential nutrient deficiencies. The sensor's spectral sensitivity covers the visible light range of 380-780 nm, which allows it to detect yellow, purple, and brown colors associated with deficiencies in nitrogen, phosphorus, and potassium. The chosen accuracy for RGB detection is ±5%, and our calculations indicate an actual accuracy of ±3%, ensuring reliable monitoring of plant health.
The power supply for the LED strip and sensors is standardized at 5 volts, which is consistent across all components to simplify power management and ensure compatibility. This voltage level is adequate to operate the LEDs and sensors efficiently, as confirmed by our calculations.

3.9.5: Photoresistor and RGB LED Demo

In this section, we detail the process and materials used to create a demonstration involving a photoresistor and an RGB LED. The goal of the demo was to showcase how the RGB LED responds to changes in light intensity detected by the photoresistor. The RGB LED lights up in different colors based on the detected light conditions: high light intensity triggers a light purple color, low light intensity turns the LED off, and medium light intensity also keeps the LED off.

3.9.5.1 Materials Used

- Elegoo MEGA 2560 Board: A microcontroller board based on the ATmega2560, compatible with the Arduino IDE.
- Photoresistor Module: A light-sensitive resistor that changes resistance based on the light intensity.
- RGB LED (4-prong): An LED with separate anodes for red, green, and blue colors, and a common cathode.
- 220 Ohm Resistors (3x): Used to limit the current through each anode of the RGB LED.
- Breadboard: For assembling the circuit without soldering.
- Jumper Wires: For making connections between components and the breadboard.
- USB Cable: For connecting the Elegoo MEGA 2560 board to a computer for programming and power.

3.9.5.2 Circuit Assembly

1. Power Rail Setup:

- Connected the 5V pin on the MEGA 2560 to the positive rail (red rail) on the breadboard.
- Connected one GND pin from the MEGA 2560 to the negative rail (blue rail) on the breadboard.

2. Photoresistor Module Setup:

- Connected the middle pin (Power) of the photoresistor module to the positive rail (5V) on the breadboard.
- Connected the "-" pin (Ground) of the photoresistor module to the negative rail (GND) on the breadboard.
- Connected the "S" pin (Signal) of the photoresistor module to A0 on the MEGA 2560.

3. RGB LED Setup:

- Connected the common cathode (-) of the RGB LED to the ground rail on the breadboard.
- Connected the red anode (R) of the RGB LED to digital pin 9 on the MEGA 2560 through a 220-ohm resistor.
- Connected the green anode (G) of the RGB LED to digital pin 10 on the MEGA 2560 through a 220-ohm resistor.
- Connected the blue anode (B) of the RGB LED to digital pin 11 on the MEGA 2560 through a 220-ohm resistor.

3.9.5.3 Code Implementation

The code used to control the RGB LED based on the light intensity detected by the photoresistor can be found here

3.9.5.4 Testing and Results

The setup was tested by shining a green light on the photoresistor and covering it to simulate different lighting conditions. The photoresistor values were monitored using the Serial Monitor in the Arduino IDE. The RGB LED responded as expected based on the detected light intensity:

- High Light Intensity (e.g., bright light): The RGB LED turned light purple.
- Low Light Intensity (e.g., covered photoresistor): The RGB LED turned off.
- Medium Light Intensity: The RGB LED remained off.

3.9.5.5 Results

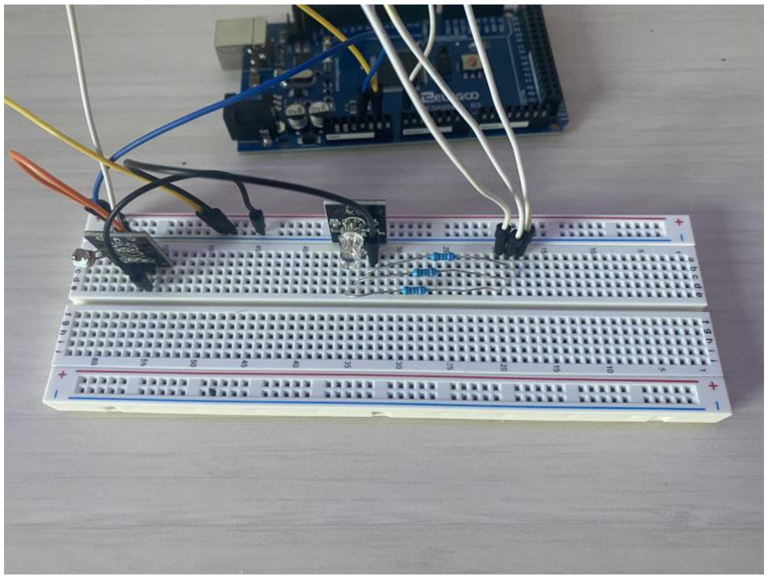


Figure 3.1. LED Results 1

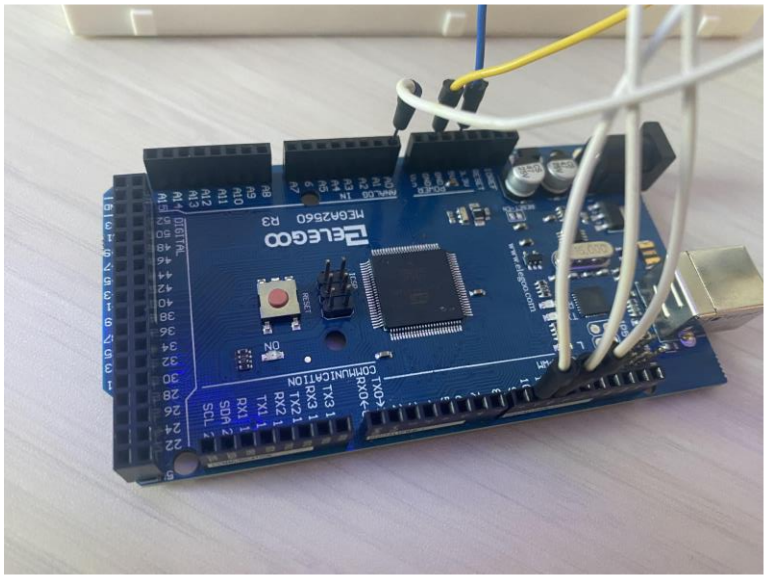


Figure 3.2. LED Results 2

This demo effectively showcased how a photoresistor can be used to detect changes in light intensity and control an RGB LED based on these changes. By adjusting the thresholds in the code, the behavior of the RGB LED can be fine-tuned to respond to various lighting conditions, making it a versatile component for projects involving light sensing and visual feedback.

3.9.6 Second Demo: Color Sensor and LED Strip Integration

The second demonstration showcases the integration of a color sensor and an LED strip, which adjusts its color based on the color detected by the sensor. This demo aims to illustrate how the our system can monitor plant health through color detection and adjust lighting conditions accordingly. By detecting green or brown paper, representing healthy or unhealthy leaves, the system changes the LED color to red or blue, respectively.

3.9.6.1 Components

- Color Sensor (TCS34725): Detects the color of the object placed in front of it.
- LED Strip: Capable of emitting different colors based on input signals.
- Arduino Board: Serves as the central controller for processing sensor data and controlling the LED strip.
- Green and Brown Paper: Used to simulate healthy and unhealthy plant leaves.

3.9.6.2 Setup and Circuit Diagram

- Connections:
    - Color Sensor:
        - VCC to 3.3V/5V on Arduino
        - GND to GND on Arduino
        - SDA to A4 on Arduino
        - SCL to A5 on Arduino
    - LED Strip:
        - VCC to 5V on Arduino
        - GND to GND on Arduino
        - Data In to a PWM-capable digital pin on Arduino (e.g., D6)
- Circuit Diagram:
    - Color Sensor (TCS34725)
        - VCC --------> 3.3V/5V (Arduino)
        - GND --------> GND (Arduino)
        - SDA --------> A4 (Arduino)
        - SCL --------> A5 (Arduino)
    - LED Strip
        - VCC --------> 5V (Arduino)
        - GND --------> GND (Arduino)
        - Data In -----> D6 (Arduino)

3.9.6.3 Code Sample

The code used to control the color sensor and LED strip can be found here.

3.9.6.4 Explanation

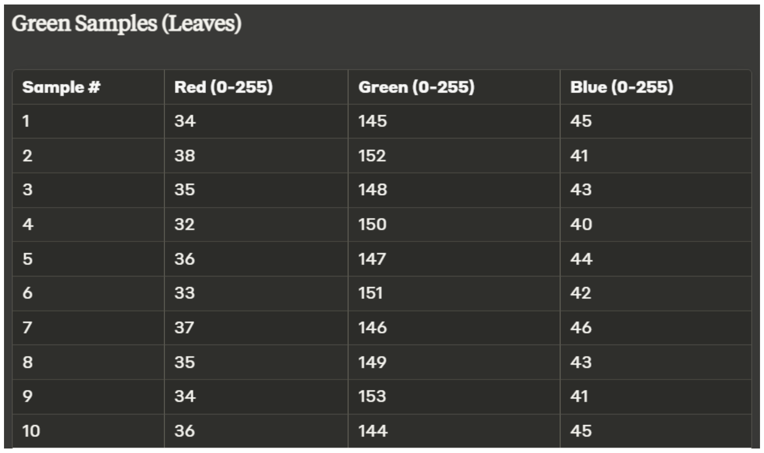
- Color Detection: The color sensor reads the raw red, green, blue, and clear light levels from the object placed in front of it. These values are used to determine the color of the object.
- Condition-Based Lighting: The Arduino processes the sensor data to identify if the object is green or brown. Based on the detected color, the LED strip adjusts its color to blue (for green paper) or red (for brown paper).
- User Feedback: The LED color change provides immediate visual feedback, simulating how the B.A.S.I.L. system can alert users to the health status of their plants through color-coded signals.

3.9.6.5 Takeaway

This demonstration illustrates the practical application of a color sensor and LED strip in monitoring and responding to plant health indicators. By detecting the color of the leaves and adjusting the LED lighting accordingly, our system can provide real-time feedback and create optimal growing conditions, ensuring healthy and productive plant growth. This integration exemplifies the system's ability to combine sensor data with automated responses to maintain an ideal environment for indoor herb cultivation.

3.9.8: Demo Readings

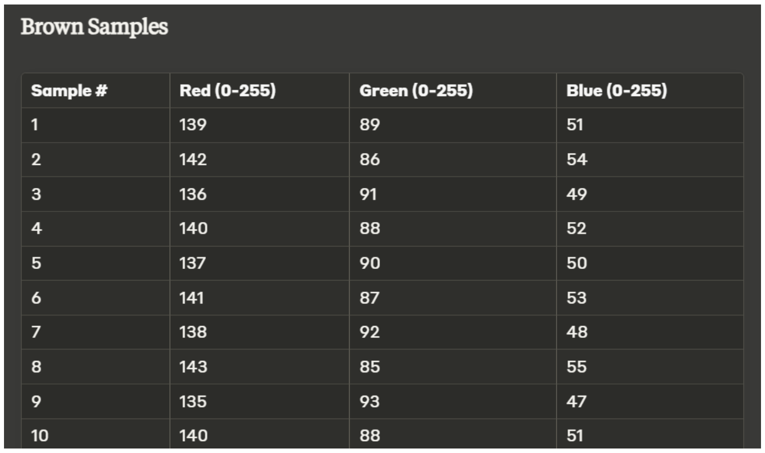
For the demo, we experimented with the three main colors that a basil plant’s leaves typically turn, green, brown and yellow. In order to read more accurately, we turned off the grid led to ensure the color sensor’s white light was the only light used. In addition, we made the break-out enclosure to the side of the plant so the leaves were still in range. All colors were detected successfully, the following tables show the results. Statistically, the colors fell within the desired goal of +- 5% FSR.



Green Samples (Leaves)

Sample #	Red (0-255)	Green (0-255)	Blue (0-255)
1	34	145	45
2	38	152	41
3	35	148	43
4	32	150	40
5	36	147	44
6	33	151	42
7	37	146	46
8	35	149	43
9	34	153	41
10	36	144	45

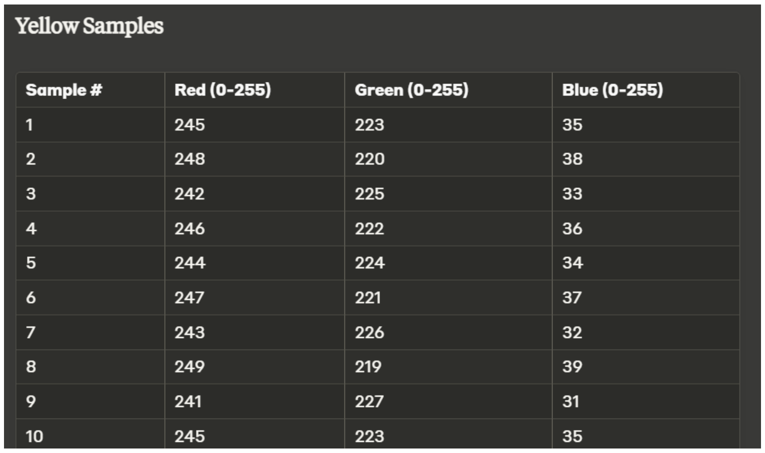
Figure 3.3 Color Sensor Results 1



Brown Samples

Sample #	Red (0-255)	Green (0-255)	Blue (0-255)
1	139	89	51
2	142	86	54
3	136	91	49
4	140	88	52
5	137	90	50
6	141	87	53
7	138	92	48
8	143	85	55
9	135	93	47
10	140	88	51

Figure 3.4 Color Sensor Results 2



Yellow Samples

Sample #	Red (0-255)	Green (0-255)	Blue (0-255)
1	245	223	35
2	248	220	38
3	242	225	33
4	246	222	36
5	244	224	34
6	247	221	37
7	243	226	32
8	249	219	39
9	241	227	31
10	245	223	35

Figure 3.5 Color Sensor Results 3

3.10 Software Research and Development

Software research is a critical aspect of this project, we wanted to use the best software tools to deliver the best product. The goal of the software in this project is to be lightweight and fast on its feet for deployment, as well as maintenance, this can start with building a CI/CD pipeline for code deployment. This section also covers the web application development research, though we wanted lightweight code deployment, we also wanted to use software tools that deliver clean UI and smooth integration with our CI/CD deployment options.

3.10.1 Cloud Engineering Research and Deployment

This section will cover how we choose to deploy our web application and which database option will be best for us, with the cloud being an incredibly popular option when it comes to software deployment, we've dedicated an entire section to research and choosing if we want to deploy on the cloud, and which cloud provider can help us deliver the best product possible.

3.10.1.1 Choosing how to deploy our web application

Deciding between on premises and one of many cloud storage options.
We plan on deploying our web application to the cloud for multiple reasons, while we could deploy our web application on internal on-premise servers, this seems like a costly option both financially and timely. A cloud infrastructure allows us to build a software pipeline through software repository applications such as GitHub or GitLabs. For example if we had our web application hosted on a container in Docker, we would need to have a secure server (typically linux) to access and then deploy any changes to our protected branches from GitHub and GitLabs. With building out a CI/CD pipeline through the cloud, we could automatically deploy any changes to the protected branches and have clear documentation of that pipeline process for debugging purposes.

Financially, going with an already established cloud infrastructure is much more affordable than having internal servers, for one, servers themselves can cost greater than $1000 dollars, then there is a maintenance and upgrade cost that scales with how many servers there are. However, on premise servers can provide arguably better security when it comes to data protection than cloud infrastructure options, this would depend on how advanced the security on the on premise servers is, though data stays in house and within the organization, if the on premise servers have less security than the cloud, then the cloud is again the better option. Having on premise servers also allows for fairly quick speeds, having a locally hosted web application can make the speed and response time better then pinging the cloud. You also don’t have to worry about possible resource sharing when on a local environment versus hosting on the cloud. Having on premise infrastructure over cloud also gives full control over the storage environment, however this feature does not make it worth it to build out for our project, pursuing this option would easily put our project over budget, which is something we are trying to avoid.

We ultimately have decided to pursue a cloud deployment option for our project. The cloud provides great versatility in terms of web deployment, the scalability that the cloud can offer is much easier than deploying on internal servers. Many cloud options also have a free tier that we can potentially leverage for the duration of our project, however we have allocated $10.00 per month for cloud infrastructure subscriptions depending on web traffic the project gets. With initial spin up cost being at an estimated $1000 dollars for buying and maintaining internal servers, along with maintenance costs that will most likely factor in, with limited scaling and possible security issues, we saw cloud deployment as a perfect substitute to what we wanted to do. Most likely, any of the cloud services we select will have better established data security and give us the financial flexibility to allocate budget in other areas instead. We can also build out a seamless CI/CD pipeline directly into our code repositories, and once triggered, can automatically deploy any changes to our web application that are well documented during the entire pipeline process. Overall the cloud better fits our needs more than the internal servers, with lower start up cost, lower maintenance cost, and higher levels of security and scalability and seamless integration without code repository, we feel confident in moving forward with cloud development.

Table 3.15. Cloud and Physical Server comparison

Cloud	Physical Servers (On-site server farm)
Shared resources	High spin up cost
Possible security issues	High maintenance cost
Slower speeds	Limited scalability
High scalability	Complete control
Low spin up cost	Fast speeds
Low price but confusing pricing options from most services	Limited resource sharing
Great integration with CI/CD pipelines	Possible security issues

3.10.1.2 Choosing a cloud environment

There is no shortage of cloud environments to choose from when it comes to deploying a web application, overall, there are four main options we have decided to consider pursuing. Amazon Web Service (AWS), Microsoft Azure (Azure), Google Cloud (GCP), and DigitalOcean. DigitalOcean is a cloud infrastructure service that supports multi-stage CI/CD pipelines and has seamless integration and hosting with multiple database clients.

DigitalOcean is a great tool for small projects such as ours, however when being implemented by an entire IT infrastructure, there may be better options. It provides a very affordable option for small projects and developers for deploying applications, it also has seamless scalability options and allows the administrator to resize the virtual machines easily. However, DigitalOcean is lacking in services compared to the big 3 in cloud development (AWS, GCP, Azure), though it allows seamless integration with multiple database clients, they are also limited in the total number of databases they support. We may want to pursue an MSSQL database instance in our project, and Digital Ocean does not support that database, limited to MongoDB, MySQL, Postgres, and redis. Though DigitalOcean supports scalability, for an enterprise scale applications the scalability might reach its limits, DigitalOcean uses droplets which come with set CPU, Memory and Storage, which is nice for simplicity and you can change the size of the droplets as well, but once scalability reaches a certain point, droplets reach a limitation that the big 3 cloud companies can exceed. For the remaining cloud deployment options (AWS, Azure, GCP), all 3 are great options and offer similar products however they do have their differences.

GCP offers Compute Engine for virtual machines and Cloud Functions for serverless computing, it also offers some Machine Learning options (AI Platform). GCP also provides high end security for the cloud, ensuring any sensitive data we may have won't be compromised. Pricing with GCP is stable as well, with offers of temporary virtual machines that we can possibly leverage for this project as a very affordable option, and if needed, upgrade to a permanent instance. However, GCP offers less services than compared to AWS, though for small projects this might not be an issue, however if this project would need to expand and grow, it might be a more difficult task then compared to AWS which has more support and popularity, and offers more options.

Microsoft Azure, created by Microsoft, offers a strong option for cloud computing, with virtual machines, and container services to host web applications that are as competitive as AWS. An advantage that needs to be considered is the integration of Azure into other Microsoft applications; Azure, being a Microsoft product, has been specifically built to integrate well with its other products, such as MSSQL, Windows Server, and Office 365. With seamless integration with MSSQL, which is a possible database option for this project, Azure becomes a very possible option to pursue. However, Azure follows closely behind AWS, in services offered and support available, it's possible as well that though Azure has close integration with other Microsoft options, it doesn't integrate as well with non-Microsoft options as AWS or GCP does. If this project uses any products outside of Microsoft, Azure might not be the best option.

Table 3.16. Comparison of Cloud Deployment Methods

	Amazon Web Services (AWS)	Microsoft Azure	Digital Ocean	Google Cloud Platform (GCP)
Pricing	Expensive (Pricing depends on usage, pay as you go mode, this project estimate $10-$20 per month)	Moderate ($9 per month)	Cheap ($5 per month)	Moderate ($12 per month)
Support	Layers (24/7 support, Dedicated technician)	Layers (24/7 support, dedicated support team)	Layers (24/7 support, 1 hour response for teams)	Layers (24/7 support, technician, rapid response)

AWS will be our best option for deploying a database for this project, AWS is the biggest producer for cloud based deployment options. It has the most options available, from which this project is projected to use the Amazon Lightsail service, which offers a cloud server and great out of the box features and predictable pricing. Though some of the services might be limited compared to other AWS services, it offers affordability and standard storage for our website. Choosing Lightsail circumnavigates around AWS' complex pricing system, with standard and predictable pricing, it takes a lot of stress off of price management, which is a known issue with AWS. It also offers the full range of support that the entirety of Amazon offers, and if we wish to scale and expand we can easily move into AWS' more complex range of options (EC2 instance). We will use Digital Ocean to deploy our web application to the cloud, Digital Ocean provides a direct plug in with GitHub to deploy our code and an easy to use web interface to interact with our deployment. It has an automated CI/CD pipeline that runs with each push to the main branch and makes it very easy to moniter the deploymeny process as well.

3.10.2 Database

For this project, we'll be using a database to house our MCU, user and measurement data. In order to be able to do this, we'll need to choose a database provider that is both supported by the cloud and can properly store the data in the way we want to store it.

3.10.2.1 Choosing a Database

There are plenty of database options available for this project, the main two in consideration are relational databases and non-relational databases. Relational databases typically use the SQL query language, however some differences do exist within the SQL languages for example MySQL and MSSQL have slightly different ways of querying, but most of the language is the same.

A difference between relational and non-relational databases is how the data is stored, in relational databases data is stored in a row column format, this is what is called a relation. With non-relational databases data is stored without strict row column format, this allows more creativity when it comes to data storage, however this increases the need for data pipelines when moving data from one database to another schema database. For example MongoDB (a non-relational DB) uses Document storage, and Redis (a non-relational DB) uses Key Value storage, since it is not a united front when it comes to data storage, exporting and moving data can become more and more complex when it comes to non-relational databases. With relational databases it can be much easier to build a pipeline for data between two different systems because the groundwork is all the same. However if you do happen to have a variety of data, then non-relational databases can be an option because it provides that very same data flexibility that causes issues with pipelines. SQL databases are rigid but this makes pipelining data easier to do, if data is static, then a SQL database is a great choice because the data doesn't have to be confined in a relational way, it already is a simple static source. Since SQL databases are strict when it comes to data sourcing, it allows data integrity as well, which will be important with our data pipeline because moving data from MCU to Database to Web Application could cause unintentional changes in data, SQL can help preserve the data integrity between the MCU and the Web Application by enforcing the data to follow the pre-existing set of conditions (primary keys, foreign keys, constraints).

Table 3.17. Comparison of SQL and noSQL Database

Relational Database (SQL)	Non-Relational Database (noSQL)
Row and column structure (relational)	Multiple structures (document, key and value, graph)
Easy to learn	Can be difficult to learn because of multiple data structures used in storage of data
Extremely structured, resulting in high data integrity	Flexible and scalable
Huge community	Less data integrity resulted from high flexibility and scalability
Best for static data and complex transactions	Cost effective

When it comes to choosing a database, we must return to which cloud deployment option was chosen. The plan is to use AWS to deploy our web application with an AWS deployed and supported database, as of time of writing AWS LightSail offers two fully supported databases, PostgreSQL and MySQL, though non-relational databases are gaining more and more popularity, we are limited to two options of relational databases. Other cloud options come with more supported options, for example Azure supports MSSQL fully. PostgreSQL is one of the supported options with AWS LightSail, with using PostgreSQL it offers advanced options when it comes to supporting data, more so than MySQL, these advanced options need to be considered when it comes to what our project will need from the database. MySQL offers a lighter and faster experience when it comes to PostgreSQL, but it also lacks advanced capabilities. When researching, we found that PostgreSQL is extremely good for complex projects that have complex queries, and is great for enterprise level projects. However, it lacks in lightweightness and speed when it comes to read-write operations, and with the requirements of our project we prefer speed and agility with data rather than having the more advanced query abilities.

The requirements of this project we've set out is to have a speedy response when it comes to updating threshold values and the communication between our web application and its physical counterpart, therefore we value speed much more than actual database capabilities, we're looking at doing very basic CRUD operations for this database and won't need extensive data integrity, we don't plan on any views, triggers, or stored procedures, and though MySQL supports these, they aren't as great at it as PostgreSQL, but since we aren't concerned about that, then

MySQL is the better option for this project. The next consideration would be pricing, but for AWS LightSail, PostgreSQL and MySQL have identical prices and therefore we decided not to factor that into our decision on which database provider to choose.

3.10.2.2 Creating the Database

The first step in creating the database is to actually decide on the pricing and performance of it. As discussed in the last section, this project will use a MySQL database instance using AWS LightSail deployment tools; there are two main options with four sub options for each, instead of the high availability database option which effectively doubles the prices of the standard database, for this project a standard plan will work fine. There are now four options for a MySQL deployment, there is a fifteen dollars a month plan, thirty dollars a month plan, sixty dollars a month, and one hundred and fifteen dollars a month plan, since our project is just using a very basic database plan, we've decided to use the fifteen dollars a month basic plan, this provides 1GB of RAM, 2vCPU processing, 40 GB SSD, 100 GB transfer data. This is a basic plan that allows us to use AWS LightSail for fifteen dollars a month with the first ninety days free, with an easy to use UI and scalability with AWS, we can easily adjust these values if needed in the future.

3.10.2.3 Database Design Research

The database design for this project is going to be simple, the goal is to keep the database as lightweight as possible to make it easy for the MCU to export any of the sensor's data to the same database and in best case scenario the same table. The timeline for updates to the database will be leveraged from the MCU, the database is just holding data for the web application, no triggers or stored procedures will be used. We want the database to be a short but efficient data pipeline to and from the MCU and the Web Application UI. In the database, there will be a column that specifies the users MCU, for now we will use only one MCU which allows simplicity on the database side to just assume all measurements and data in the table is from one MCU. Whether the MCU will insert a new value into the table or update existing values will be determined, the value of keeping all data is good for possible data analysis after the project is finished to potentially enhance how we adjust values, or we can replace the column values without inserting new rows into the table, this saves money and also seems like a more integrated way to allow for more MCUs (meaning more users). Each column will also be a sensor value, and we'll leverage those values in the web application, some columns will hold active values for our user manipulated pieces of our project as well (EMF Generator, Light sensor LED, and artificial sunlight values). With that, we can track the values the user has updated and export those values to the MCU which will then adjust the threshold values for the sensors, this will most likely happen in our web application through potentially Bluetooth or Wi-Fi, this is to be determined.

3.10.3 Web Application Research

When developing this web application, our goal was to use frameworks and libraries to deliver a great product, but also be easy on us as developers as well. Main contributions to our decisions on what to use include previous experience, time to deploy, ease of access, learning curve, and data integrity.

3.10.3.1 Web Application Design Research

One of the main software components of the project is the web application. The web application will be the main way a customer will be able to see the data that the MCU and sensors relay to the database. The customer will be able to log in securely and safely, able to view the data from the plant repository and adjust certain values to relay back to the MCU for adjustments. The idea is to give the customer a one stop shop for all of their plant needs, to provide OpenAI integrated feedback of the data ingested by the MCU and provide advice on how to possibly give better care to their plant and overall become a better plant carer.

We want to provide a single page application that supplies the tools and empowers the customer to make the plant care process as simple as possible. This goal can be achieved by accurate data analysis and suggestions, another contribution is a scientifically simple UI, this may be as important as the AI suggestions, the ability for the customer to quickly and effectively digest information given to them through a seamless and accurate UI will predictably make a huge difference in the user experience. The Web Application can be broken down into 3 major parts that can be attributed to a full stack application, examples such as LAMP, MEAN, MERN stack applications can all be broken down into 3 components, being the Database, Frontend and Backend design.

Firstly, the application will have a frontend code component, this will contain the UI code for the user and will allow the user to input change values, at this level, there will be basic input cleansing and preventative measures. Second, the application will have a backend component, this component will do all of the heavy lifting when it comes to the ML/AI data analysis, with an OpenAI public API we'll communicate with ChatGPT to do data analysis upon the users plant data for possible suggestions and comments, this is also where all of the projects API components will be, there will be heavy input cleansing here to ensure proper cyber security and to prevent any possible cyber threats. The third component will be the database, which will be storing all data shown to the user and updated by the user, this is the last layer before being sent to the MCU for updates.

3.10.3.2 Backend Design Research

When designing a backed for a web application, there are limitless options ranging from frameworks that support JavaScript, Python, Java, C#, Ruby, PHP, and countless other programming languages. The objective of the backend in web applications is to be the communicator between the frontend / UI, to the Database and everything in between. The backend is where all of the API endpoints will be housed and where most of the data cleansing will be taking place for cyber security purposes. Our main goal for this project's backend is going to be data retrieval, data analytics through OpenAIs ChatGPT and data security.

- **PHP**

  When looking into researching types of backends there is a long list, PHP is an old and proven backend that works well with Apache and Nginx. It integrates well with HTML and is an open source language, it also integrates smoothly with databases both SQL (MySQL, MsSQL, PostgreSQL) and noSQL (MongoDB, Redis). It also supports the basic C.R.U.D operations easily, and has easy hashing functions for passwords. Though PHP is an old and true option, it also struggles with staying relevant and it has drawbacks, it is not as versatile as other options and though it is easy to integrate with frontend and databases, it does have a steep learning curve for new developers, which could make its use for this project difficult.

- **Ruby on Rails**

  Ruby on Rails (or Ruby), its learning curve is not nearly as steep as some of the other options for backend development, and its spin up time is actually quite quick, which can make it an exceptional option for our project. Ruby on Rails uses the language Ruby which has a similar syntax to Python or Perl, which are high level and easy to learn languages, which makes this another reason to possibly use this backend structure. It has a big community and is supported by its network which is great for scaling, it has built in functions for C.R.U.D operations like PHP. Ruby on Rails uses REST APIs by default. Ruby is a very interesting option because it excels in basic tasks. This web application wants to be extremely lightweight on the backend to provide quickness and lightweightness to our frontend. Though it excels in lightweight operations, it does have a major drawback for this project, it is extremely slow compared to other backend structures, but considering that our project has a basic design, this might not be a major set back.

- **.NET**

  A very popular C# backend structure is .NET (Dotnet), which is Microsoft’s C# backend. .NET is a fully supported development platform for developers to create vast API networks or basic applications through the use of .NET and C#. C# is very similar to Java with some differences.

  A .NET backend has great performance speeds and if the developer has Java experience, the learning curve isn’t nearly as steep. It supports OOP development, like Java, .NET supports multi-platform development, across Windows, Linux and MacOS. With .NET comes a big community of support, with a company like Microsoft constantly creating and maintaining it, this is a reliable option that allows scaling and stability for a backend infrastructure. .NET also supports multiple core frameworks that enhance the development experience, an example of one framework is the Entity Core Framework, this allows the automatic creation of C.R.U.D operation APIs to the application, once connected to the database, this allows for an almost low-code or even no-code situation for developers, though these operations only extend to very basic API calls, any more scaling for specific API needs would need to be done through programming. With all these tools from .NET to leverage, this also creates a steep learning curve, especially with an intense infrastructure.

There also comes the cost of licensing when it comes to rights from Microsoft, it can get pricey, which with a small budget project, could become an issue with scaling. The time it might take to create an infrastructure that supports all of these tools might also be a time crunch, though for the long run it might provide great support and growth, the time it takes to spin up might not be a viable option.

- **Apache Tomcat**

  When creating a Java backend, there are many backend infrastructure options, for this project, Apache Tomcat is being considered as the main option. One of Java's biggest strengths is its classic Write Once, Run Anywhere JVM design. Java Virtual Machine (JVM) allows a Java application to be run on any system with a JVM, this multi platform design could provide flexibility for our project.

  Java, being one of the most popular languages on the planet, also has a great support system and community, being open source also provides even greater support through multiple companies such as Oracle (being the actual maintainer of Java), Amazon and IBM. Being a high profile language, similar to C# and .NET, supports an in depth system of tools, like multithreading, garbage collection, and many libraries to leverage. Apache with Java allows for great resource management, it should be able to handle any amount of traffic that this project might recieve, also Apache Tomcat provides great security features through HTTP calls, and has advanced scalability with load balancing options. However, Java with Apache Tomcat runs into a similar issue that .NET did, though once set up, it is an amazing backend resource, the time crunch it would take to spin up an infrastructure like this might be too great for a project with this timeline. With a pretty steep learning curve and in depth infrastructure, we might look for a more easy to learn and spin up option that might not support as much as Java or .NET does. This shouldn't be an issue since our project looks to be lightweight and small, to be as effective as possible with doing as little as possible, using Java with Tomcat might be overkill for what this project needs are.

- **Flask**

Python offers many frameworks for web development, the one considered for this project is Flask. Python, known for having a small learning curve and a great starting programming language, also carries over the simplicity design for Flask; its spin up speed is exceptional. Flask being easy to learn and use makes this a great option for this project, however we must think of scaling, Flask does struggle quite a bit on the ability to scale, particularly with larger projects, multiple database connections and complex APIs. It becomes a bit more granular when it comes to development with all of these configurations, Flask requires a lot of manual work to set up these configurations which could become time consuming. Additionally, Flask does not provide a great infrastructure for out of the box security, many responsibilities fall upon the developer to bear that responsiblity. Python has great options for spinning up a small project like ours but the scalability and security issues are of concern.

Table 3.18. Comparing Backend Frameworks

	Python (Flask)	JavaScript (Express)	.NET	Java (Apache)	Ruby	PHP
Development Time	0-2 days	0-2 days	2-5 days	2-5 days	0-2 days	1-3 days
Learning Curve	0-3 month s	0-3 months	3-6 months	3-6 months	1-4 months	3-6 months

- **ExpressJS**

  JavaScript provides a long list of backend structures, for this project we are considering the backend platform Express. Express is a backend structure built on top of NodeJS, NodeJS is a cross platform runtime environment that allows an application to run on MacOS, Windows and Linux, and allows web applications to be deployed. Express runs on top of a NodeJS environment, it is a very diverse and scalable option for this project. It is a minimalist approach to a backend structure which follows our design pattern of a lightweight backend, it is also very free flowing, and allows the user to have high level control over the backend. Though this can bring its own issues with inexperienced developers, the high level of control can cause mistakes and steep learning curves. Its routing makes creating RESTful APIs very simple and light, and connections to databases come very easy, it also provides a high level of performance, being able to keep up with frameworks of lower level languages fairly well.

3.10.3.3 Frontend Design Research

Frontend design is an important concept for this project, it will be the UI and include any customer facing code logic, the goal is to keep the frontend design lightweight and intuitive for the user. When creating a frontend, the very first component we want to think about is the HTML document, the idea behind the HyperText Markup Language (HTML) is to create a structured environment for web content, typically a web application, allowing data to be presented in an organized way. Though HTML excels in structure, it does lack in aesthetics, many struggle to read through white page data for long periods of time, that’s where CSS (Cascading Style Sheets) comes in. Many see HTML and CSS working hand in hand to provide a beautiful way to present data to users in the most structured way possible. CSS manipulates the way an HTML document looks, not the actual data within the HTML document, this lets developers create and design and present the data in a pleasing manner. Studies show that almost 50% of customers say that web design is the number one factor for judging the credibility of a company [36]. This shows not only how important the data within the web application is, but *how* it’s presented. Now with a web application, the data and how it’s presented is incredibly important, introducing JavaScript allows the manipulation of the data within the HTML, if your web application wants to be able to be dynamic and change with user activity, then JavaScript will need to be applied behind the scenes. JavaScript allows a web application to take the next step of taking a good application to a great application, we can leverage JavaScript on frontend design by using a framework or library that allows for smooth integration of JavaScript within HTML and CSS (or SASS). There are many options to picking a front end design framework or library.

- **AngularJS**

  AngularJS (Angular) is a web application framework designed and maintained by Google. It revolves around creating a single page application, where to the user, could potentially look like multiple pages, in reality are all within a single HTML document.

  Angular follows a component structure, which means within the application, it can be broken up into smaller pieces called “components”, using components allows the application to show specific components to the user, without having to show everything within the app.js file. Component structure is broken into 3 pieces, the template file (HTML), the component file (TS) and the design file (SASS or CSS). The component file uses a superset of JavaScript called TypeScript (TS), TypeScript is a strongly typed programming language, where JavaScript is not, which puts more responsibility upon the developer, TypeScript follows a stricter set of rules which may seem annoying but in reality help avoid pitfalls later in the development period of the project where mistakes that would happen with JavaScript was caught much earlier on in TypeScript. All 3 of these files make up one single component, which allows design and behavior to stay within the component. A group of these similar components can share a service file (TS), which usually contains utility functions and HTTP calls to gather data from the backend.

A service file will usually contain variables that need to be “shared” between two components, for example if a user selects an item to purchase, that item will be added to a “cart” variable in a service file, and in the checkout component, the component will leverage that “cart” variable in the service file to know what the user selected on previous “pages” (user thinks its another page) in reality it’s within the same HTML, this is called dependency injection. This single page application design allows extremely high performance and speed, since all data was rendered on page load, it no longer needs to load any data since it's all been loaded already.

Angular integrates with HTML by using two way data binding. Two way data binding allows “communication” between both the component file (TS), and the template file (HTML). By communication, meaning that there is a direct line of change between both files, if the user changes a variable by user input, then the component file automatically listens to that change, this works both ways, if the component file changes a variable value that change can be seen in the template file, this allows a streamlined way of communication with user input and changes that could possibly change within the page.

Angular also allows something called directives, this allows for snippets of code to be placed into the HTML, for example Angular allows for *ngIf statements and *ngFor statements, which works the same way as an “if” and “for” statements. These directives allow another level of data manipulation in the template file, another kind of directive from Angular is it allows variables to be placed within the HTML, which is denoted by “{{ variable }}”, though Angular allows some JavaScript injections, it does not allow extensive scripts within the HTML document, it prefers as much JavaScript work within the component file than the template file. Angular also has an integrated CLI, which helps streamline application development by allowing the developer to write scripts to automatically build out components and services that are automatically added to definition files and module files, this allows the developer to focus on web application development for the user rather than administration set up.

- **ReactJS**

  React is a JavaScript Library that is built by Facebook, it is used for single page applications, but is different from Angular because it does not follow a strict set of rules to creating an application, which can be helpful for a more free flowing development set. React also uses something called Components but this is not the same as Angular's component structure, components are small and reusable chunks of code that allows for an object oriented way of developing a web application. React fully embraces integrating HTML and JavaScript, it breaks apart the HTML by static and dynamic values, and with the dynamic values of the HTML will then be manipulated and returned from functions and lifecycles in the JavaScript, this is called the Virtual DOM. Dynamic changes are managed by lifecycle hooks and component states. Since React interacts directly with HTML, it leverages the JSX syntax extension, which helps error handling within HTML and JS and is a bit stricter, similar purpose as TypeScript is in Angular.

- **VueJS**

  Vue is a framework same as Angular, it also follows component structure, in a single file component structure with a .vue extension. Vue uses JavaScript language for development, it has parent components and child components with a similar structure to Angular, which helps support single page applications, it also has a similar structure to directives, which allows injections of JavaScript code like "if" and "for" inside of the HTML. It also has data binding so you can see the reflection of data changes due to internal changes or user input. Vue uses routing like angular to route from component to component based upon what the user is clicking and based on feedback from them. Vue does encapsulate HTML, CSS and JavaScript all within the same file which can help developers better visualize how their app can communicate between the template and JS sections of the files. Since Vue is a single page component, it also supports JSX same as React, which helps developers by keeping a stricter set of rules within JavaScript. Vue also has an integrated CLI to develop components quickly by scripting which is very similar to Angular.

Table 3.19. Comparing Frontend Frameworks and Libraries

	Angular	**React**	**Vue**
Learning Curve	2-6 months	0-4 months	0-4 months
Development Time	0-2 days	0-2 days	0-2 days

3.11 Hydroponic Systems

Upon researching similar products to what we wanted to construct we noticed that most of them used hydroponics instead of soil which we were going to use originally. We assumed soil would be the straightforward and easy solution but were surprised to find out the benefits of ditching soil.

One of our main goals with this project is to enhance the cultivation of herbs by accelerating their growth thus increasing the yield. Hydroponics has been proven to do just that. In a hydroponics system nutrients and water are always being circulated into the plants roots which allows for easier absorption. Normally plants have to search for nutrients and rely on consistent watering in order to flourish. By using hydroponics we are consistently helping the plant achieve its necessary threshold of nutrients and water which has been proven to help the plants mature faster and produce a higher yield.

Hydroponics also eliminates a lot of the inefficient use of water and nutrients that usually occurs with natural planting in soil. Even though the roots are submerged in water because the water is filtered and recirculated, it actually tends to use about 90% less water than conventional agriculture. Also, we are able to integrate and measure the exact amount of required nutrients directly into the water supply which allows for more efficient usage of said nutrients thus eliminating more waste that normally occurs.

Another main goal of our project was to create more opportunities for consumers to enjoy organic herbs that haven’t been modified or been sprayed with pesticides. Due to the nature of hydroponics, the absence of soil prevents most of the common pests and pathogens that typically plague plants grown in soil. This produces herbs that have no chemical residue and are healthier than soil grown herbs.

The obvious downside to hydroponics is that it is more complicated to set up and can cost more money upfront to get going. Despite this, we believe the benefits outweigh these problems because it helps us achieve our goals more efficiently.

3.11.1 Nutrient Film Technique

The Nutrient Film Technique, or NFT, is one of the most commonly used hydroponic systems in which a thin nutrient solution constantly flows over the plant's root zone. The roots are only covered by the nutrient solution, so they are well supplied with nutrients and oxygen. NFT systems have proven effective regarding water and nutrient usage, resulting in less expenses and more sustainable systems. It helps grow healthy plants due to the constant supply of nutrients and can be easily adjusted to different plants. However, NFT systems have a few disadvantages such as components prone to pump failures and the constant need for monitoring to avoid the roots from drying out.

3.11.2 Deep Water Culture

DWC is a type of hydroponics where plants are floating in a nutrient-filled solution and their roots are directly immersed. This system is easy to implement and manage and is suitable for what we want to accomplish. An issue that most people have when using DWC systems is that you need to monitor the temperature and the pH level of the water. For our project, this is not a disadvantage as we already intended on monitoring these factors and because of this DWC is a viable option for us to utilize.

3.11.3 Drip System

The drip system entails taking water containing nutrients to plants at their root by use of tubes and drip emitters. This method enables the plant nutrients and water to be dispensed in the exact proportions, which minimizes waste while ensuring that the plant gets the right amount of water. However, drip systems can be a hassle as they need regular maintenance to be done on the emitter to prevent clogging and make sure the nutrients are being dispersed evenly.

3.11.4 EBB and Flow

Ebb and Flow systems operate by flooding the plant’s roots with water that contains nutrients, and then allowing the water solution to drain back into a holding tank. This cycle provides the roots with the required nutrients and simultaneously delivers oxygen. However, much like the drip system, this requires regular maintenance to ensure the pump is working correctly and not overflooding. We want our product to be as self sufficient as possible so this fact is a concern for our project.

3.11.5 Aeroponics

Aeroponics is a technique that requires the suspension of the plant roots with an arm in the air and the spraying of nutrient solution over them. This method ensures that high oxygen concentrations reach the roots to spur growth and production in the plant. Aeroponics systems are very efficient with the use of water and nutrients, and they also produce plants faster than other methods. However, they are challenging to design and manage, especially regarding the operational parameters of the misting regimen and the nutrient solution. Aeroponics is best used on a larger scale than what we are implementing. Due to the difficulty of maintenance and the high price, our project is better off not implementing this method of hydroponics.

Chapter 4 - Standards and Design Constraints

In the development of our project, adhering to recognized standards and understanding the design constraints are crucial to ensure the success and reliability of our system. Standards provide a framework of best practices and guidelines that enhance the safety, performance, and interoperability of our system. They help us align with industry norms, comply with regulatory requirements, and meet the expectations of our stakeholders.

Constraints define the boundaries within which we must operate, considering factors such as material limitations, cost, time, and environmental impact. Addressing these constraints from the outset allows us to devise practical and efficient solutions, mitigate risks, and optimize the design process. In this section, we explore the key standards that have guided our design decisions and discuss the constraints that have influenced the development of our project. By integrating these standards and constraints, we aim to deliver a robust, reliable, and safe hydroponics system that meets the highest quality benchmarks.

4.1 Standards

ASTM E308 Overview
ASTM E308 standardizes color calculation using spectral data and the CIE color space. It ensures consistent and accurate color determination across industries reliant on precise color measurement.

Importance in Color Sensor Applications:
Accurate color measurement is crucial for various applications, including quality control and our horticultural lighting system. ASTM E308 guarantees reliable color sensor calibration and data for informed decision-making.

Core Components of ASTM E308:
- Spectral Data: Detailed spectral data is essential for precise color calculation.
- CIE Color Space: This mathematical model represents colors as coordinates, providing a standardized framework.
- Color Matching Functions: These simulate human eye response to different wavelengths for accurate color conversion.

Application in Our Design:
- Sensor Calibration: ASTM E308 ensures precise color sensor calibration for accurate plant leaf color measurement.
- Data Interpretation: Consistent color data interpretation enables informed lighting adjustments for optimal plant growth.
- Quality Assurance: Adherence to ASTM E308 guarantees high-quality color data for reliable system performance.

Implementation Steps:
- Spectral Measurement: Precisely measure light source and plant leaf spectra.
- Sensor Calibration: Calibrate color sensors according to ASTM E308 guidelines.
- Data Processing: Convert spectral data to colorimetric values using CIE color matching functions.
- Continuous Validation: Regularly verify system accuracy against standards.

Takeaways:
ASTM E308 is essential for precise color measurement in our system. By following its guidelines, we optimize plant growth, enhance data reliability, and maintain high-quality standards.

IEC 60950 Overview
IEC 60950, titled "Safety of Information Technology Equipment," is an international standard that addresses the safety requirements for information technology equipment and related devices. This standard is essential for ensuring that electronic equipment used in various applications is safe for users, protects against

electrical hazards, and mitigates risks associated with mechanical and thermal hazards.

Key Features

- **Electrical Safety**
  Insulation Requirements: Specifies the necessary levels of insulation to protect users from electric shock.
  Creepage and Clearance Distances: Defines the minimum distances between conductive parts to prevent electrical arcing and ensure safe operation.
  Grounding and Earthing: Provides guidelines for proper grounding to prevent electrical shock and ensure safe discharge of electrical currents.
- **Mechanical Safety**
  Enclosure Design: Sets requirements for the mechanical strength and stability of enclosures to protect internal components and users.
  Protection Against Mechanical Hazards: Ensures that equipment design prevents users from accessing moving parts and other mechanical hazards.
- **Thermal Safety**
  Temperature Limits: Defines maximum allowable temperatures for accessible parts and internal components to prevent burns and overheating.
  Thermal Insulation: Specifies requirements for insulating materials to manage heat dissipation effectively.
- **Fire Safety**
  Flammability Requirements: Sets standards for the flammability of materials used in equipment to reduce the risk of fire.
  Fire Enclosure: Provides guidelines for designing enclosures that can contain fires originating within the equipment.
- **Protection Against Hazards**
  Overcurrent Protection: Requires the implementation of fuses and circuit breakers to protect against overcurrent conditions.
  Surge Protection: Specifies the need for protection against voltage surges and transient overvoltages.

Applications

Implementing IEC 60950 ensures that all information technology equipment, including sensors, controllers, and power supplies, meet stringent safety requirements. Key applications include:

- **Electrical Safety:** Ensuring that all PCBs and electrical connections within the system are insulated and grounded properly to protect against electric shock.
- **Mechanical Safety:** Designing robust enclosures for sensors and control units to prevent user access to hazardous parts and ensure mechanical stability.
- **Thermal Safety:** Implementing adequate thermal management solutions for LEDs and power supplies to prevent overheating and ensure safe operation.

- **Fire Safety:** Using flame-retardant materials and designing enclosures to contain potential fires, thereby protecting the entire system.
- **Hazard Protection:** Incorporating overcurrent and surge protection devices to safeguard sensitive electronic components from electrical faults.

Takeaways

Adopting IEC 60950 provides a comprehensive framework for ensuring the safety and reliability of information technology equipment. By adhering to this standard, we can mitigate risks associated with electrical, mechanical, thermal, and fire hazards, ultimately creating a safer and more reliable hydroponics system.

IEC 62471 Overview

IEC 62471, the "Photobiological Safety of Lamps and Lamp Systems" standard, safeguards human health by addressing the potential risks of light-emitting devices, such as LEDs. This international guideline evaluates photobiological hazards, ensuring products minimize adverse effects on skin and eyes.

LED Lighting and IEC 62471:

LEDs, while energy-efficient and versatile, emit potentially harmful blue light and other radiation. IEC 62471 is vital for ensuring LED lighting systems are safe for both people and plants. This is especially critical in horticulture where intense light exposure is common and workers may be at risk.

Key IEC 62471 Components:

- Hazard Classification: Lamps and systems are categorized into risk groups (Exempt to Risk Group 3) to clarify potential harm.
- Measurement Techniques: Specific methods assess light source spectral power distribution and radiance for hazard evaluation.
- Exposure Limits: Research-based limits protect against acute and long-term health risks from UV, visible, and IR radiation.

Applying IEC 62471 to Our Design:

- Plant Safety: Optimizing light intensity and spectrum while adhering to IEC 62471 safeguards plant health, preventing damage like photoinhibition.
- Human Health: Protecting users from harmful radiation through IEC 62471 compliance prevents eye strain, skin issues, and other health problems.
- Regulatory Adherence: Meeting IEC 62471 requirements facilitates product certification, market access, and legal protection.

Implementation Strategy:

- Initial Assessment: Evaluate all light-emitting components to determine their IEC 62471 risk group.
- Design Refinement: Make necessary adjustments (filters, diffusers, intensity, duration) to achieve standard compliance.
- Ongoing Monitoring: Continuously monitor and reassess the lighting system to maintain IEC 62471 adherence.

Takeaways:

Prioritizing IEC 62471 in our LED lighting system design safeguards both plants and humans. By optimizing light conditions and mitigating risks, we enhance product effectiveness, build trust, and comply with regulations.

IPC-2221 Overview

The IPC-2221 standard is a globally recognized guideline that provides comprehensive requirements for PCB design. It encompasses various aspects of design, including material selection, mechanical and electrical considerations, layout, and testing procedures. The standard is designed to ensure that PCBs are reliable, manufacturable, and meet specific performance requirements.

Comprehensive Coverage

IPC-2221 covers a wide range of design considerations, ensuring that all critical aspects of PCB design are addressed. This includes guidelines on:

- Material Selection: Ensuring that the materials used for PCBs are suitable for the intended application and environment.
- Mechanical Design: Providing recommendations for board thickness, hole sizes, and mechanical strength.
- Electrical Design: Addressing signal integrity, power distribution, and electromagnetic compatibility (EMC).
- Thermal Management: Offering strategies to manage heat dissipation and prevent overheating.
- By following IPC-2221, we can ensure that our PCBs are robust and capable of withstanding the operational demands of our intelligent hydroponics system.

Improved Reliability and Performance

Adhering to IPC-2221 helps enhance the reliability and performance of our PCBs. The standard provides guidelines to minimize potential failure points and ensure consistent performance. This includes:

- Design for Manufacturability (DFM): Ensuring that PCBs can be manufactured consistently and cost-effectively.
- Design for Testability (DFT): Facilitating efficient testing and quality control processes.
- Environmental Considerations: Addressing issues related to humidity, temperature, and other environmental factors that could impact PCB performance.

Implementing these guidelines ensures that our PCBs will function reliably over the product's lifespan, reducing the likelihood of failures and maintenance issues.

Global Acceptance and Industry Best Practices

IPC-2221 is widely accepted in the electronics industry and is considered a benchmark for PCB design. By adhering to this standard, we align our design practices with global best practices, ensuring that our product meets or exceeds industry expectations. This is particularly important for:

- Regulatory Compliance: Ensuring that our product complies with relevant safety and quality regulations.
- Market Acceptance: Increasing the likelihood that our product will be accepted and trusted by customers and industry stakeholders.

Scalability and Future-Proofing

IPC-2221 provides a scalable framework that can be adapted as our product evolves. This is crucial for future-proofing our design and accommodating potential upgrades and modifications. The standard's guidelines can be applied to various types of PCBs, from simple single-layer boards to complex multi-layer designs, allowing us to scale our product without compromising quality.

Specific Applications

In our BASIL H.E.R.B. project, the application of IPC-2221 is particularly relevant in several areas:

- Sensor Integration: Ensuring that the PCBs housing our light, color, and other sensors are designed for optimal performance and reliability.
- LED Control Systems: Designing PCBs that manage LED power and control, ensuring efficient heat dissipation and electrical performance.
- Data Processing Units: Implementing robust PCBs for microcontrollers and data processing units, ensuring reliable operation and data integrity.

Takeaways

The decision to implement IPC-2221 in our PCB design process is a strategic choice aimed at ensuring the highest standards of quality, reliability, and performance for our project. By adhering to this globally recognized standard, we can confidently develop a product that meets industry best practices, complies with regulatory requirements, and satisfies our customers' expectations. IPC-2221 provides a solid foundation for our PCB design, enabling us to build a robust and scalable system that will support the successful cultivation of organic herbs in an intelligent and automated environment.

4.2 Design Constraints

Economic

Budgetary Limitations:

The project's budget is limited to approximately $360, necessitating careful planning and allocation of resources. This constraint affects every aspect of the design and implementation process, from component selection to assembly and testing. The primary goal is to deliver a functional and reliable system without exceeding the allocated budget.

Cost-Effective Component Selection:
To stay within budget, we must prioritize cost-effective components while ensuring they meet the necessary performance standards. This involves making strategic decisions about which components to use and where to potentially cut costs without compromising the overall quality and functionality of the system. One significant example is the decision to use a color sensor instead of a spectrometer.

Using a Color Sensor Instead of a Spectrometer:
A spectrometer, while highly accurate, is a significantly more expensive option for measuring light and color. The cost for the hardware alone can run into the thousands of dollars, and the ongoing expenses to run the software further increase the total expenditure. Although spectrometers provide detailed spectral data beneficial for precise applications, their high cost makes them impractical for our project and infeasible for our average user. Instead, we opt for a color sensor, which is significantly more cost-effective. While a color sensor may not provide the same level of detail as a spectrometer, it is sufficient for our needs, particularly when calibrated according to standards like ASTM E308. By using a color sensor, we achieve a balance between cost and functionality, ensuring accurate color measurement and monitoring without exceeding our budget.

Balancing Cost and Quality:
While opting for cheaper components can initially reduce costs, this approach often introduces reliability issues or requires more frequent replacements, potentially leading to higher long-term expenses and maintenance headaches. Conversely, choosing more expensive components may offer superior performance and durability, but can quickly deplete the allocated budget, limiting the scope and capabilities of the overall design. Therefore, it is essential to find components that strike the best performance-to-cost ratio, ensuring both immediate functionality and long-term reliability without exceeding financial constraints. Our approach involves thorough research and comparison of available components to identify those that meet our project's specific needs while providing good value. This process includes evaluating technical specifications, user reviews, and performance benchmarks to ensure that selected components can deliver the required performance within budget. For instance, in the selection of LEDs, we prioritize mid-range options that offer sufficient light intensity and spectrum suitable for optimal plant growth. These LEDs are chosen because they provide a balance of efficiency and cost-effectiveness, avoiding the exorbitant prices of high-end models while ensuring the quality and reliability needed for our application.

Bulk Purchasing and Supplier Negotiations:
Another strategy to manage economic constraints is bulk purchasing and negotiating with suppliers. Buying components in larger quantities often leads to discounts, reducing the overall cost significantly. Additionally, negotiating with suppliers for better rates or bulk discounts can further stretch the budget, allowing us to allocate resources more effectively. Moreover, Texas Instruments offers the opportunity to sample sensors, which can be particularly advantageous for our

project. These samples not only provide an initial supply of high-quality components but also allow us to evaluate their performance before committing to larger purchases. Texas Instruments sensors are also very affordable compared to those available on Amazon or other online storefronts, offering a cost-effective solution without compromising on quality. By leveraging these purchasing strategies, we can maximize our budget and ensure we acquire reliable components at the best possible prices.

Maintenance and Operational Costs:
Economic constraints extend beyond the initial build phase to include maintenance and operational costs. Choosing components with low power consumption, such as efficient LEDs and sensors, helps reduce ongoing electricity costs. Additionally, selecting durable components minimizes the frequency and cost of replacements and repairs.

Leveraging Open-Source Software and Tools:
Utilizing open-source software and tools is another effective way to manage costs. Open-source solutions are typically free and supported by a community of developers, offering robust functionality without the licensing fees associated with proprietary software. This approach can significantly reduce software development and deployment costs.

Takeaways:
Economic constraints play a crucial role in shaping the design and implementation of our project. By carefully managing the budget, prioritizing cost-effective components, and employing strategies such as bulk purchasing and in-house solutions, we can deliver a functional and reliable system within the allocated budget. Balancing cost and quality, leveraging open-source tools, and making informed trade-offs are essential to achieving the project's goals without exceeding financial limitations.

Accuracy vs. Cost Trade-off
Importance of Sensor Accuracy:
Sensor accuracy is critical in our system, as it directly influences the quality of data collected and, consequently, the decisions made based on that data. Accurate sensors ensure that the environmental parameters such as light intensity, color, temperature, and humidity are monitored precisely. This precision is essential for optimizing plant growth conditions, diagnosing plant health issues, and ensuring the safety of the lighting system.

Cost Constraints:
However, achieving high sensor accuracy often comes at a high cost. High-precision sensors typically involve more sophisticated technology, advanced materials, and rigorous manufacturing processes, all of which contribute to higher prices. Given the budget limitations of our project, it is impractical to equip the system with the highest-end sensors available on the market.

Balancing Accuracy and Cost:
To address this challenge, we employ a strategic approach to balance accuracy and cost. This involves selecting sensors that provide adequate accuracy for our application while remaining within our budget constraints. For instance, instead of opting for the most expensive sensors with the highest accuracy, we select mid-range sensors that offer a reasonable trade-off between performance and cost. Additionally, we implement calibration techniques and error correction algorithms to enhance the accuracy of these sensors without significantly increasing costs.

Case Study: Light Sensors:
For our light sensors, we choose models that offer sufficient accuracy for detecting the light intensity required for optimal plant growth. While high-end spectrometers could provide superior accuracy, their cost is prohibitive. Instead, we use photodiodes or phototransistors with moderate accuracy and employ calibration procedures to ensure their readings are reliable. This approach allows us to maintain accurate light measurements while staying within budget.

Necessity of Recalibration:
Color sensors, which are crucial for monitoring the color of plant leaves and diagnosing potential issues, require periodic recalibration to maintain their accuracy. Over time, factors such as sensor aging, environmental conditions, and exposure to varying light intensities can affect the sensor's performance, leading to drifts in accuracy.

Recalibration Procedures:
Recalibration involves adjusting the sensor's output to match a known reference or standard. This process compensates for any deviations and ensures that the sensor continues to provide accurate color measurements. Recalibration can be performed using calibration tools and reference materials, such as standardized color targets or light sources with known spectral properties.

Impact on Cost:
While recalibration is essential for maintaining sensor accuracy, it introduces additional costs related to calibration equipment, labor, and potential downtime. These costs must be factored into the overall budget and maintenance plan for the system. To minimize these expenses, we explore cost-effective recalibration methods and schedule recalibration at optimal intervals to balance accuracy and cost.

Example: Implementation in Color Sensors:
In our system, color sensors are periodically recalibrated using standardized color targets that simulate the range of colors observed in plant leaves. This ensures that the sensors accurately detect changes in leaf color, which are indicative of plant health. By incorporating recalibration into our maintenance schedule, we maintain high accuracy in color detection while managing costs effectively.

Modular Design:
Implementing a modular design allows for flexibility in sensor selection and upgrades. By designing the system with interchangeable sensor modules, we can start with cost-effective sensors and upgrade to higher-precision models as budget permits. This approach ensures that the system remains functional and accurate while providing a pathway for future improvements.

Regular Maintenance and Calibration:
Scheduling regular maintenance and calibration ensures that sensors continue to perform accurately over time. By planning these activities and budgeting for them, we can maintain sensor accuracy without incurring unexpected costs. Regular calibration also helps in identifying and addressing any sensor degradation early, preventing costly errors or system failures.

Takeaways:
Balancing sensor accuracy and cost is a critical constraint in the design and development of our horticultural lighting and sensing system. By optimizing sensor selection, implementing periodic recalibration, and employing strategies such as modular design and data fusion, we can achieve the necessary accuracy within our budget limitations. This approach ensures that the system remains reliable, effective, and affordable, ultimately contributing to the success of our project.

Maintenance Requirements
Regular Calibration of Color Sensors:
The accuracy and reliability of color sensors are paramount for monitoring and maintaining optimal plant growth conditions. Over time, the performance of these sensors can drift due to factors such as environmental conditions, aging of components, and exposure to varying light intensities. Regular calibration is essential to ensure that the color sensors continue to provide precise and accurate measurements.

Calibration Procedure:
Calibration involves comparing the sensor’s readings with a known standard or reference. This process typically requires a controlled environment where variables such as light intensity and spectral distribution can be precisely managed. The calibration procedure may involve:
- Using reference color samples with known colorimetric values.
- Adjusting the sensor’s output to match the reference values.
- Recording the calibration data and applying it to the sensor’s readings to correct any deviations.

Frequency of Calibration:
The frequency of calibration depends on the usage conditions and the stability of the sensors. For our horticultural system, it is recommended to perform calibration:
- Initially, before the first use to establish a baseline.

- After any significant changes in the environment or lighting conditions, such as moving the system to a different location or replacing the lighting components.

Challenges and Solutions:
Calibration can be time-consuming and requires specialized equipment and knowledge. To address these challenges, we can:

- Develop a detailed calibration protocol that can be followed by maintenance personnel.
- Invest in automated calibration tools that simplify the process.
- Provide training for users and maintenance staff to ensure they understand the importance and procedure of calibration.

Importance of Maintenance:
The hydroponics system is the backbone of our plant growth setup, providing essential nutrients and water to the plants. Over time, parts of the hydroponics system can accumulate debris, algae, and mineral deposits, which can impede water flow and nutrient delivery. Regular cleaning and part replacement are critical to maintaining the efficiency and longevity of the system.

Cleaning Procedure:
Regular cleaning involves:

- Flushing the water reservoir and pipelines to remove any buildup of algae or mineral deposits.
- Cleaning the water pump and filters to ensure unobstructed water flow.
- Sanitizing the water reservoir and growing containers to prevent the growth of harmful bacteria and fungi.
- The cleaning process should be carried out using non-toxic, plant-safe cleaning agents to avoid any adverse effects on plant health.

Frequency of Cleaning:
The hydroponics system should be cleaned:

- Monthly, to prevent the buildup of algae and mineral deposits.
- After each growing cycle, to prepare the system for the next batch of plants.
- Immediately, if any signs of contamination or blockage are observed.

Part Replacement:
Some components of the hydroponics system, such as water pumps, filters, and tubing, may need to be replaced periodically due to wear and tear. The frequency of part replacement depends on the quality of the components and the intensity of use. It is recommended to:

- Inspect the components regularly for any signs of wear or damage.
- Replace water pumps and filters every 6-12 months, depending on their condition and performance.
- Replace tubing and other plastic components if they show signs of brittleness or leaks.

Takeaways
Regular maintenance is vital for maximizing the performance and lifespan of our horticultural lighting and hydroponics system. A structured maintenance plan, encompassing color sensor calibration, hydroponic system cleaning and part replacement, and thorough training and documentation, is essential. By proactively addressing system needs, we create an optimal environment for plant growth, minimize unexpected issues, extend component life, and ultimately ensure the project's success.

Adaptability Constraints in the Design
Light Requirements:
Basil requires 10-12 hours of light per day with a focus on the blue (450-480 nm) and red (620-750 nm) spectra to promote vegetative growth and flowering. Our LED lighting system is tailored to these requirements, providing the specific wavelengths and intensity needed for optimal basil growth.

Temperature and Humidity:
The ideal temperature range for basil is between 21°C and 29°C, with a relative humidity of around 50-60%. Our system maintains these conditions through precise environmental control mechanisms.

Nutrient Requirements:
Basil thrives in a nutrient-rich environment with specific needs for nitrogen, phosphorus, and potassium. Our hydroponic system is calibrated to deliver these nutrients in the optimal ratios for basil.

Varied Light Requirements:
Different plants have varying light requirements in terms of both duration and spectrum. For instance, while basil benefits from a specific balance of blue and red light, other plants like lettuce may require a different ratio or additional green light. Adjusting the LED lighting system to cater to these varied needs without compromising efficiency is a significant challenge.

Temperature and Humidity Variations:
The optimal temperature and humidity conditions for basil may not be suitable for other herbs or plants. For example, rosemary prefers drier conditions and cooler temperatures compared to basil. This necessitates a flexible environmental control system that can be easily reprogrammed or adjusted to meet the specific needs of different plants.

Nutrient Solution Adjustments:
Different plants have unique nutrient requirements. While basil requires a balanced nutrient solution with high nitrogen levels, other plants like tomatoes may need higher potassium levels during the fruiting stage. Developing a hydroponic system

capable of dynamically adjusting nutrient ratios based on the specific requirements of each plant type is essential for adaptability.

Programmable LED Lighting:
Implementing programmable LED lighting systems that can adjust the light spectrum and intensity based on user input or preset profiles for different plants. This would involve integrating smart controls and possibly developing a user-friendly interface that allows easy customization of lighting settings.

Experimental Trials:
Conducting experimental trials with different herbs and plants to identify the optimal settings for each species. These trials would help refine the system's adaptability features and ensure that it can support a wide range of plants effectively.

User Feedback:
Gathering feedback from users who grow different plants using the system. This feedback would be invaluable in identifying areas for improvement and ensuring that the system meets the needs of diverse plant species.

Takeaways:
While our system is primarily designed for basil cultivation, addressing the adaptability constraints is crucial for broadening its applicability. By incorporating programmable lighting, modular environmental controls, and flexible nutrient delivery systems, we can enhance the system's versatility and make it suitable for growing a wide range of herbs and plants. This approach not only increases the system's value but also aligns with the broader goal of promoting sustainable and efficient indoor agriculture.

Chapter 5 - Comparison of ChatGPT with other Platforms

As students embarking on our senior design projects, we're witnessing a significant transformation in our approach to academic work, driven by the advent of Large Language Models (LLMs). These advanced AI systems are revolutionizing our capabilities in text generation, language translation, creative writing, and information retrieval. Trained on extensive datasets of text and code, LLMs offer us access to a vast repository of knowledge and analytical tools.

Our senior design projects represent a critical juncture in our academic careers, challenging us to apply our accumulated knowledge to real-world problems. The integration of LLMs into our project workflow presents exciting opportunities, from enhancing our ideation processes to facilitating in-depth research. However, we must approach this integration thoughtfully, aware of potential challenges such as the risk of encountering inaccuracies, the importance of maintaining our critical thinking skills, and the need to supplement LLM knowledge with specialized expertise in our specific design fields.

This chapter offers a comparative analysis of four prominent LLMs: ChatGPT, Claude, Llama, and Gemini. We'll examine their respective capabilities, strengths, and limitations, focusing on key performance indicators (KPIs) that are particularly relevant to our senior design projects. These KPIs include context window size, token limits, accuracy, and cost-effectiveness.

By conducting this analysis, we aim to equip ourselves with the insights necessary to select the most appropriate LLM for our individual project requirements. We recognize that each of our projects has unique needs, and the optimal choice of LLM may vary accordingly. This comparative study will enable us to make informed decisions about which LLM to incorporate into our work, potentially enhancing the quality and efficiency of our senior design projects.

5.1 Common Benchmarks for LLM Comparison

As we evaluate Large Language Models (LLMs) for our senior design projects, it's crucial that we establish clear metrics for comparison. These benchmarks will help us assess the capabilities of ChatGPT, Claude, Llama, and Gemini, allowing us to determine how each can best support our design process. We've identified four key performance indicators (KPIs) that are particularly relevant to our needs:

1. **Context Window:** This metric is essential for our complex design problems. It measures how much text an LLM can consider when generating responses. A larger context window will allow us to input more information about our design challenges, potentially leading to more nuanced and comprehensive solutions. This capability is especially valuable when we're synthesizing research or trying to grasp intricate design concepts.

2. **Token Limit:** For our senior design projects, we often need to generate lengthy reports, detailed summaries, or extensive creative content. The token limit - the maximum number of words or characters an LLM can process in one request - directly impacts our ability to receive comprehensive and detailed responses. A higher token limit will allow us to engage with the LLM more deeply on complex topics.

3. **Accuracy:** As we conduct technical research, perform data analysis, and validate our design decisions, the accuracy of the information we receive is paramount. This KPI measures how well an LLM provides factual and relevant information. We need to be confident in the data we're using to inform our design choices.

4. **Cost-Efficiency:** As students, we need to be mindful of our budgets. Different LLM platforms offer various pricing models, including free tiers and pay-per-use

options. We'll need to balance the capabilities we require with the costs we can manage.

By evaluating each LLM against these four benchmarks, we'll be able to determine which platform is best suited for the various tasks we'll encounter in our senior design projects. This analysis will help us make informed decisions about which LLM to use and when, potentially enhancing the quality and efficiency of our work.

5.2 In-Depth Analysis of Each LLM

5.2.1 ChatGPT

Key Features and Advantages:
ChatGPT is a widely-recognized LLM, praised for its intuitive interface and versatile capabilities. It excels in generating various creative text formats, translating languages, and providing informative answers. For senior design projects, ChatGPT can be particularly useful in ideation sessions, crafting project narratives, and developing engaging presentations.

Constraints:
Context Window: ChatGPT may have a more limited context window compared to some competitors. This could potentially impact its ability to fully comprehend complex design challenges, especially when dealing with extensive research or multifaceted design considerations.

Token Limit: The token limit of ChatGPT might restrict the length and detail of its outputs. This could pose challenges for tasks requiring extensive reports, in-depth summaries, or elaborate creative content relevant to design projects.

Accuracy Concerns: While ChatGPT aims for accuracy, users should be aware of potential factual errors, particularly in technical domains. Verifying ChatGPT's information with authoritative sources is crucial.

Pricing Model:
OpenAI offers a free tier of ChatGPT with basic features. For senior design projects requiring more intensive or frequent use, paid tiers with enhanced capabilities may be necessary.
Influence on Senior Design Projects:

Positive Example: ChatGPT can significantly aid in the ideation phase. By offering creative prompts and exploring diverse design approaches, it can help students overcome creative blocks and generate novel solutions.

Potential Drawback: Uncritical acceptance of ChatGPT's technical information could lead students astray during the design process. It's essential to validate technical details with reliable sources.

5.2.2 Claude

Key Features and Advantages:
Claude, an Anthropic creation, is a formidable competitor in the LLM arena. It offers a user-friendly interface and matches ChatGPT in capabilities like creative text generation, translation, and informative responses. Claude's standout feature is its:

Expanded Context Window: Claude boasts a significantly larger context window compared to ChatGPT. This enables it to process and comprehend vast amounts of information, making it particularly suited for tasks that require extensive research synthesis, comprehensive summarization, and tackling complex design challenges.

Constraints:
Potential Expense: While Claude offers a free tier with basic features, senior design projects might necessitate paid tiers for regular or intensive use. This cost factor could be challenging for some students.

Noted Accuracy Issues: Despite Claude's commitment to accuracy, some users have reported occasional factual discrepancies. As with ChatGPT, it's essential to verify information from reliable sources, especially in technical domains.

Pricing Model:
Anthropic provides a free tier for Claude with usage limitations. Premium tiers with enhanced capabilities are available at a cost, which may be a consideration for student budgets.

Influence on Senior Design Projects:
Positive Example: Claude's expanded context window makes it an excellent research tool for senior design projects. It can efficiently process large volumes of information, enabling students to gain comprehensive insights into complex design topics and identify relevant research trends.

Potential Drawback: While informative, Claude's responses might sometimes be overly technical and lack direct practical application in the design process. Students need to effectively translate Claude's information into actionable steps relevant to their specific design projects.

5.2.3 Llama

Key Features and Advantages:
Meta's LLM, known as Llama, distinguishes itself through its emphasis on parameter efficiency. Available in multiple sizes (7 billion, 13 billion, and 70 billion parameters), Llama strikes a balance between power and resource utilization. It excels in:

Informative Responses: Llama can provide clear, informative answers to design-related queries, drawing from its extensive knowledge base.

Potential Visual Capabilities: A noteworthy feature of Llama is its reported ability to generate visual aids such as concept sketches based on design ideas. This could prove invaluable for initial brainstorming and visualization.

Constraints:
Access Restrictions: As of July 2024, Llama may still be in a closed beta phase, limiting its availability to many students. This could restrict its use in senior design projects.
Possible Design Discipline Focus: There's a possibility that Llama might be tailored to specific design fields (e.g., graphic design, product design). If your senior design project falls outside its focus area, its applicability might be limited.

Pricing Model (based on available information):
As a Meta product, Llama's cost structure remains uncertain. It's possible that Meta might offer free access for research purposes, but this requires confirmation.

Influence on Senior Design Projects:
Positive Example: If accessible, Llama's potential to generate concept sketches could be revolutionary for senior design projects. It could help students swiftly visualize their design ideas, promoting creative exploration.

Potential Drawback: Limited accessibility due to the closed beta phase could hinder Llama's usefulness for senior design projects. Moreover, if Llama focuses on specific design disciplines, its application might be restricted depending on your project's field.

5.2.4 Gemini

Key Features and Advantages:
Gemini, developed by Google AI, is a robust LLM known for its outstanding performance across various benchmarks. It excels in:

Comprehensive Responses: Like Llama, Gemini can provide thorough and informative answers to design-related queries, leveraging its vast knowledge base.

Up-to-Date Information Access: A significant advantage of Gemini is its capability to access and process real-time data. This allows it to stay current with the latest trends, offering valuable insights for design projects that require contemporary market analysis or competitive research.

Constraints:
Possible Token Restrictions: While specific details are not yet public, Gemini might have a token limit that could constrain the length and depth of its outputs.

This could pose challenges when generating extensive reports or complex design documents.

General Knowledge Focus: There's a possibility that Gemini might prioritize broad information retrieval over specialized design knowledge. For tasks requiring highly specific design expertise, other LLMs like Claude might be more suitable.

Pricing Model (as of July 2024):
Currently, Gemini offers free public access, making it a cost-effective choice for students engaged in senior design projects.

Influence on Senior Design Projects:
Positive Example: Gemini's access to real-time data makes it an invaluable tool for researching current market trends and analyzing competitor strategies. This information can be crucial in guiding design decisions and ensuring your project's market relevance.

Potential Drawback: If Gemini favors general information over design-specific knowledge, it might oversimplify complex design challenges. It's crucial to apply your design expertise to evaluate and refine the information Gemini provides.

5.3 Comparative Analysis and Recommendations

This section builds on our individual examinations of ChatGPT, Claude, Llama, and Gemini. We'll now compare these LLMs side by side, focusing on their capabilities and limitations based on our chosen key performance indicators (KPIs): context window, token limit, accuracy, and cost-efficiency.

By thoroughly comparing these aspects, we aim to provide you with a clear understanding of how each LLM performs in areas crucial to senior design projects. This comparison will help you see the strengths and weaknesses of each platform in relation to one another, rather than just in isolation.

Our goal is to equip you with the knowledge needed to choose the LLM that best fits your specific senior design project requirements. We recognize that different projects may have varying needs, and what works best for one project might not be ideal for another.

This comparative analysis will highlight which LLM might be most suitable for certain types of tasks or project phases. For example, one LLM might excel at initial brainstorming, while another could be better suited for in-depth research or technical writing.

By the end of this section, you should have a comprehensive view of how these LLMs stack up against each other. This will enable you to make an informed decision about which platform to use, ensuring you can leverage the most appropriate tools to enhance your senior design project's quality and efficiency.

Table 5.1. LLM Comparison

KPI	ChatGPT	Claude	Llama	Gemini
Context Window	Limited	Large	Potentially Large	Large
Token Limit	Moderate	High	Unclear	Potentially Moderate
Accuracy	Generally good, requires verification	Generally good, requires verification	Unclear	Generally good, requires verification
Cost-Efficiency	Free tier with limitations, Paid tiers available	Free tier with limitations, Paid tiers available	Unclear (Potential free access)	Free access (as of July 2024)

Balancing Options and Recommendations

Selecting the most suitable LLM for your senior design project requires a thoughtful assessment of your project's unique needs:

Project Complexity: For projects involving extensive research synthesis and analysis of complex design concepts, Claude's expansive context window makes it a strong candidate. Its ability to process and understand large amounts of information could be particularly beneficial for intricate design challenges.

Current Information Requirements: If your project demands up-to-date market analysis and competitor research, Gemini's capability to access and process real-time data offers a significant advantage. This feature could be crucial for projects that need to stay aligned with current industry trends.

Idea Generation: When brainstorming and exploring design ideas is a key focus, ChatGPT's user-friendly interface and proficiency in creative text generation can be invaluable. Its ability to assist in generating diverse ideas could help stimulate innovative thinking.

Visual Conceptualization: If visualizing design concepts is an important aspect of your project, Llama's potential for generating concept sketches could be highly beneficial (subject to availability). This feature could significantly enhance the ideation and communication phases of your design process.

Budget Considerations: As of July 2024, Gemini offers free access, making it an attractive option for budget-conscious students. However, if your project requires in-depth design knowledge and your budget allows, exploring the paid tiers of Claude might be a worthwhile investment.

Important Considerations: It's crucial to recognize that no single LLM is flawless, and each has its limitations. Here are some general guidelines to keep in mind:

Utilize Multiple LLMs: Don't hesitate to leverage the strengths of different LLMs. For example, you might use Gemini for initial market research and then switch to Claude for more detailed analysis of specific design topics. This approach allows you to benefit from the unique strengths of each platform.

Exercise Critical Judgment: Always verify the information provided by LLMs using reputable sources, especially in areas where factual accuracy is crucial. Developing a habit of cross-checking information will help ensure the reliability of your project.

Prioritize Design Expertise: While LLMs are powerful tools, they should complement, not replace, your design knowledge and critical thinking skills. Use these platforms to enhance and streamline your design process, but rely on your expertise for core decision-making and creative direction.

5.4 Takeaways

In this chapter, we've explored how Large Language Models (LLMs) can potentially boost our senior design projects. We compared four major LLMs – ChatGPT, Claude, Llama, and Gemini – looking at what they can do, where they fall short, and how much they cost. We used key performance indicators (KPIs) like context window, token limit, accuracy, and cost-efficiency to break down these factors, aiming to help us make smart choices when using LLMs in our projects.

It's crucial that we pick the right LLM based on what our specific project needs. Some LLMs are great for research (like Claude), others for real-time data (Gemini), or creative writing (ChatGPT). Understanding these strengths and weaknesses will help us use the best features for our individual projects.
We should keep in mind that LLMs aren't replacements for our own design skills. They're here to help with research, brainstorming, and accessing lots of information. We still need to think critically about what they produce and apply our own design thinking – that's key to doing well.
LLM technology is changing fast, and we're just starting to see how it might impact design education. Future improvements could mean even more powerful LLMs tailored specifically for different design fields. This could open up some exciting possibilities for us and future design students.
This chapter is our starting point for figuring out how to use LLMs to improve our senior design projects. By understanding what these tools can and can't do, we can really make the most of them and take our design thinking to the next level.

Chapter 6 - Hardware Design

This section provides a detailed and comprehensive overview of the hardware design for the our system. It delves into the explicit design and integration of key optical components, including the TCS34725 color sensor, the OPT3001 ambient light sensor, and the WS2812B RGB LED system. Each of these components plays a critical role in ensuring the optimal functioning of the hydroponic system, and their integration is carefully planned and executed to maximize efficiency and effectiveness.
In addition to the detailed description of the optical components, this section includes a variety of visual aids to enhance understanding. Subsystem block diagrams are provided to illustrate the interconnections between various hardware elements, showcasing how each component interacts within the broader system. Schematic diagrams offer a more granular view, detailing the specific electrical connections and circuit designs that underpin the system's operation. Furthermore, structural illustrations are included to provide a clear visual representation of the overall architecture, highlighting the physical layout and spatial arrangement of the hardware components.
Through these detailed descriptions and visual aids, readers will gain a comprehensive understanding of the hardware components and their specific roles within the our system. This section not only explains the functionality of each component but also demonstrates how they collectively contribute to the system's ability to maintain optimal growing conditions for the plants. By presenting a clear and thorough analysis of the hardware design, this section ensures that readers are well-informed about the technical foundations of the BASIL H.E.R.B. project and the innovative approaches used to achieve its goals.

6.1 Power Requirement

The system's power demand is supplied by a standard 120 VAC, 15 A wall outlet, using an AC to DC power adapter. This adapter converts the high voltage alternating current to low voltage direct current, enabling the system to operate safely and efficiently. The use of low voltage DC is particularly important in this application due to the presence of open access to water, which increases the risk of electrical hazards. By operating at a lower voltage, the risk of electrical shock or short circuits is significantly reduced.

To determine the precise low voltage DC requirements for the system, a thorough investigation of the power needs of all connected devices was conducted. This investigation involved analyzing the power consumption and voltage specifications of each device to ensure they operate within safe and optimal parameters. The detailed results of this investigation are presented in Table 6.1, which outlines the specific power requirements and provides a comprehensive overview of the system's electrical needs.

Table 6.1. Power Requirement Segregated by Voltage Type 5 VDC

Voltage	Part#	Part Description	QTY	Current Total
5 VDC	SEN0161	pH Sensor	1	50uA
5 VDC	QR50L	Water pump	1	0.96A
5 VDC	TPS565208	Regulator, 5V	1	600uA
5 VDC	WS2812B-8x8	LEDs for plant	1	2.6A
5 VDC	G5LE-1 DC5	Relay	2	200mA
Total 5V Current				**3.76A**
Total 5V Power				**18.8W**

Table 6.2. Power Requirement Segregated by Voltage Type 3.3 VDC

Voltage	Part#	Part Description	QTY	Current Total
3.3 VDC	ESP32-WROOM-32	Microcontroller	1	0.5A
3.3 VDC	DHT-11	Temperature and humidity sensor	1	150uA
3.3 VDC	OPT3001	Ambient Light Sensor	1	1.8uA
3.3 VDC	TPS565208	Regulator, 3.3V	1	600uA
Total 3.3V Current				**0.50A**

Total 3.3V Power	**1.65W**

Table 6.3. Power Requirement Segregated by Voltage Type 1.8 VDC and System Total Power

Voltage	**Part#**	**Part Description**	**QTY**	**Current Total**
1.8 VDC	TPS565208	Regulator, 1.8V	1	600uA
1.8 VDC	TCS34725	Color Sensor	1	235mA
Total 1.8V Current				**0.24A**
Total 1.8V Power				**0.432W**
System Total Power				**20.88W**

The results of the investigation reveal that the system requires three different voltages: 5 VDC, 3.3 VDC, and 1.8 VDC. Efforts were made to consolidate the voltage requirements of the devices as much as possible. After identifying the necessary voltage levels, the total current required for each voltage type was calculated. However, due to the varying voltage requirements, simply summing the total system current would not accurately reflect the system's power needs. Therefore, each voltage and current requirement was converted into its respective power requirement in watts to normalize the results.

The total power requirement for the system was calculated to be 20.88 watts. To ensure reliable operation and account for any potential power surges, a design overhead of 20% was added.

$$System\ Total\ Power\ Required\ =\ 20.88\,W\ *\ 1.2\ = 25.06\,W$$

Thus, the AC to DC power supply adapter must provide 5 VDC at approximately 25.06 watts, which translates to 5 VDC at 5 amps.

After SD1, it was determined that the color sensor we have selected could be powered via 3.3VDC, which eliminated the need for the 1.8VDC circuit. In total the AC / DC power adapter needed was 12VDC at 3A.

6.2 Overall System Schematic

The system schematic was developed using KiCAD which is a free open source electrical design CAD software that can develop schematics utilizing its parts and symbol library out of the box in addition to having the ability to create custom symbols and download additional symbols, PCB footprints, and 3D models from sites such as Ultra librarian, Mouser, and Digi-key. The contents of the schematics extend beyond the PCB and include power upstream of the PCB as well as loads downstream from the PCB. This can be seen in Figure 6.1 which provides an overall system schematic.

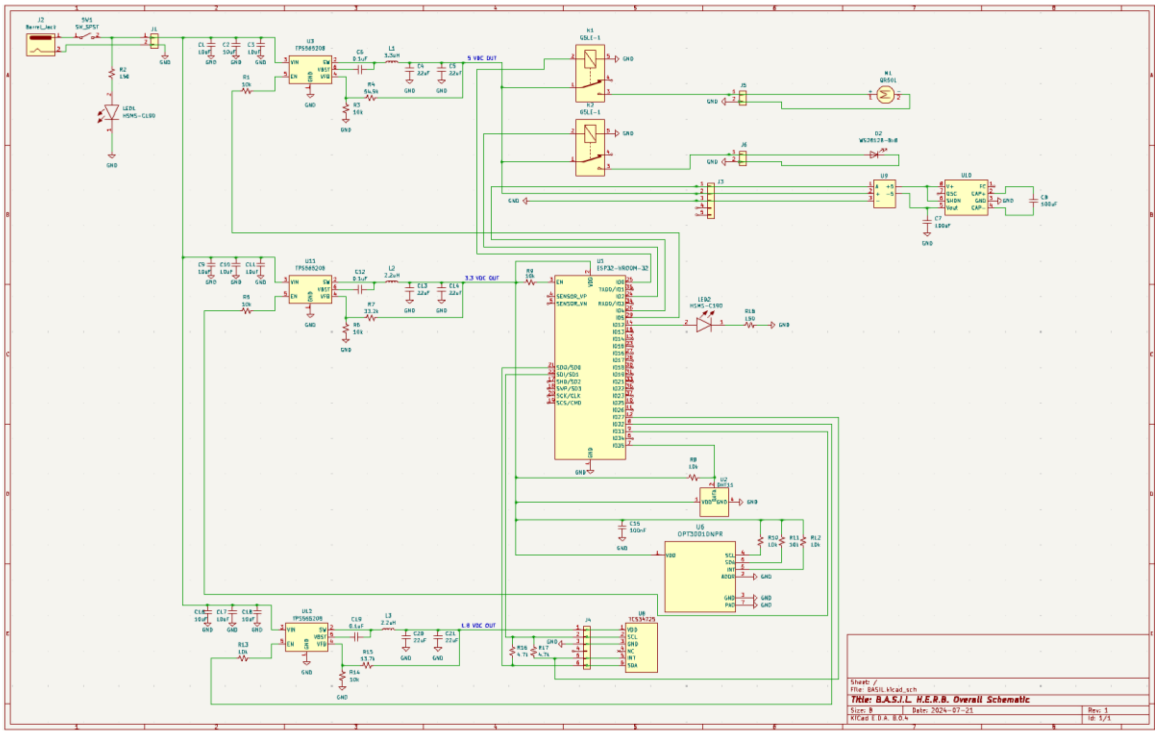


Figure 6.1. Overall System Schematic

6.3 Voltage Regulation

Voltage regulation is crucial in electronic systems to ensure that devices receive a stable and consistent voltage supply. This protects electronic components from damage that can be caused by voltage spikes or drops, which could otherwise lead to malfunction or premature failure. It also ensures that circuits function reliably and maintain consistent performance, as many rely on precise voltage levels.

By providing only the necessary voltage, voltage regulation improves efficiency and reduces energy waste. Additionally, it helps filter out electrical noise and interference, resulting in cleaner power delivery and better performance of sensitive components. Proper voltage regulation also aids in heat management, as excessive voltage can lead to increased heat dissipation, affecting the longevity and reliability of the device.

In our system the devices require a step down from 5 VDC to 3.3 VDC and also 1.8 VDC. This can be seen in Figure 6.2. The voltage regulator chosen for this project was a common part for all circuits due to the wide input voltage range of 4.5V to 17V and 5A output exceeded the demand for any of the circuits in our system. The TPS565208 by texas instruments is a synchronous step down voltage regulator that provides fast transient response time and can be used with common resistors and capacitors in different combinations to provide the voltage outputs required by each circuit.

Finally, voltage regulation is needed to protect components, ensure stable operation, improve efficiency, reduce noise, and manage heat within electronic systems.

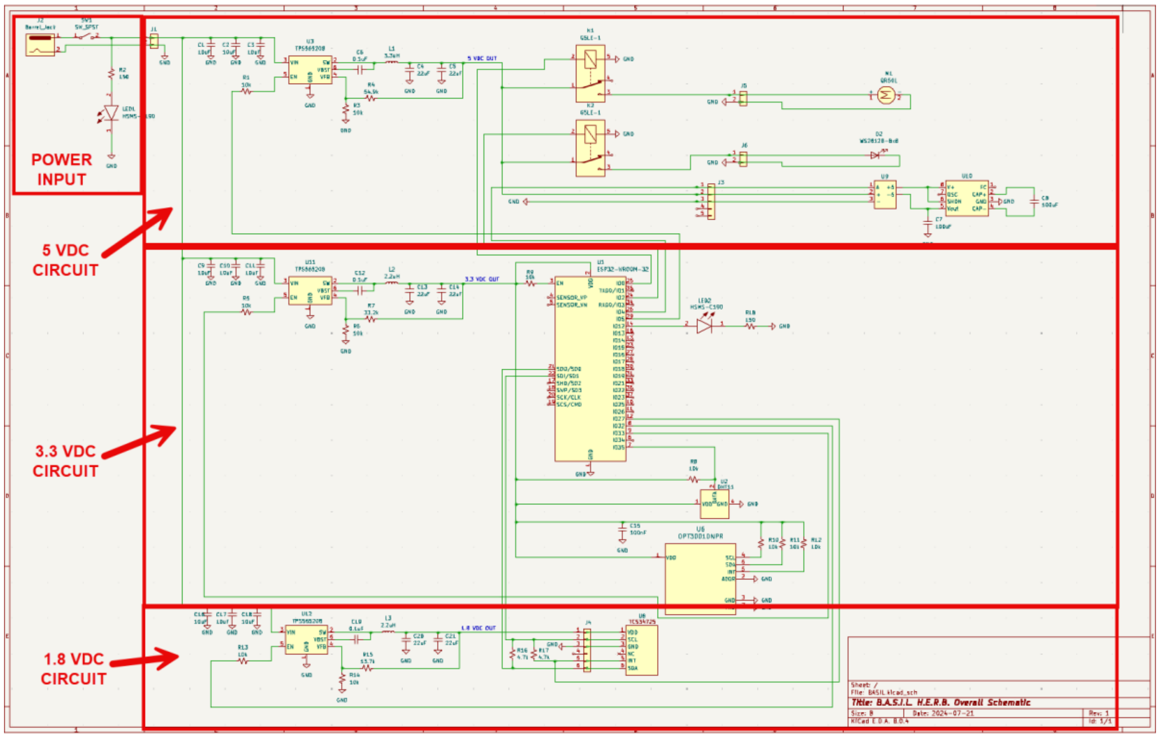


Figure 6.2. System Schematic Divided into Subsystem Circuits

6.3.1 Subsystem 5 VDC Circuit

Figure 6.3. Depicts the first subsystem to be discussed is the 5 VDC circuit. This circuit is responsible for four key tasks. These are; providing power for the submerged water pump, water pump motor relay control circuits, power for the growth stimulation LEDs, and finally the pH sensor circuit. Texas Instruments provides a typical application diagram, Figure 6.4, along with a table of recommended component values to obtain each of the three output values needed for this system. Figure 6.5 shows the component values highlighted to obtain a 5 VDC output.

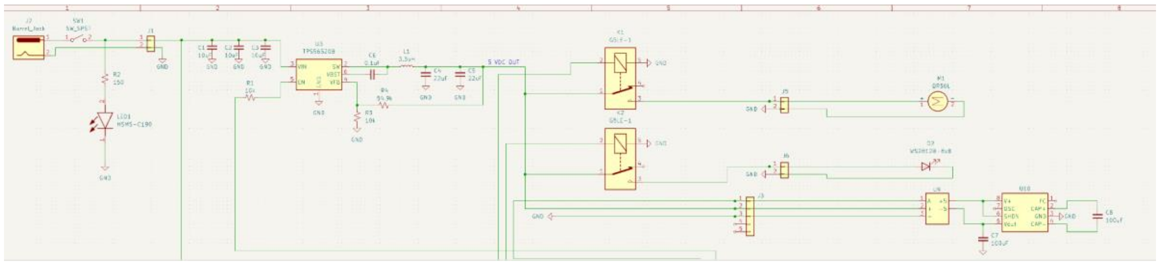


Figure 6.3. 5VDC Subsystem

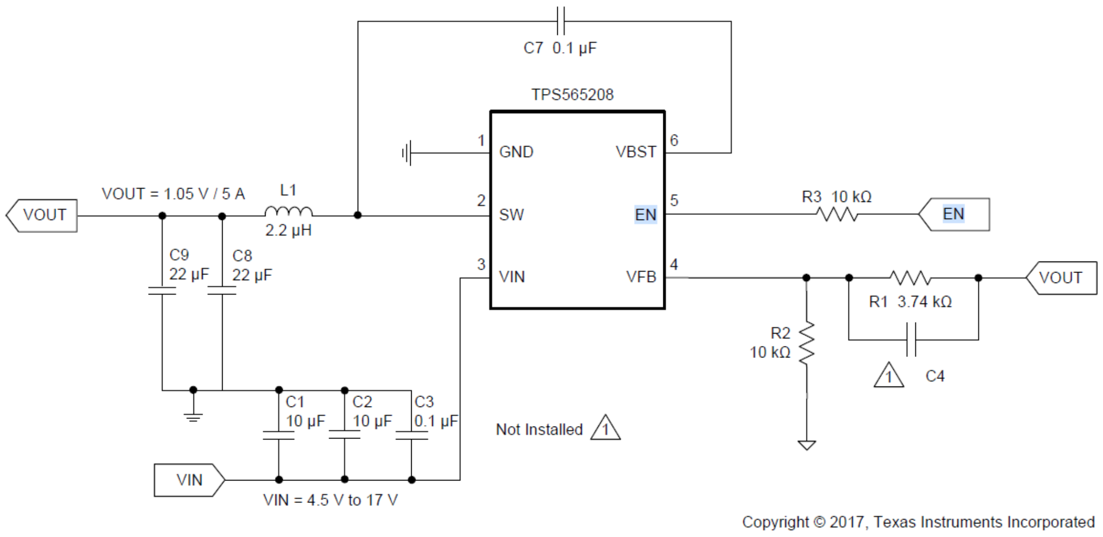
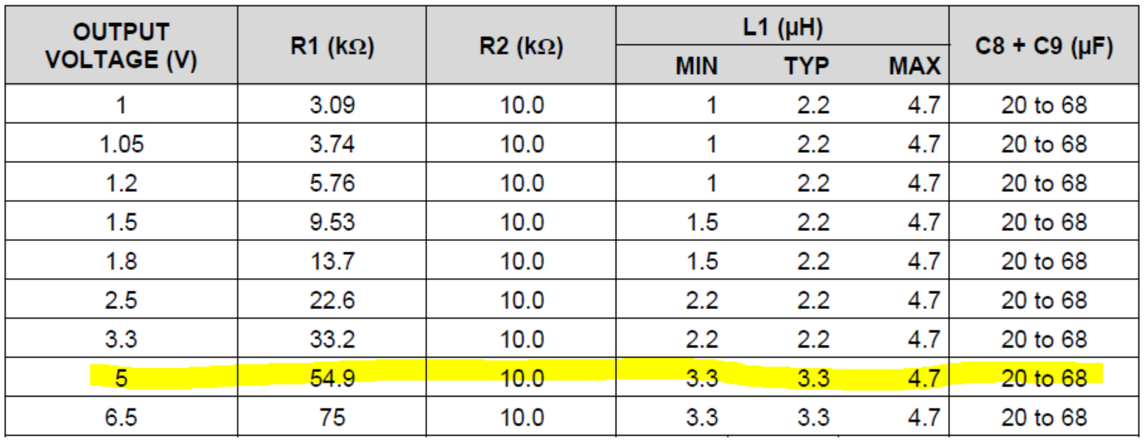


Figure 6.4. Typical Application of TPS565208 by Texas Instruments



OUTPUT VOLTAGE (V)	R1 (kΩ)	R2 (kΩ)	L1 (µH)			C8 + C9 (µF)
			MIN	TYP	MAX	
1	3.09	10.0	1	2.2	4.7	20 to 68
1.05	3.74	10.0	1	2.2	4.7	20 to 68
1.2	5.76	10.0	1	2.2	4.7	20 to 68
1.5	9.53	10.0	1.5	2.2	4.7	20 to 68
1.8	13.7	10.0	1.5	2.2	4.7	20 to 68
2.5	22.6	10.0	2.2	2.2	4.7	20 to 68
3.3	33.2	10.0	2.2	2.2	4.7	20 to 68
5	54.9	10.0	3.3	3.3	4.7	20 to 68
6.5	75	10.0	3.3	3.3	4.7	20 to 68

Figure 6.5. Recommended Component Values to Obtain Vout = 5 VDC

6.3.2 Subsystem 3.3 VDC Circuit

Figure 6.6. Depicts the 3.3 VDC circuit This circuit is responsible for providing power to the MCU, temperature and humidity sensor, and the ambient light sensor. Also figure 6.7 shows the components required to obtain Vout = 3.3 VDC.

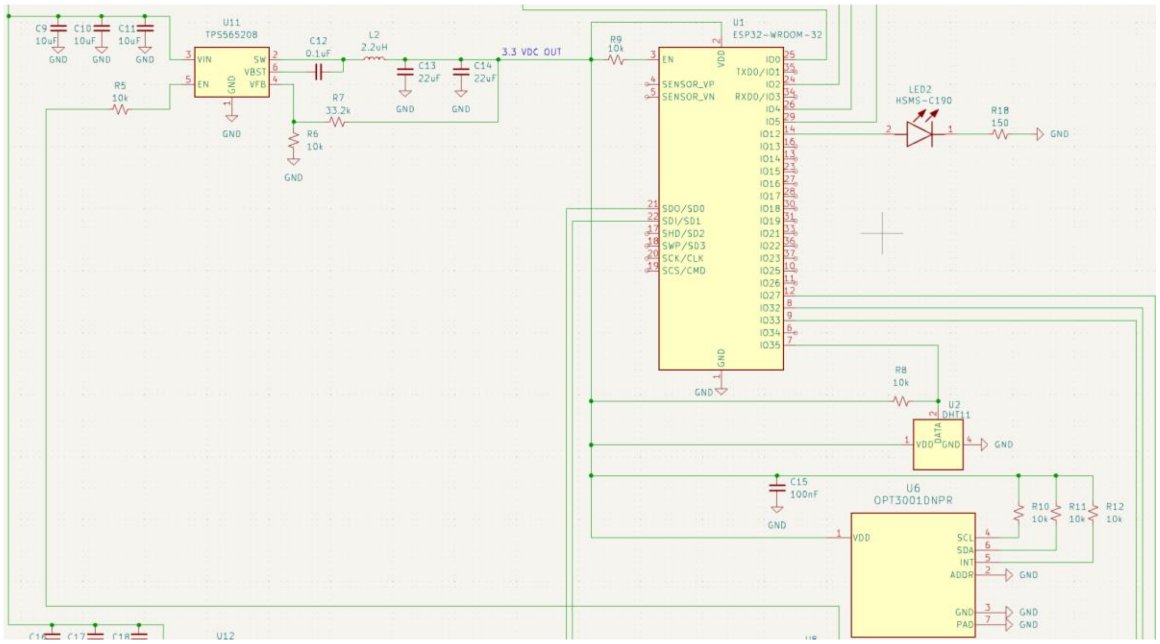
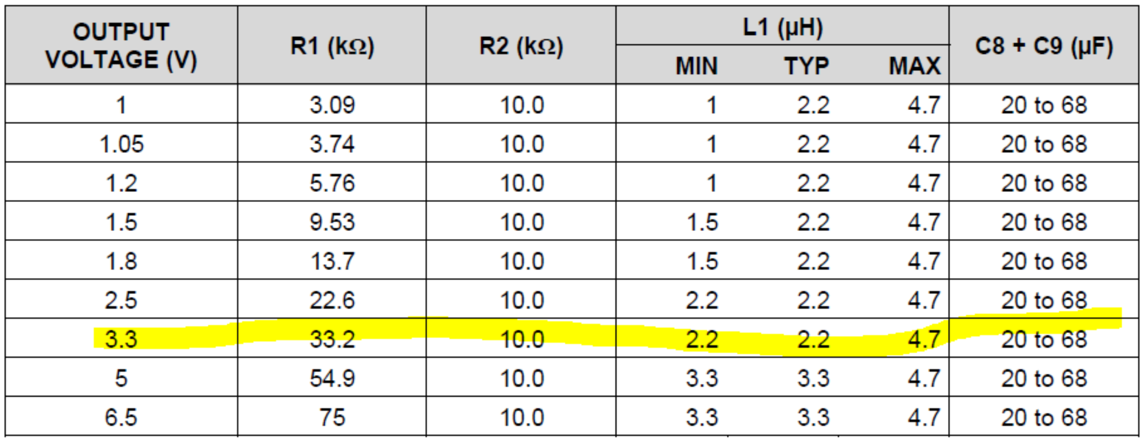


Figure 6.6. 3.3VDC Subsystem



OUTPUT VOLTAGE (V)	R1 (kΩ)	R2 (kΩ)	L1 (µH)			C8 + C9 (µF)
			MIN	TYP	MAX	
1	3.09	10.0	1	2.2	4.7	20 to 68
1.05	3.74	10.0	1	2.2	4.7	20 to 68
1.2	5.76	10.0	1	2.2	4.7	20 to 68
1.5	9.53	10.0	1.5	2.2	4.7	20 to 68
1.8	13.7	10.0	1.5	2.2	4.7	20 to 68
2.5	22.6	10.0	2.2	2.2	4.7	20 to 68
3.3	33.2	10.0	2.2	2.2	4.7	20 to 68
5	54.9	10.0	3.3	3.3	4.7	20 to 68
6.5	75	10.0	3.3	3.3	4.7	20 to 68

Figure 6.7. Recommended Component Values to Obtain Vout = 3.3 VDC

6.3.3 Subsystem 1.8 VDC Circuit

The 1.8 VDC circuit is an essential component of our design, providing the necessary power to the color sensor to ensure its accurate and efficient operation. This circuit is detailed in Figure 6.8, illustrating the precise layout and connections required to deliver a stable 1.8 VDC output. The color sensor, which plays a critical role in monitoring the health and nutrient status of the plants, is mounted on its own dedicated development board. To facilitate seamless integration between the sensor board and the main PCB, a specialized connector is included on the PCB. This connector serves as a reliable interface for the wires, ensuring secure and efficient power transfer between the two boards.

In addition to the physical layout and interconnection details, the specific component configuration necessary to achieve a stable Vout of 1.8 VDC is depicted in Figure 6.9. This configuration includes all the essential components such as resistors, capacitors, and any other necessary elements arranged in a manner that ensures consistent voltage output. The diagram provides a clear visual representation of how these components are organized and interact to maintain the required voltage level. This careful design and configuration are crucial for the optimal performance of the color sensor, as it ensures that the sensor receives a stable and precise voltage supply, enabling it to function accurately and reliably within the system.

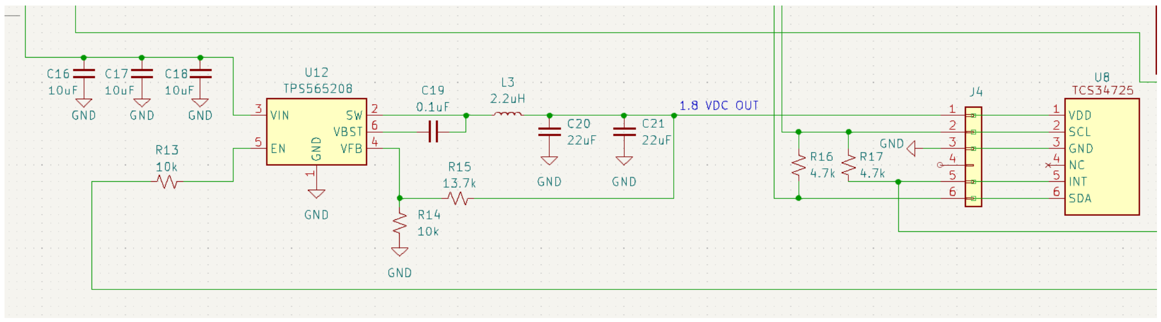
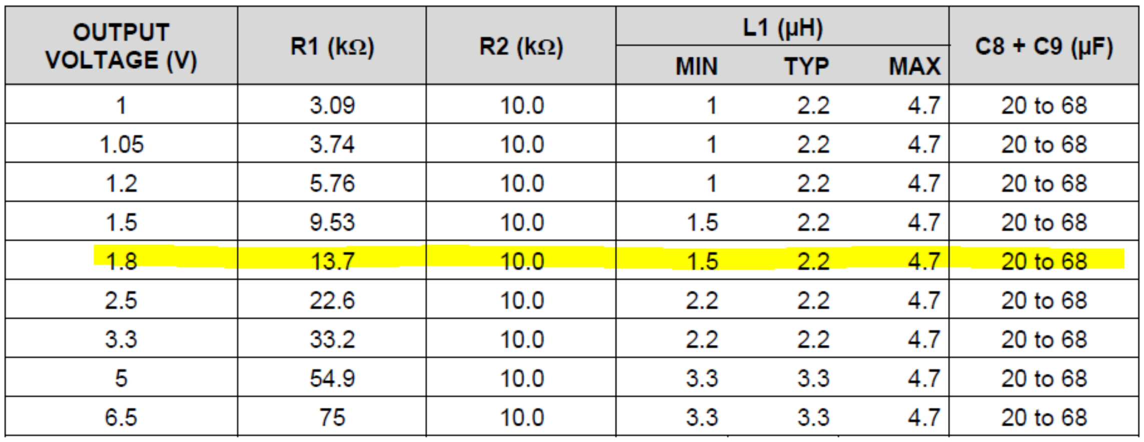


Figure 6.8. 1.8 VDC Subsystem



OUTPUT VOLTAGE (V)	R1 (kΩ)	R2 (kΩ)	L1 (μH)			C8 + C9 (μF)
			MIN	TYP	MAX	
1	3.09	10.0	1	2.2	4.7	20 to 68
1.05	3.74	10.0	1	2.2	4.7	20 to 68
1.2	5.76	10.0	1	2.2	4.7	20 to 68
1.5	9.53	10.0	1.5	2.2	4.7	20 to 68
1.8	13.7	10.0	1.5	2.2	4.7	20 to 68
2.5	22.6	10.0	2.2	2.2	4.7	20 to 68
3.3	33.2	10.0	2.2	2.2	4.7	20 to 68
5	54.9	10.0	3.3	3.3	4.7	20 to 68
6.5	75	10.0	3.3	3.3	4.7	20 to 68

Figure 6.9. Recommended Component Values to Obtain Vout = 1.8 VDC

Changes were made following the design shown in SD1. The following shows the updated schematics developed in SD2.

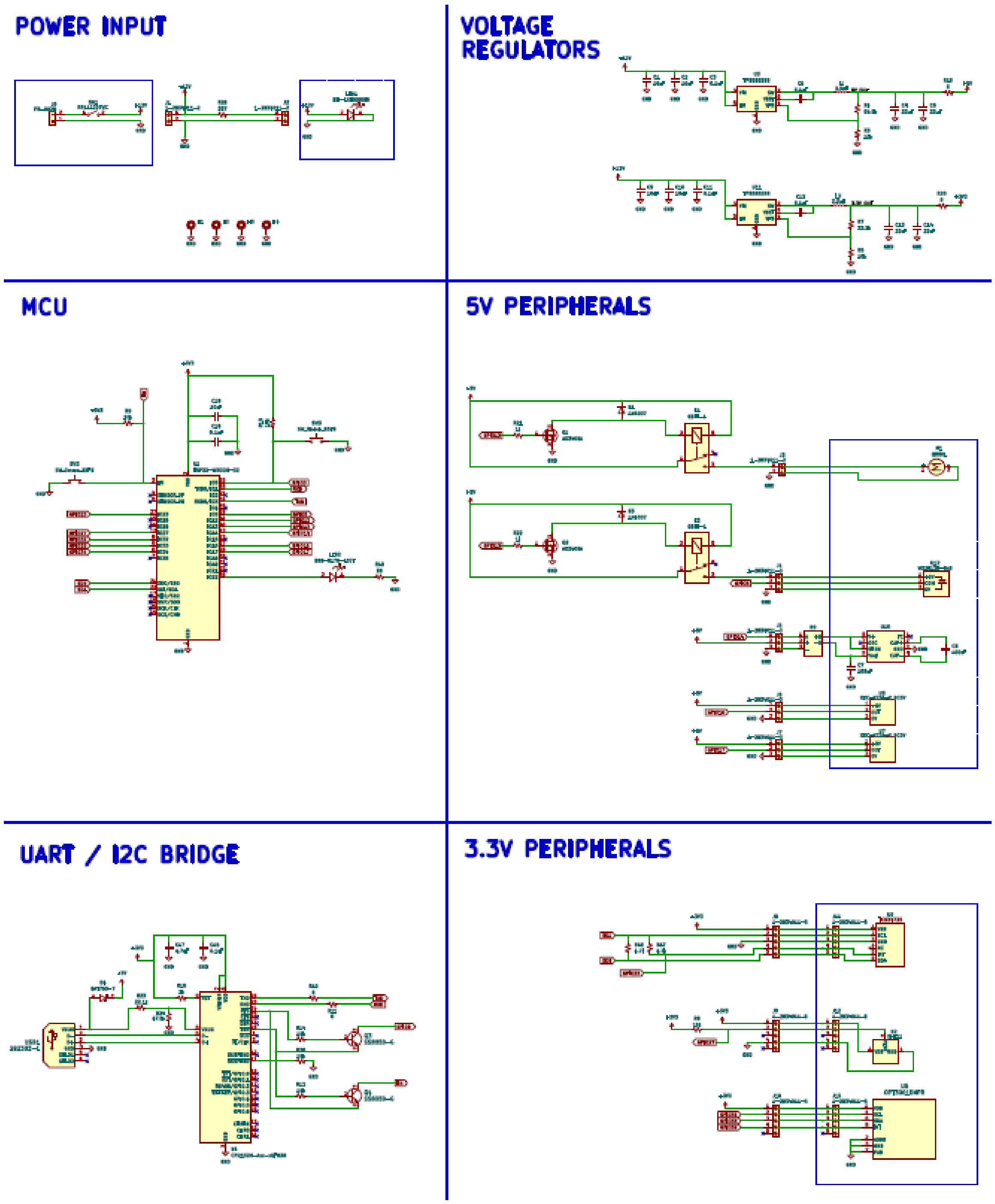


Figure 6.10. SD2 Updated Overall System Schematics

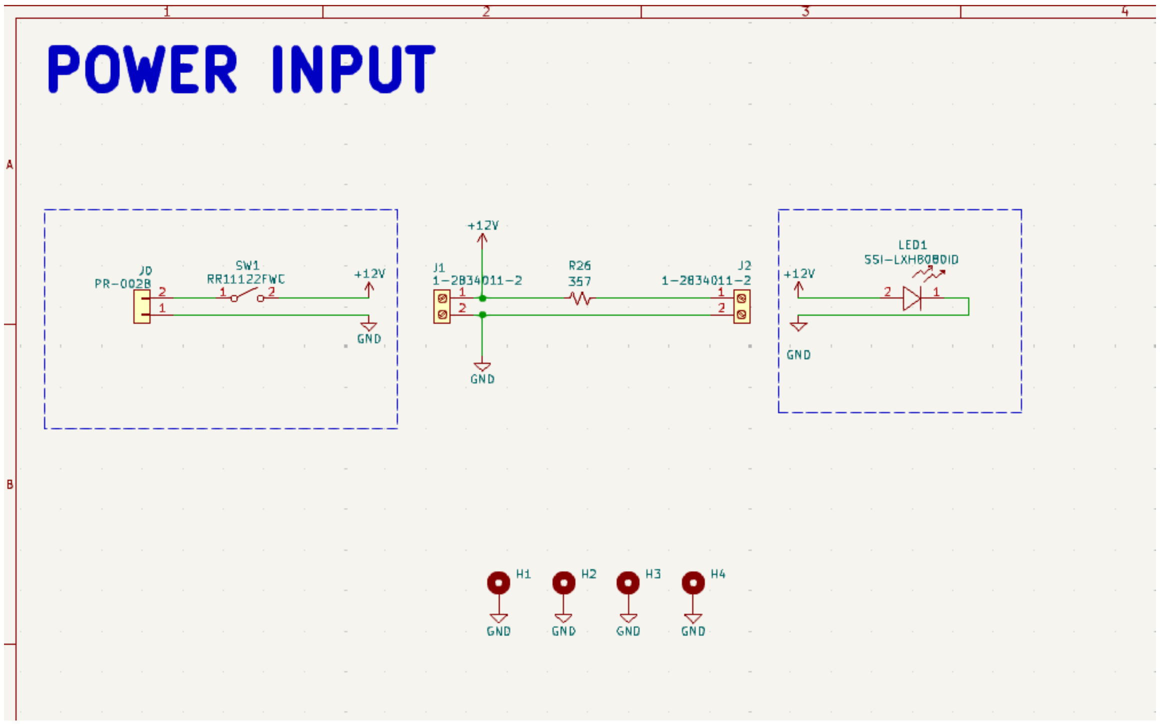


Figure 6.11. SD2 Updated Power Input Schematic

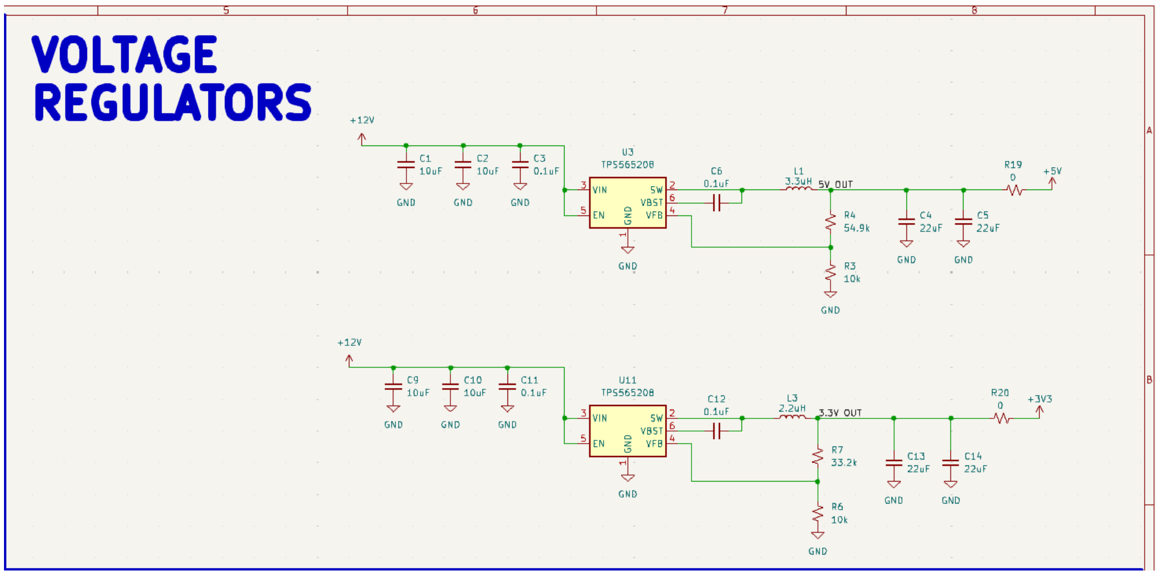


Figure 6.12. SD2 Updated Voltage Regulators Schematic

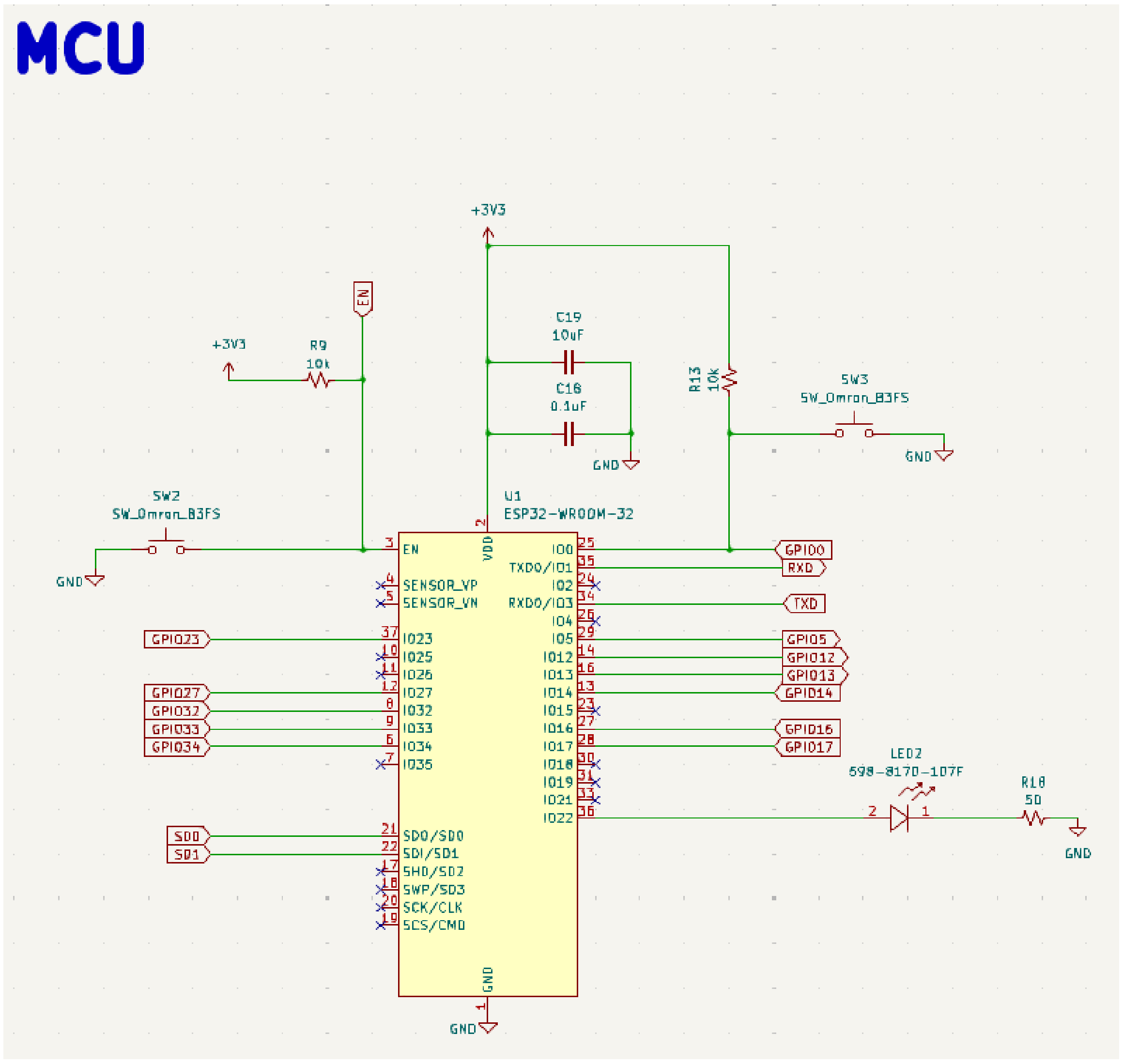


Figure 6.13. SD2 Updated MCU Schematic

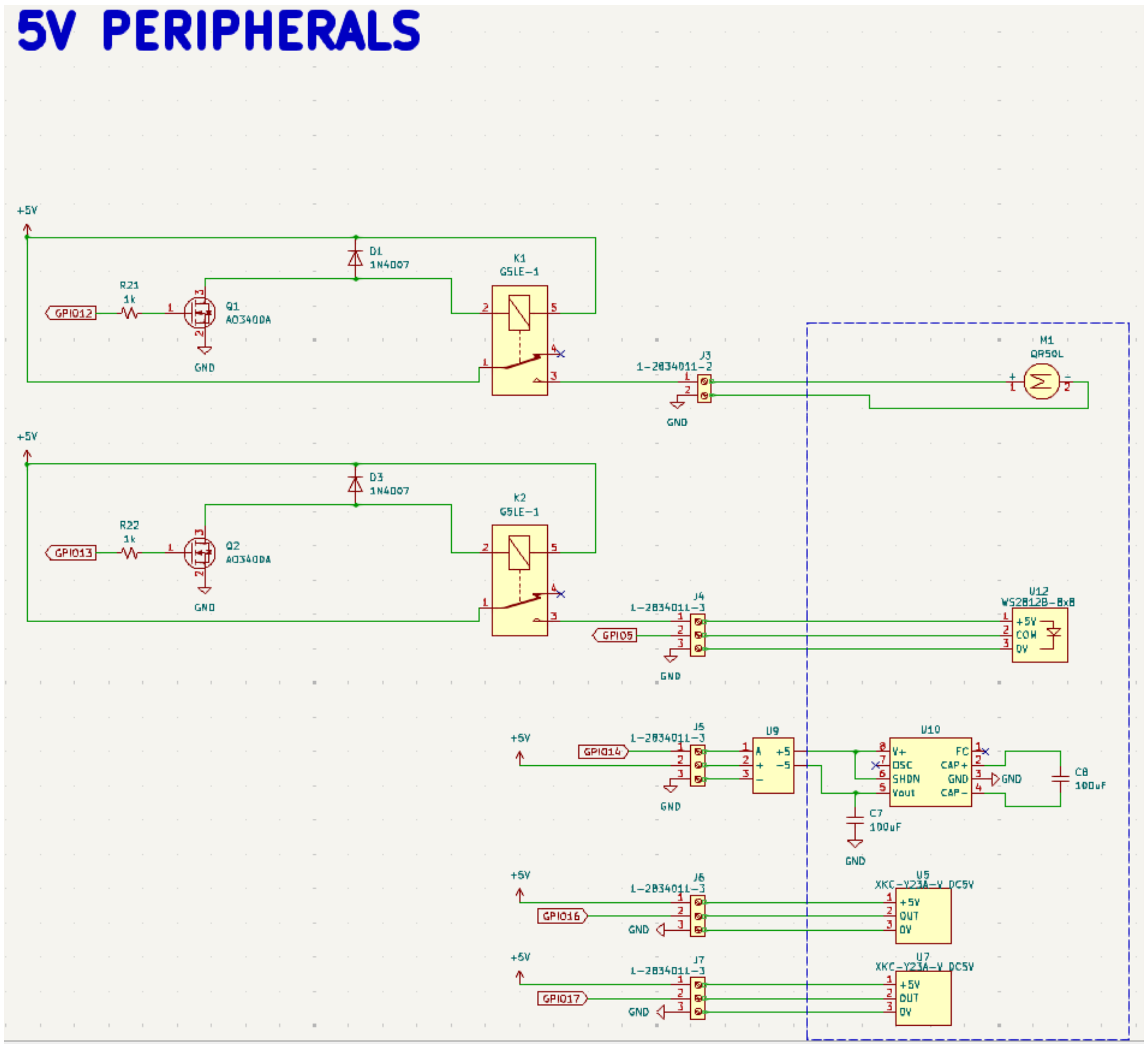


Figure 6.14. SD2 Updated 5V Peripherals Schematic

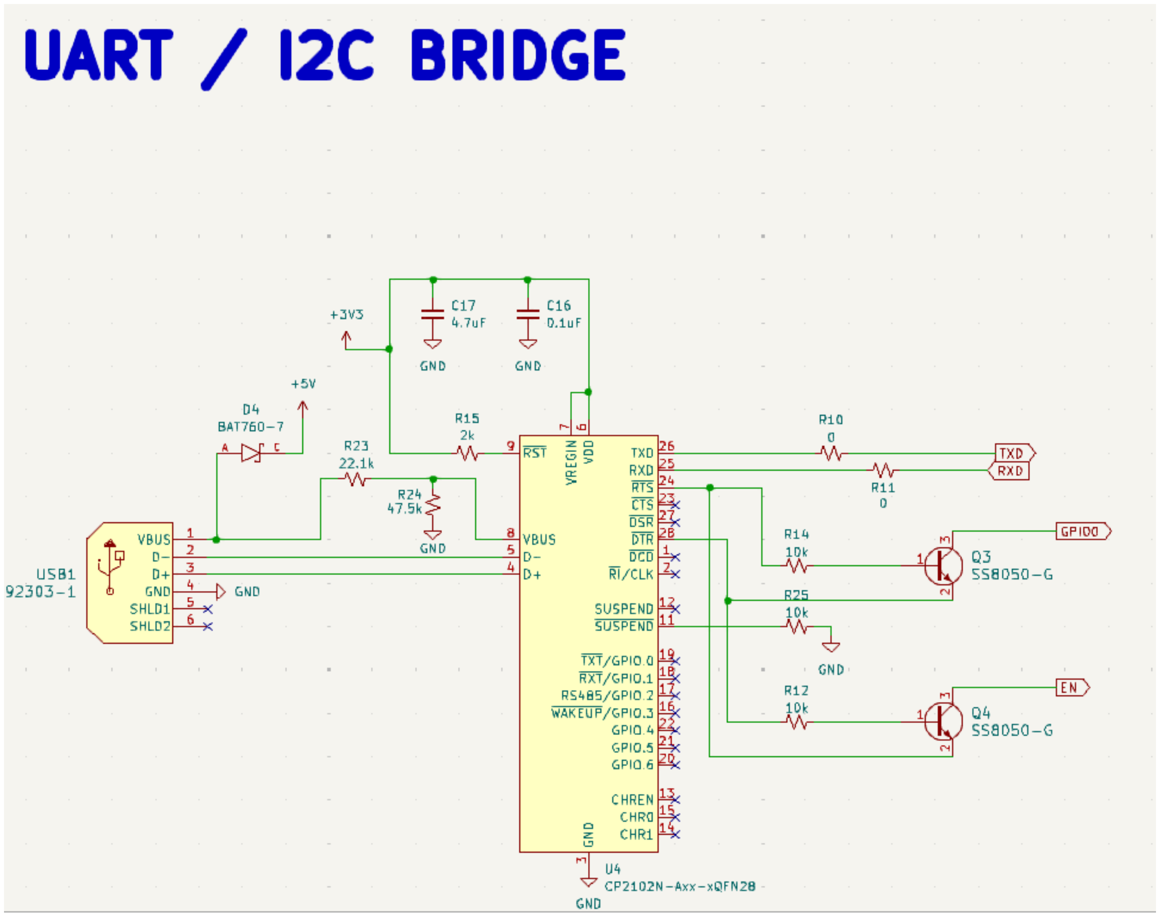


Figure 6.15. SD2 Updated UART / I2C Bridge Schematic

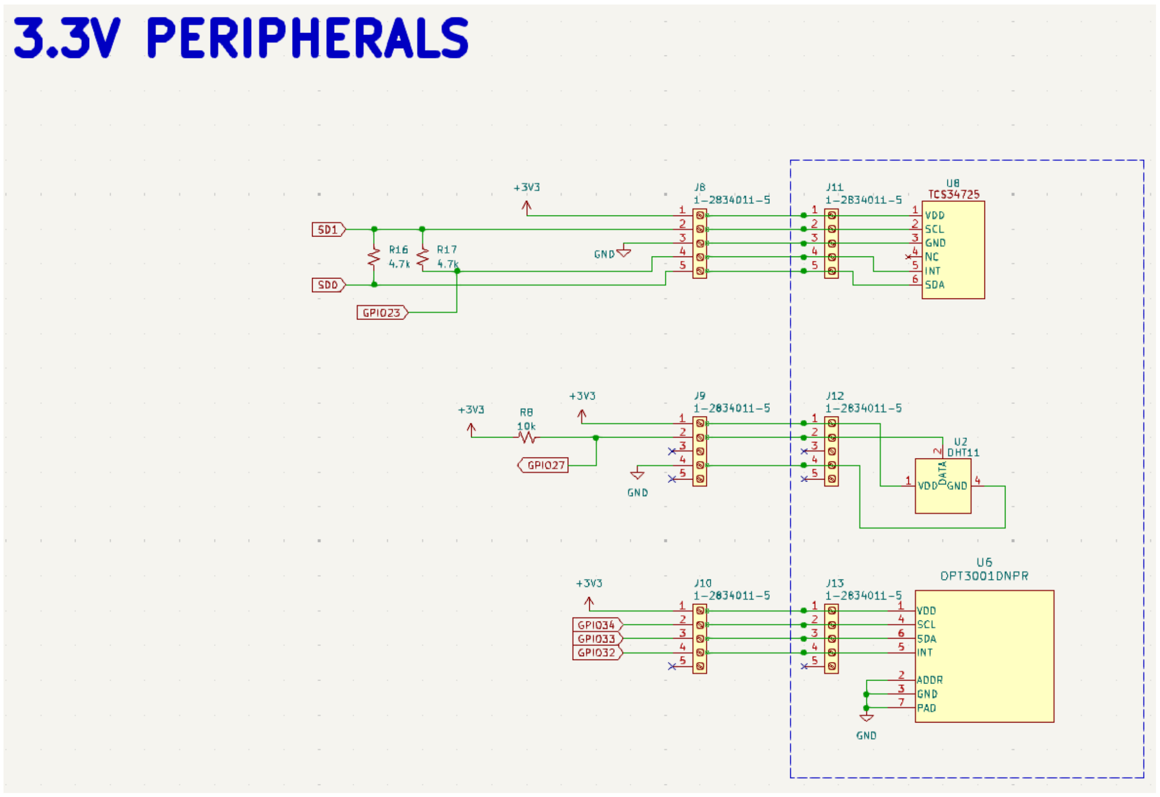


Figure 6.16. SD2 Updated 3.3V Peripherals Schematic

6.3 Design of Optical Components

The integration of advanced optical components is crucial for the successful operation of our project. These components form the backbone of our system’s ability to create and maintain an optimal environment for plant growth. By incorporating a sophisticated lighting system, precise color sensing, and reliable ambient light detection, we can ensure that the plants receive the appropriate light spectrum and intensity throughout their various growth stages. The sophisticated lighting system utilizes WS2812B RGB LEDs, which can be finely tuned to provide specific wavelengths of light that are essential for different phases of plant development, from seedling to full maturity.

This section outlines the design and functionality of these optical components in detail, highlighting how they work together to monitor plant health, adjust lighting conditions, and notify the system owner of any potential issues. The TCS34725 color sensor plays a pivotal role in detecting subtle changes in leaf coloration, which can indicate nutrient deficiencies or other health issues. Meanwhile, the OPT3001 ambient light sensor ensures that the system accurately measures the ambient light levels to prevent under or overexposure. These sensors feed real-time data to the control unit, which then makes dynamic adjustments to the lighting system to ensure plants receive consistent and optimal light conditions.

Furthermore, this integrated approach not only enhances plant growth but also simplifies the management process for users. The system is designed to be highly responsive, automatically adjusting to the plants' needs and providing alerts to the owner when manual intervention is required. This could include notifications about light levels falling outside the optimal range, prompting the user to check for potential power failures or LED malfunctions. By creating a responsive and efficient environment that promotes robust plant growth, we aim to reduce the time and effort required for plant care, making it more accessible for users, regardless of their expertise in gardening or hydroponics. This comprehensive optical system thus ensures that users can enjoy healthy, thriving plants with minimal hassle.

6.3.1 LED Lighting System

Component: BTF-LIGHTING WS2812B RGB 5050SMD LED
The BTF-LIGHTING WS2812B RGB 5050SMD LED is a high-performance, individually addressable LED module. It offers a wide range of colors and brightness levels, making it ideal for providing the precise lighting conditions needed for different stages of plant growth. The LEDs are capable of emitting light across the full spectrum, which is essential for promoting photosynthesis and healthy plant development.

Design and Placement: To ensure optimal lighting conditions, the LEDs will be strategically positioned around the growing area to provide even light distribution.
The design will include:

- **Arrangement:** The LEDs will be arranged in a grid pattern above the plants, ensuring that each plant receives an equal amount of light.
- **Distance:** The LEDs will be placed approximately 12 inches above the plants to provide sufficient light intensity without causing heat damage. This distance may be adjusted based on the specific light requirements of the basil plants at different growth stages.

Operation and Control
The LED lighting system will be dynamically controlled to match the nutritional needs and growth stages of the basil plants. The control mechanisms include:

- **Intensity Adjustment:** The intensity of the LED light will be adjusted based on sensor data from the TCS34725 color sensor. For example, if the sensor detects signs of nutrient deficiency that indicate a need for more light, the system will increase the LED intensity.
- **Wavelength Control:** The system will adjust the wavelengths of light emitted by the LEDs to match the growth stage of the plants. During the

vegetative stage, blue light (wavelengths between 450-480 nm) will be emphasized to promote healthy leaf growth. During the flowering stage, red light (wavelengths between 620-750 nm) will be increased to encourage flowering and fruiting.

- **Automation:** The LED system will be integrated with the main control unit, allowing for automated adjustments based on real-time data from the sensors. This ensures that the plants receive optimal lighting conditions continuously, even in the absence of manual intervention.

Our LED provides a versatile and efficient lighting solution for our system. By carefully designing the placement and control mechanisms of these LEDs, we can ensure that the basil plants receive the precise light conditions needed for optimal growth and health.

6.3.2 Color Sensor

Component: TCS34725 light sensor
The TCS34725 light sensor is an advanced color sensing module that features a digital RGB color sensor with an IR filter. It is designed to detect the color and intensity of ambient light and output this data digitally via an I2C interface. The sensor's high sensitivity, wide dynamic range, and ability to filter out infrared light make it ideal for precise color measurements in a variety of lighting conditions.

Design and Placement: The TCS34725 sensor will be strategically placed within 6 inches of the basil plant leaves. This proximity ensures accurate detection of subtle color changes that indicate the health and nutrient status of the plant. The sensor will be mounted on a stable platform to maintain a consistent distance from the leaves and to minimize interference from ambient light sources.

Mounting Considerations:

- **Stable Platform:** Ensures the sensor remains at a fixed distance and angle relative to the plant leaves.
- **Shielding from Ambient Light:** Use of a hood or enclosure to minimize external light interference and improve measurement accuracy.
- **Maintenance Access:** Design the mounting to allow easy access for cleaning and calibration of the sensor.

Operation and Data Interpretation:
The TCS34725 operates by emitting light from an onboard LED and measuring the reflected light from the plant leaves. It captures red, green, blue, and clear (no filter) light components, which are then processed to determine the precise color

of the leaves. The sensor's data is sent to the control unit via the I2C interface, where it is analyzed to detect signs of nutrient deficiencies or other health issues.

Mechanism:

- Light Emission and Detection: The sensor emits light and captures the reflected light in RGB and clear components.
- Data Processing: The raw data is processed to compute the color values and detect any deviations from the expected healthy leaf color.

Data Interpretation:

- Yellowing Leaves: Detected as a decrease in the blue and green components, indicating potential nitrogen deficiency.
- Purple Leaves: An increase in the red component, suggesting phosphorus deficiency.
- Brown Leaves/Edges: Identified by specific patterns in the RGB values, pointing to potassium deficiency.

Integration with Control System

The color sensor is integrated into our control system, which continuously monitors the sensor data and adjusts the LED lighting accordingly. The system uses predefined algorithms to interpret the sensor data and make real-time adjustments to the lighting conditions to address the detected deficiencies.

Control Logic:

- Adjusting LED Intensities: Based on the detected deficiencies, the system increases or decreases the intensity of specific wavelengths (e.g., more blue light to address nitrogen deficiency).
- Feedback Loop: The system periodically re-evaluates the leaf color to ensure the adjustments are having the desired effect.

User Notifications:

In addition to automatic adjustments, the system notifies the plant owner of detected issues via a user interface. This interface provides detailed information about the detected deficiencies and recommended actions to further support plant health.

Notification Features:

- Real-Time Alerts: Notifications sent to the user’s device when a deficiency is detected.
- Detailed Reports: Periodic reports summarizing the health status of the plants and the actions taken by the system.

- User Interface: A dashboard that displays sensor readings, historical data, and recommended actions.

6.3.3 Photoresistor

Component: NOYITO OPT3001 Ambient Light Sensor Measurement Light Intensity Single Chip Illuminometer.

Function: The OPT3001 photoresistor used to detect ambient light levels, ensuring that the LED lights are functioning correctly. It measures light intensity accurately, providing data essential for maintaining optimal lighting conditions for the basil plants.

Design and Placement: The OPT3001 sensor will be strategically placed within the growing area to detect light emitted by the LED system. It will be positioned in such a way that it can accurately measure the light intensity received by the basil plants.

Considerations:

- **Mounting:** The sensor was mounted securely to avoid any movement that could affect its accuracy.
- **Orientation:** It was oriented to avoid direct exposure to external light sources, ensuring it only measures light from the LEDs.
- **Shielding:** Shielding was used to block ambient light from outside sources, ensuring the sensor measures the intended light.

Mechanism:

- The OPT3001 sensor measured light intensity and converted it into digital data that the system can read and analyze.
- It operated by detecting the ambient light levels in the growing area, providing real-time data on the presence and intensity of light.

Failure Detection:

- The sensor continuously monitors the light levels, and any significant drop in intensity that deviates from the expected range indicates a possible power failure or malfunction of the LED system.
- In the event of a detected anomaly, the system will trigger an alert to notify the plant owner.

Integration with Notification System:

- **Data Processing:** The data from the OPT3001 sensor was sent to the central control unit, where it was processed and compared against predefined thresholds.
- **Alert System:** If the light intensity fell below the threshold, indicating that the lights were off or malfunctioning, the system sent an alert to the plant owner. This is done via email, SMS, or a notification through a dedicated mobile app.
- **User Notification:** The notification will included information about the detected issue and recommended actions, such as checking the power supply or inspecting the LED system.

Example Workflow:

1. **Light Detection:** The OPT3001 sensor continuously measured the ambient light intensity.
2. **Data Transmission:** The sensor sent real-time data to the control unit.
3. **Data Analysis:** The control unit analyzed the data to ensure light levels were within the optimal range.
4. **Failure Detection:** If the light levels fell below the expected range, the control unit identified a potential failure.
5. **Alert Generation:** An alert is generated and sent to the plant owner, detailing the issue and suggesting corrective actions.
6. **User Action:** The plant owner received the alert and took necessary steps to resolve the issue, ensuring the basil plants received adequate light.

Benefits:

- Reliability: The OPT3001 sensor ensured reliable monitoring of light intensity, crucial for maintaining optimal growth conditions for the basil plants.
- Timely Alerts: Early detection of lighting issues allowed for timely intervention, preventing potential damage to the plants due to inadequate lighting.
- User-Friendly: Integration with a notification system ensures that the plant owner is always informed about the status of the lighting system, even remotely.

The NOYITO OPT3001 Ambient Light Sensor played a critical role in our system by providing accurate light intensity measurements and ensuring the LED lights function properly. Its integration into the system enhances the reliability and efficiency of the hydroponic setup, ultimately contributing to the healthy growth of basil plants.

6.3.4 Integration and System Workflow

Data Flow and Communication

Data Collection:

- **Color Sensor** (TCS34725): The TCS34725 collects color data from the basil leaves, detecting variations that indicate nutrient deficiencies. This sensor operates within 6 centimeters of the plant leaves to ensure accurate detection.
- **Ambient Light Sensor** (OPT3001): The OPT3001 monitors the ambient light intensity to ensure the LED lights are functioning properly.
- **LED System** (WS2812B RGB 5050SMD): The LED system's operational status and intensity levels are monitored and adjusted based on sensor inputs.

Data Transmission:

- All sensor data is transmitted to a central control unit (e.g., a microcontroller such as Raspberry Pi or Arduino).
- The control unit processes the data in real-time, using predefined algorithms to interpret the sensor readings.

Communication Protocols:

- **I2C Protocol:** Utilized for communication between the sensors and the control unit, ensuring efficient and reliable data transfer.
- **SPI Protocol:** Used for communication with the LED system, allowing precise control over light intensity and color adjustments.

Algorithm for Adjusting LED Intensities:

- **Sensor Data Analysis:** The control unit analyzed the color data from the TCS34725 sensor to identify nutrient deficiencies. For example, yellowing leaves may indicate a nitrogen deficiency.
- **Light Intensity Adjustment:** Based on the growth stage and detected deficiencies, the control unit adjusts the LED light intensity and color spectrum. For instance, blue light (450-480 nm) promotes vegetative growth, while red light (620-750 nm) encourages flowering.
- **Feedback Loop:** The system continuously monitors the light intensity using the OPT3001 sensor. If the light levels fall outside the optimal range, the control unit recalibrates the LED settings to maintain consistent lighting conditions.

Workflow for Detecting and Notifying Power Failures:

- **Light Detection:** The OPT3001 sensor detects the ambient light levels in the growing area.
- **Data Comparison:** The control unit compares the current light intensity with predefined thresholds. A significant drop indicates a potential power failure or LED malfunction.
- **Alert Generation:** If a failure is detected, the control unit generates an alert, which is sent to the plant owner via email, SMS, or a dedicated mobile app.
- **User Notification:** The plant owner receives the alert with detailed information about the issue and suggested corrective actions (e.g., checking the power supply).

Data Presentation:

- **Real-Time Monitoring:** The user interface provides real-time updates on the status of the lighting system and plant health metrics.
- **Historical Data:** Users can access historical data to track the progress of their plants and make informed decisions about nutrient and lighting adjustments.

Control Features:

- **Manual Override:** Users can manually adjust the LED settings through the interface if needed.
- **Notification Management:** Users can customize notification preferences to receive alerts via their preferred communication channels.

Interface Design:

- **Web Application:** A user-friendly web application that displays sensor data, system status, and alerts.
- **Mobile App:** A mobile application that offers remote monitoring and control capabilities, ensuring users can manage their system from anywhere.

Integrating the TCS34725 color sensor, OPT3001 ambient light sensor, and WS2812B LED system into our project created a cohesive and responsive hydroponic environment. The detailed data flow and communication protocols ensure that all components worked harmoniously to maintain optimal growing conditions. The control logic and user interface enhance user experience by providing real-time insights and remote-control capabilities, ensuring the health and growth of basil plants are consistently monitored and managed.

6.3.5 Takeaways

The integration of advanced optical components, including the TCS34725 color sensor, the OPT3001 ambient light sensor, and the WS2812B RGB LED system, has significantly enhanced the functionality and efficiency of our project. These components worked in tandem to create a responsive and adaptive environment that ensures optimal growing conditions for basil plants.

Enhanced Monitoring and Control:

- The TCS34725 color sensor provided precise detection of leaf color changes, enabling early identification of nutrient deficiencies and allowing timely interventions.
- The OPT3001 ambient light sensor ensured continuous monitoring of light intensity, detecting power failures or LED malfunctions and notifying the plant owner promptly.

Adaptive Lighting System:

The WS2812B RGB LED system dynamically adjusted light intensity and spectrum based on the plant's growth stage and detected deficiencies, promoting healthy growth and maximizing yield.

Efficient Data Flow and Communication:

Utilizing I2C and SPI communication protocols ensures reliable and efficient data transfer between sensors and the control unit, facilitating real-time adjustments and monitoring.

User-Friendly Interface:

The integration of a web and mobile application provides users with real-time updates, historical data tracking, and remote-control capabilities, enhancing the overall user experience and ensuring that plant care can be managed effectively, even from a distance.

Benefits of the Integrated System:

- **Reliability:** The combined use of these advanced sensors and LEDs ensures a robust system that can consistently maintain optimal growing conditions, reducing the risk of plant stress or failure.
- **Scalability:** The modular design allows for easy expansion or modification, making it adaptable to different types of plants or larger growing setups in the future.

User Empowerment: By providing detailed insights and control options, the system empowers users to make informed decisions and take proactive measures to ensure the health and productivity of their plants.

Wrap Up:
The integration of the TCS34725 color sensor, OPT3001 ambient light sensor, and WS2812B LED system within our project demonstrated the potential of advanced optical components to transform indoor gardening. By leveraging these technologies, we have created a smart, responsive, and user-friendly hydroponic system that ensures optimal growth conditions for basil plants, paving the way for future innovations in automated plant care. With the addition of our other sensors and possibly an electromagnetic field generator, the user is guaranteed a product more customizable and growth-focused than any other current product on the market.

Chapter 7 - Software Design

Software Design is an integral part of this project, from the low level C and C++ code in the MCU, to the higher level JavaScript to power the web application, without the software design, the autonomy we are aiming for would be impossible to achieve. The standard we've set out for the software design of this project allows for advanced communication between an MCU, a web server using NodeJS deployed on AWS Lightsail, and a database deployed on the AWS cloud.

7.1 Creating a Cloud Instance

As stated prior, we are leveraging AWS Lightsail to deploy our application on the web, when creating an instance in Lightsail there are a multitude of options to choose from, we decided to go with a Linux deployment supported by Bitnami for a MERN stack web application. Though for a MERN stack, they supplied us with a MongoDB database instance, that is not the database type this project will be using; AWS Lightsail also provides the ability to create a standalone database instance, for this project we created a MySQL instance from Lightsail.

AWS provides a CLI within its application that allows for scripting within the virtual Linux machine, with Bitnami a lot of our application is out of the box supported, with a public IPv4 address (54.234.247.131), and a MongoDB instance. However we changed the database reference to the MySQL database instance instead, this was not hard to achieve even though the MongoDB was the out of the box option, a clever option was to just use the AWS CLI to SSH into the MySQL server instance we created through AWS Lightsail as well.

AWS allows for containerizing a web server in an EC2 container possibly within an S3 bucket, however, we have opted out of this option because of the complexity and possible price increases it may bring to this project. Though deploying in an EC2 instance might be good for elastic growth for this project, for now we don’t foresee a huge amount of growth that would need automatic growth, ultimately, we can choose to manually grow into a more data centralized plan if needed.

7.2 Frontend Design

The frontend design taking place will be as simple as possible for the user, the goal was to leverage it as a single page application supported by the AngularJS framework alongside the Bootstrap design library. The application landing page is a user log in, supported by an HTTP login endpoint with encrypted credentials created from a hash function; this provides a safe and secure environment for our users. The user also has what is called a JSON Web Token (JWT), this token will leverage every endpoint to determine if the token is still active, if not, then the endpoint will throw an error. After logging in, the application navigates to the main page.

From here, the user has a multitude of options, one being the ability to use ChatGPT to analyze the plants current data and get suggestions from ChatGPT on how to better take care of their very own plant. There will also be 4 different options for user input, a water pump, artificial sunlight, EMF generator, and a diagnostic LED, the user will be able to adjust all of these values to whatever they may want, but ideally they’ll use the ChatGPT analysis to make changes based upon that. The main page presents data in a digestible and pleasing way for our users, the goal is to make it a one stop shop for all concerns they may have with their plant, and we want our UI to reflect those needs. The user also has a logout feature available to them, which allows them to sign out of the application for security purposes, however there will be a timeout feature, if the user is inactive for > 20 minutes the user will be automatically logged out for their own safety.

When designing the frontend, the JavaScript framework Angular is used, the reason we leverage Angular is that it excels in single page applications and load times, it renders everything on page load so updates are fluid and seamless, this fits our goal of having a fast update turnaround for the user input of our application. The Angular CLI (Command Line Interface) also empowers our fast development time, the CLI creates and manages component layout and folder layout so developers don’t have to.

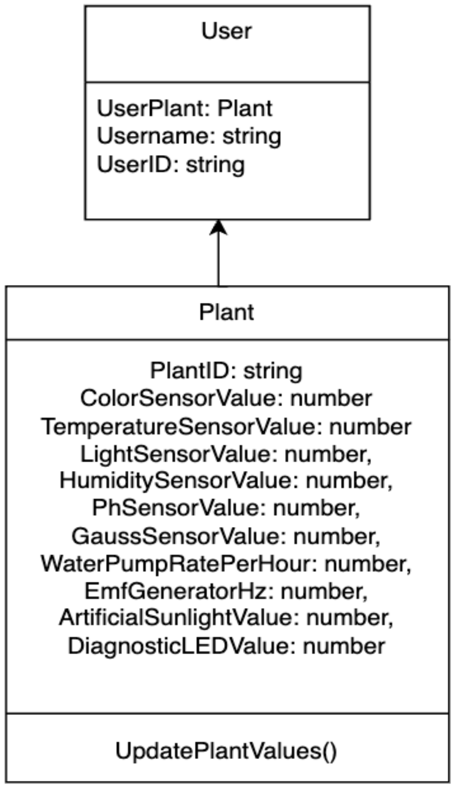

User
UserPlant: Plant
Username: string
UserID: string
Plant
PlantID: string
ColorSensorValue: number
TemperatureSensorValue: number
LightSensorValue: number,
HumiditySensorValue: number,
PhSensorValue: number,
GaussSensorValue: number,
WaterPumpRatePerHour: number,
EmfGeneratorHz: number,
ArtificialSunlightValue: number,
DiagnosticLEDValue: number
UpdatePlantValues()

Figure 7.1. Frontend Design

7.3 Connecting Microcontroller to the Database

To enable a microcontroller to communicate with a database, we needed to establish a system that allows data to be sent to and received from the microcontroller over a network. This involved setting up the web server, configuring the microcontroller in an arduino environment to connect to the wifi, and creating communication protocols for data exchange.

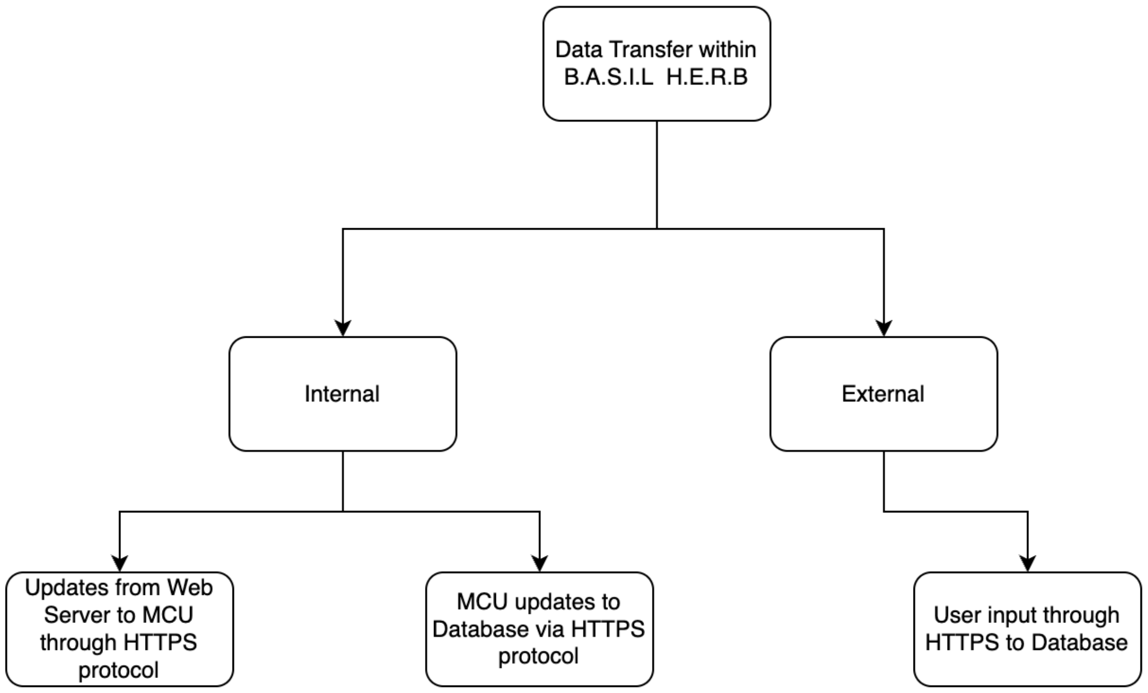
The libraries WiFi.h and WebServer.h are crucial for this process. WiFi.h allows the microcontroller to connect to a Wi-Fi network which enables the necessary internet connectivity. WebServer.h is what allows the microcontroller to run a web server giving it the ability to handle incoming HTTP requests and sending responses.

The web server will be built using a popular backend framework called Express in Node.js. This serves as the intermediary that processes incoming requests and sends responses. It will host a web interface where the users would be able to input commands or view the data being extracted from our sensors. Using the ESP-WROOM-32D we can utilize its internet connectivity to host a web server and handle HTTP requests from the Node.js server.

Since we are using an AWS Lightsail server, it provides the ability to control our ESP32 microcontroller remotely, eliminating the need to be on the same Wi-Fi network. By hosting the Node.js server on AWS Lightsail, we can create a central point of communication that is accessible over the internet. By leveraging the public IP address or a domain name associated with the AWS Lightsail instance, we can ensure secure and reliable remote access to our microcontroller which enables flexible and convenient control of our design from anywhere.

The process of sending data to the microcontroller will be handled by utilizing HTTP requests. The microcontroller will receive these requests, process the data, and then perform the required actions. Conversely, the microcontroller will be able to send an acknowledgement back to the server which is what allows for real-time monitoring and control via the web interface. This bi-directional communication ensures that the microcontroller can both report its status and be controlled remotely.

An example of how this whole process would work is the user would access the web page hosted by a Node.js server and input the desired frequency, such as 1000 Hz, for the LED lights. The web page would send this frequency value to the Node.js server, which constructs an HTTP POST request and sends it to the ESP-WROOM-32D's web server. The ESP32 would then receive the request, extract the frequency value, and adjust the PWM signal to set the LED lights to the new frequency. Lastly, it would send a confirmation back to the Node.js server and update the web page to inform the user that the frequency change was successful.


Data Transfer within B.A.S.I.L H.E.R.B
Internal
External
Updates from Web Server to MCU through HTTPS protocol
MCU updates to Database via HTTPS protocol
User input through HTTPS to Database
Figure 7.2. Data Transfer

7.4 Backend Design

When designing the backend for this project, we have decided to go with the ExpressJS framework which is hosted on a NodeJS server, this is supplied by the AWS Lightsail MEAN stack deployment method that provided us with the out of the box solution for our stack. On our code base, we are developing in a Node server environment and using ExpressJS which would make this compatible with what our cloud servicer has provided.

The backend service would handle all data cleansing from possible fraudulent users before going to the database or microcontroller, for the microcontroller this would be done by confirming the data inputted by the user would strictly be an integer, this would help protect our microcontroller from unauthorized users and possible injection attacks. Unauthorized commands from unsanitized user input could affect our microcontroller in many ways, for example is a command injection, which is a type of attack where a malicious user inserts commands for the microcontroller which could cause unintended actions by the microcontroller.

There has to be another layer of security design as well for the database, which is notorious for being susceptible to SQL injection attacks. SQL injection attacks can be prevented by data cleansing, with the table designs we have in mind, the only threat of SQL injection we face, is going to be user input of microcontroller values that will be sent to a MySQL database for logging and presentational purposes for our web application. By far the safest way to prevent SQL injection attacks is by data sanitation, which can have and for this project will have multiple layers.

The first layer is input sanitation, which will include ensuring the data sent from the front end HTTP call will be what is expected, for this project we are expected a number type in TypeScript or an integer, if we get an unexpected type like a string value for example, we will throw a 400 error (bad request). If any special characters come through, for example, a * is an obvious choice for SQL injection attacks, we will throw a 400 error back to the front end to show the user.

The second layer of defending against SQL injection is the use of SQL parameters, when creating custom SQL statements to be used in our database, we could possibly use standard string concatenation of a SQL statement. However, this allows for possible SQL injection because when using string concatenation MySQL does not know the difference between variable and data, so it considers everything within that statement executable. When using SQL parameters, MySQL understands what is considered a variable and can consider that value a variable, which cannot be executed on its own (which SQL injection can do). This is one of the safest ways to prevent SQL injection and a great practice for our web application.

The backend infrastructure will consist of a collection of endpoints that support parts of C.R.U.D (Create, Read, Update, Delete) operations, the create operation will drive the first initial updates the user makes to default values to the microcontroller sensors, every iteration or update after that will be driven by the update portion of the C.R.U.D operations, deletion will be made possible by setting the microcontroller values back to default. Read operation will be triggered after the login endpoint is successful and will allow the user to see the values of the sensors. Outside of C.R.U.D operations, there will be a GET endpoint to perform the data analysis using ChatGPT, and a login endpoint to ensure the user is an authorized user that can perform these updates.

7.5 Database Design

For this project, we decided to use MySQL database structure. MySQL is a relational database with a row and column data structure system, for this project there are at least three tables we will be using.

- The user table held the usernames and passwords securely for the application. To maintain user security, we leveraged a hashing algorithm library in our backend API, hashed the values, and stored them safely in the database. This prevented user data from being exploited even if malicious users got into our database.

Table 7.1. Design of User Table

UserID	Username	HashedPassword	PlantID
int	varchar(50)	varchar(250)	int

- **Measurement Table:** This table held the measurement data that the MCU exported to the database. From this table, we showed the user the data from their plant. This could have included an over-time graph as well as the most recent data. Ideally, the columns represented the sensor values as well as the UTC time of when the data was inserted. This data was merely used for documentation and UI purposes and was not updated by our backend.

Table 7.2. Design of Measurement Table

McuID	PlantID	Humidity	Temperature	pH	Light	WaterLevel	Color
int	int	float	float	float	int	float	int

- **Threshold Table:** This table initially held the default values of the sensors and was shown to the user through the UI. It also held the changes the authorized user made to these sensors; for example, the rate of circulation of the water pump could be changed. This table acted as both a logger and a tool used in our UI that was updated frequently. These values were only inserted or updated into this table once an ACK requirement was received from the MCU after the values had been updated.

Table 7.3. Design of Threshold Table

PlantID	WaterRate	EmfValue	DiagnosticLED
int	float	float	int

These three tables enabled us to achieve our goal of maintaining simplicity in terms of data storage and data integrity. Since we used only three tables, all structured to be lightweight, this allowed us to remain scalable. With the structure we had, it did not limit us from pursuing more opportunities for growth, nor were we tied down to a specific structure. Using MySQL, it supported a data integrity model but, within limits, allowed growth at a moderate level. MySQL also revolved around relationships between entities; for our project, there were only two entities, the user and the plant, with the user having a plant attribute. Maintenance of the database was limited as well, considering how small the data set was. We hoped to possibly implement triggers that might update a backup table or even have a logging database, though that might have been out of our price range depending on activity.

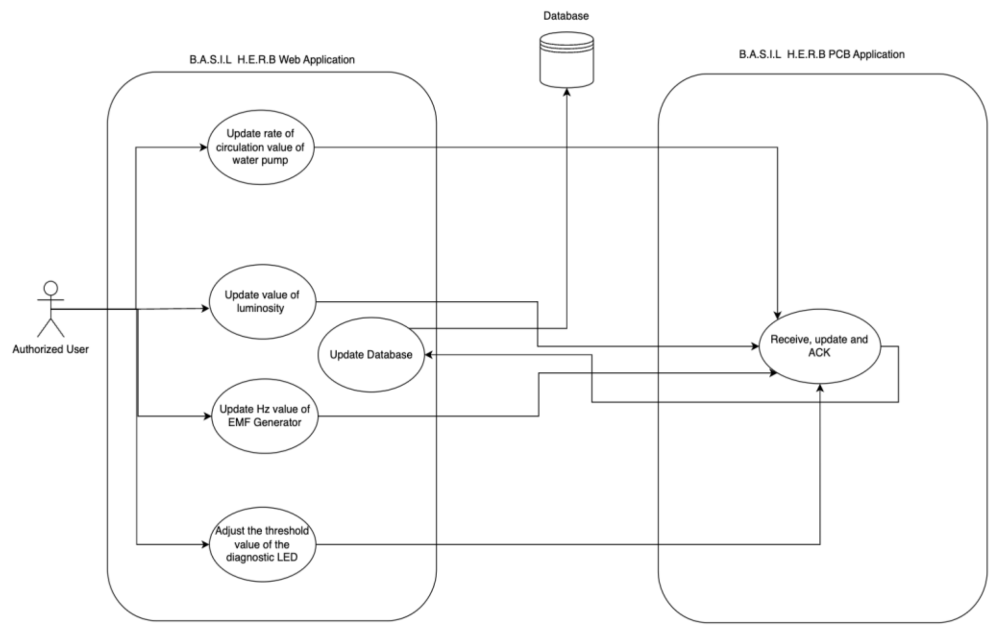

Database
B.A.S.I.L H.E.R.B Web Application
B.A.S.I.L H.E.R.B PCB Application
Update rate of circulation value of water pump
Update value of luminosity
Update Database
Update Hz value of EMF Generator
Adjust the threshold value of the diagnostic LED
Authorized User
Receive, update and ACK

Figure 7.3. Case Diagram For Web Application

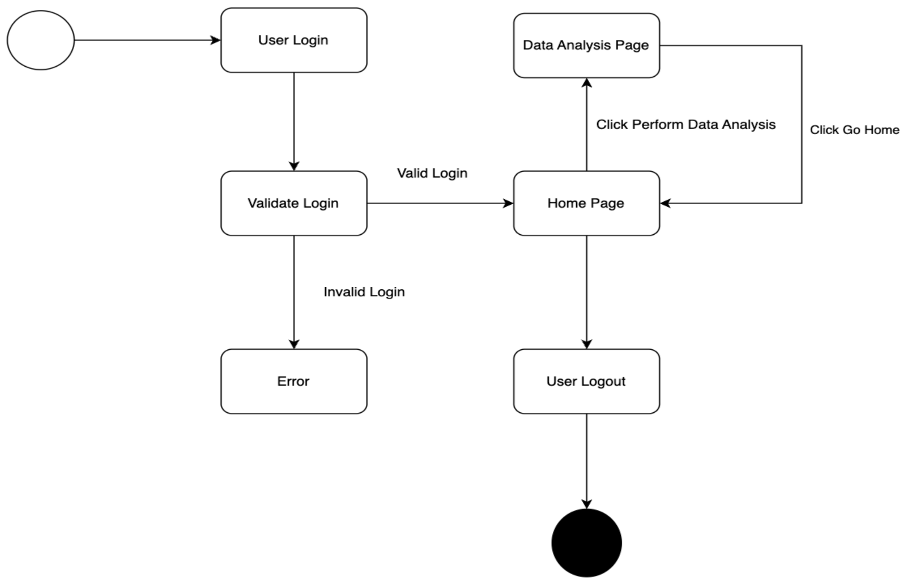


Figure 7.4. State Diagram of Web Application

7.6 Leveraging OpenAI's ChatGPT

For this project, we sought to leverage an AI/ML model to perform data analysis on plant data. To achieve this, we leveraged OpenAI's ChatGPT model. The option of creating our own AI sounded like an incredible goal, but one we opted to avoid due to time concerns. As ChatGPT had undergone multiple iterations and versions, each more powerful than the last, we were confident it could provide accurate and surefire results for us. ChatGPT also conveniently had a public API, which developers like us could use for free or at very low cost (for smaller projects like ours). For ChatGPT, there was possible billing; when using their API, we were limited to 40,000 tokens per minute or 200,000 tokens per day for ChatGPT 3.5 Turbo. This was a free tier and one we most likely stayed within for this project.

When using the open API, we first installed the OpenAI library from the NodeJS library using the command npm install --save openai. This allowed us to use the OpenAI API framework and responses. The return of the GET request was an array of responses, with index 0 being the response to the first prompt, up to n responses.

Chapter 8 - System Fabrication & Prototype Construction

In this chapter we detail the system fabrication and prototype construction of our project. This chapter focuses on the PCB layout and the steps involved in assembling the prototype. It covers the design and implementation of the printed circuit boards, detailing the placement of components and routing of electrical connections. The chapter also describes the assembly process, including the soldering of components and integration of the PCB into the overall system. Through these detailed descriptions, readers will gain insights into the practical aspects of bringing our system from design to a functional prototype.

8.1 PCB Design

This is an integral part for our project as per the guidelines for Senior Design. We are required to fully design our own customized PCB to exactly fit the needs of our project's components.

8.1.1 PCB Design Process

The PCB design process is reliant on KiCAD software for generating the design schematic drawings and PCB layouts drawings and additionally manufacturing files called gerber files were generated and sent to a PCB manufacturer for quote and creation.

8.1.2 PCB Schematic Design Symbols

KiCAD comes with a large symbol library out of the box but there were several parts that had to be sourced from third party websites or even created manually. The schematic symbols maintain an association with a part if assigned that can then be part of a generated BOM.

8.1.3 PCB Layout

The major components placed in the layout space include the MCU, two motor control relays, three voltage regulators, a temperature and humidity sensor, a status light, pin headers for connection to other devices and development boards, resistors, capacitors, and inductors. After completing the schematic, the workflow transitions from the schematic to the PCB layout phase.

In the PCB layout, each symbol in the schematic is linked to a physical footprint for either surface mount devices (SMD) or through-hole devices (THD), as well as a corresponding 3D model. This association helps in visualizing and optimizing the placement of components to maintain short, direct traces with minimal overlap. The schematic connections guide the user in arranging the components efficiently.

When routing the traces, if the wiring becomes congested, vias will be employed to route the trace to another layer of the PCB. Vias are small holes that allow electrical connections between different layers of the PCB. While some PCBs can have multiple layers, a standard design typically includes at least two layers: one for positive power supply and another called the ground plane.

Figures 8.1 and 8.2 illustrate the PCB layout process. Figure 8.1 shows the PCB workspace with a defined border, component footprints placed and connected via traces. Figure 8.2 presents the completed board with all devices installed or soldered in place. These figures highlight the transition from a digital layout to a physical board, ensuring each component is correctly placed and connected for optimal performance.

This detailed layout ensures that the PCB design is efficient and functional. The final product is a well-organized and reliable PCB ready for deployment in the system.

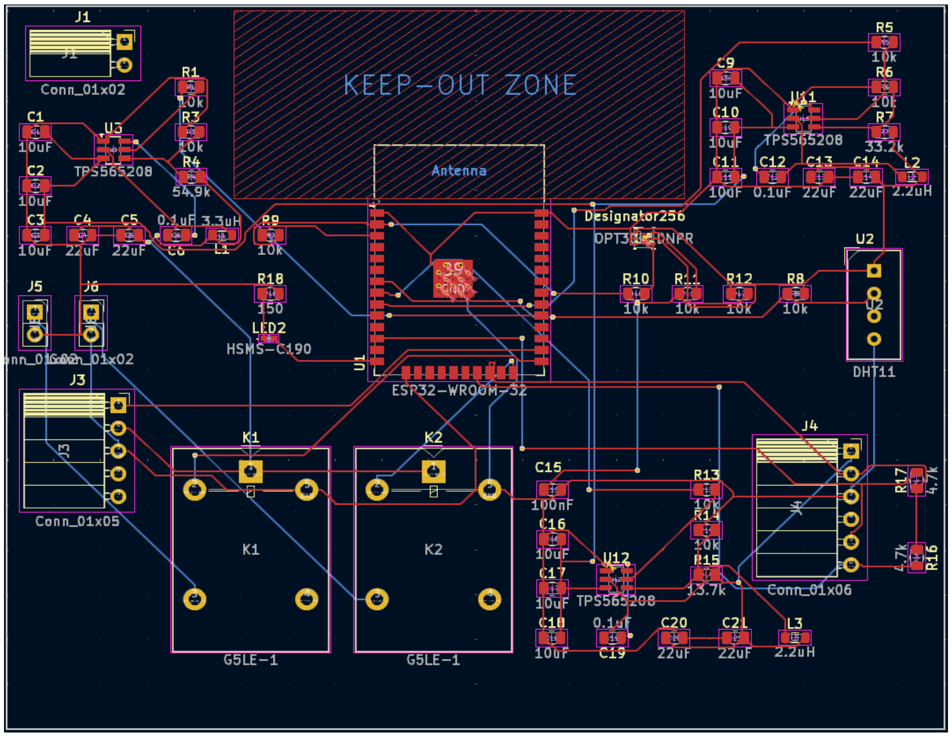


Figure 8.1. PCB Layout

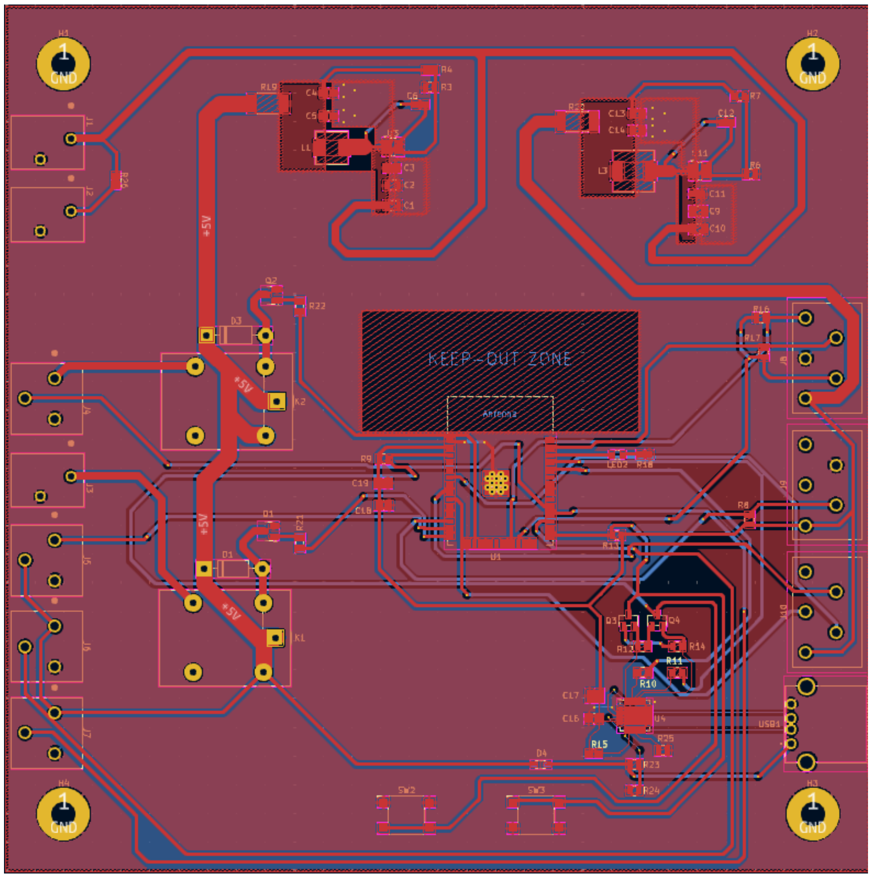


Figure 8.2. SD2 Updated PCB Layout

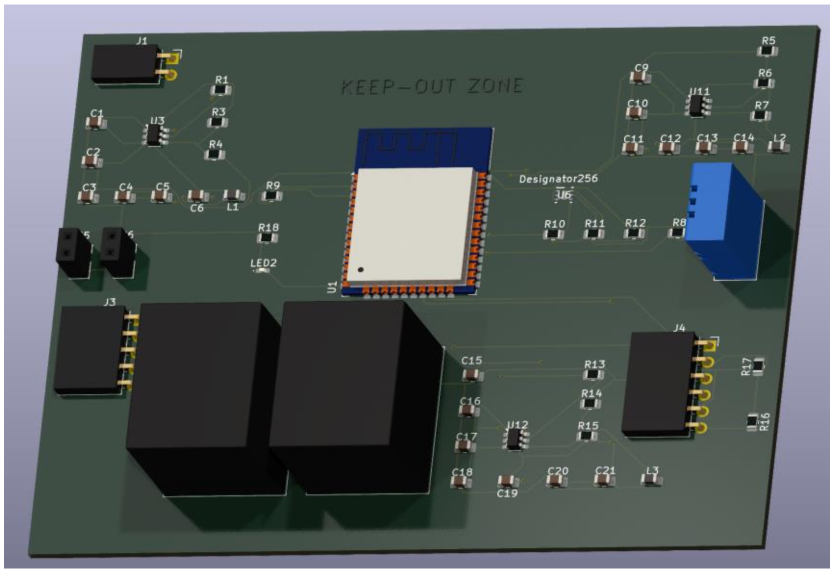


Figure 8.3. PCB Rendering

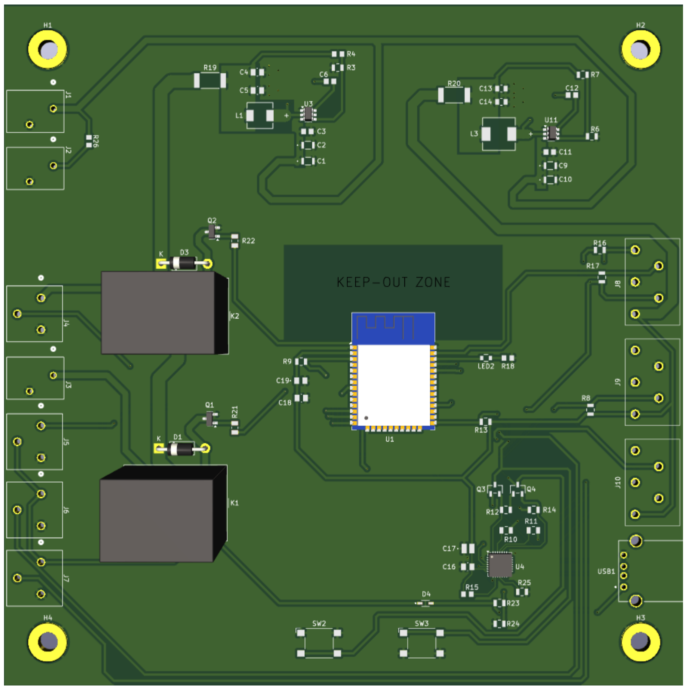


Figure 8.4. SD2 Updated PCB Rendering

Chapter 9 - System Testing and Evaluation

In this section, we delve into the comprehensive testing and evaluation process undertaken to ensure the reliability, accuracy, and effectiveness of the B.A.S.I.L. H.E.R.B. system. This process involves a series of rigorous tests on individual components, as well as the integrated system, to verify that all functionalities meet the specified requirements and performance standards.

9.1 Software Testing

Software testing encompasses a multitude of products in this project, it relates to both the C and C++ code of the MCU, the cloud deployment software to ensure the web application will be properly deployed to the web, and the actual web application code to ensure the web application runs effectively.

9.1.1 Web Application Frontend Testing:

The frontend testing for this application was done in a local environment by Casey. This was done in an AngularJS environment using the ‘npm start’ command, which started a local server and deployed to a localhost web application. This allowed us to have a local server that reflected the code on their current branch. Therefore, manual testing was conducted to ensure the UI reflected the changes made to the web application. It was also used to test the backend by hitting endpoints triggered by the frontend UI code. Unit testing in frontend design was an important concept to ensure expected results for our end users. That’s why we used a common AngularJS unit testing framework called “Mocha.” Mocha allowed for automated and asynchronous testing, integration with our NodeJS server, and integrated browser testing to emulate the end user.

9.1.2 Web Application Backend Testing:

Given that our backend was ExpressJS hosted on a NodeJS server, there were multiple backend testing opportunities that could be taken. First was ensuring the endpoints were functional, which could be achieved by using a service such as Postman. Postman was an API platform that could be used to test API service endpoints in a web application; we imitated a frontend by sending a payload to the endpoint. Postman received that data and returned a response, and from this response, we determined whether or not the endpoint was behaving as expected. Following the same trend as the frontend design, we continued to write unit tests and automated tests using Mocha, which allowed us to scale the backend as quickly as needed to keep up with the demand of development.

9.1.3 MCU Communication with Web Application Testing:

To ensure the MCU worked as intended in our design, we performed a multitude of tests. First, we connected the MCU to a Wi-Fi network, which allowed for communication with the web app. This was done through programming in the Arduino IDE. Once connected, it was essential to ensure the ESP32 could send HTTP POST requests to the AWS Lightsail server and handle responses correctly. This was verified through the receipts in the server logs.

Functional testing focused on ensuring the MCU could process commands from the web application and perform the intended actions. For example, Casey sent a request to change the daylight cycle of the LED lights. To test this, a command was sent from the web app, which communicated the change to the MCU and was verified through the serial output. Another key part of the testing involved assessing how the MCU handled stress. It needed to handle multiple requests simultaneously and operate without failure. This involved multiple users each sending commands and checking to see if the MCU handled them without crashing. Lastly, we tested to ensure the MCU could handle errors by sending invalid inputs. It rejected the requests and continued to function normally. All of this testing ensured the MCU remained reliable while performing the necessary tasks in our design.

9.1.4 Plan for SD2 Software Development:

All sensors in our system were functioning correctly and provided accurate readings. While investigating variations in our pH measurements, we discovered that these changes corresponded with a UCF water valve burst incident that temporarily affected the water supply's acidity levels during our testing period.

We had deployed a MySQL database via AWS Lightsail as well as a web server for a MEAN stack application deployed on the cloud via AWS Lightsail. We also had a code repository deployed on GitHub where the code included an AngularJS frontend with a Bootstrap library.

With this being said, the goal for SD2 was first to create the ExpressJS backend and start implementing the database tables design. Once that was completed, we then started designing the frontend. The biggest challenge there was creating an appealing UI while still holding value in terms of showing the user the plant's empirical data. The next step was to develop the endpoints for the ExpressJS backend, and we tested these endpoints to ensure we got expected results. Once the API layer was complete, we inserted test values into the database, completing the initial phase of the web application.

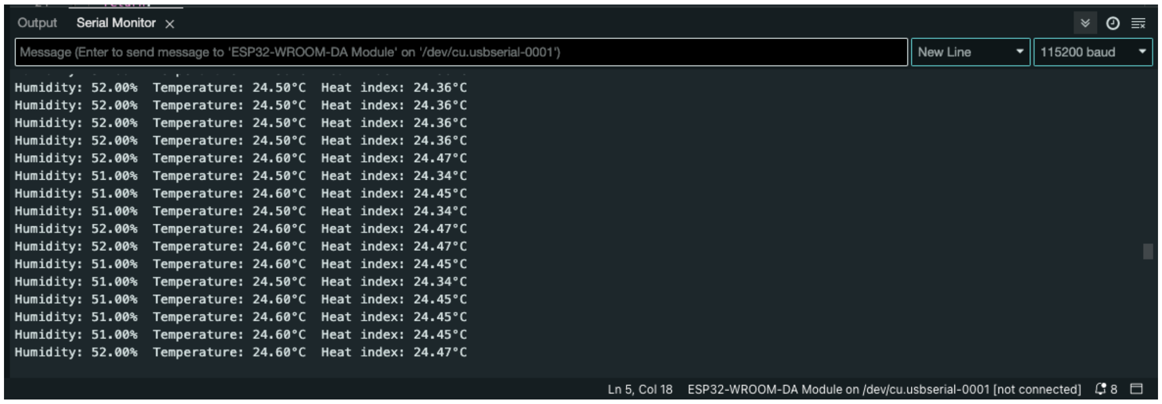
We also thoroughly tested the connection of the MCU to the web application via HTTP calls using AWS Lightsail. We added functionality for the MCU with the web app and tested the data pipeline from the web app to the MCU through AWS Lightsail HTTP calls, both when connected to the same WiFi and on different WiFi networks to ensure off-network functionality was possible. Once we confirmed the data flowed properly, we also tested the timing. As stated in the requirements, we ensured that this data moved through the pipeline in under 5000ms. We aimed not only to deliver a reliable application but a responsive one as well. We also developed the ChatGPT API layer to perform data analysis on the data from the plant. These APIs were tested in a local environment with unit testing to ensure accurate delivery of analysis from the ChatGPT API layer.

9.2 Sensor Testing

All sensors in our system are functioning correctly and providing accurate readings. While investigating variations in our pH measurements, we discovered these changes corresponded with a UCF water valve burst incident that temporarily affected the water supply's acidity levels during our testing period.

9.2.1 Humidity and Temperature Sensor

The DHT11 was plugged into a breadboard in conjunction with the ESP-WROOM-32D and wired up to pin 4. The microcontroller was then selected in the Arduino IDE and programmed to extract the data in the sensor to be viewed in the serial monitor.


Output
Serial Monitor
Message (Enter to send message to 'ESP32-WROOM-DA Module' on '/dev/cu.usbserial-0001')
New Line
115200 baud
Ln 5, Col 18
ESP32-WROOM-DA Module on /dev/cu.usbserial-0001 [not connected]
8

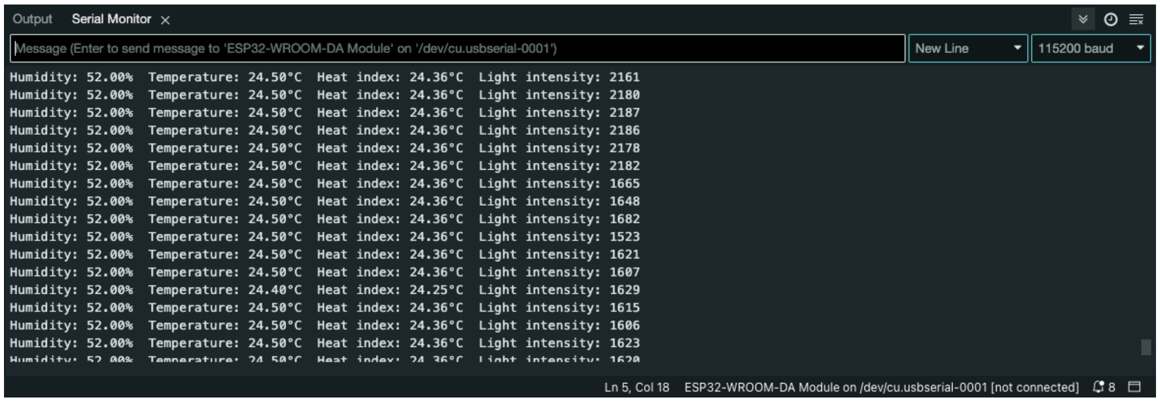
```
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C
Humidity: 52.00%  Temperature: 24.60°C  Heat index: 24.47°C
Humidity: 51.00%  Temperature: 24.50°C  Heat index: 24.34°C
Humidity: 51.00%  Temperature: 24.60°C  Heat index: 24.45°C
Humidity: 51.00%  Temperature: 24.50°C  Heat index: 24.34°C
Humidity: 52.00%  Temperature: 24.60°C  Heat index: 24.47°C
Humidity: 52.00%  Temperature: 24.60°C  Heat index: 24.47°C
Humidity: 51.00%  Temperature: 24.60°C  Heat index: 24.45°C
Humidity: 51.00%  Temperature: 24.50°C  Heat index: 24.34°C
Humidity: 51.00%  Temperature: 24.60°C  Heat index: 24.45°C
Humidity: 51.00%  Temperature: 24.60°C  Heat index: 24.45°C
Humidity: 51.00%  Temperature: 24.60°C  Heat index: 24.45°C
Humidity: 52.00%  Temperature: 24.60°C  Heat index: 24.47°C
```

Figure 9.1. Output of testing the DHT11

The results were as expected, and we can confirm the sensor reacted to changing conditions.

9.2.2 Light Sensor

The sensor was plugged into a breadboard in conjunction with the ESP-WROOM-32D and wired up to pin 36. The microcontroller was then programmed to extract the data in the sensor to be viewed in the serial monitor.


Output
Serial Monitor
Message (Enter to send message to 'ESP32-WROOM-DA Module' on '/dev/cu.usbserial-0001')
New Line
115200 baud
Ln 5, Col 18
ESP32-WROOM-DA Module on /dev/cu.usbserial-0001 [not connected]
8

```
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 2161
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 2180
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 2187
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 2186
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 2178
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 2182
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 1665
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 1648
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 1682
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 1523
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 1621
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 1607
Humidity: 52.00%  Temperature: 24.40°C  Heat index: 24.25°C  Light intensity: 1629
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 1615
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 1606
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 1623
Humidity: 52.00%  Temperature: 24.50°C  Heat index: 24.36°C  Light intensity: 1620
```

Figure 9.2. Output of testing OPT3001

The results were satisfactory and the changes in the values were associated with us covering the sensor to ensure it reacted.

9.2.3 Color Sensor

To verify the accuracy and functionality of the TCS34725 color sensor, we conducted tests by following a similar setup to the LED testing process. The sensor was connected to a breadboard, which was interfaced with a Microsoft Surface Pro. For testing, we used an iPhone to shine various RGB values onto the sensor. The objective was to determine whether the sensor could detect and report a general range of RGB values corresponding to the colors displayed. Through our custom software application, we cycled through different colors on the iPhone screen and recorded the sensor's output. The TCS34725 successfully detected the RGB values, providing readings that accurately represented the colors within a reasonable range. This testing confirmed the sensor's ability to identify color variations, which is essential for diagnosing nutrient deficiencies in the basil leaves based on their color changes in our system.

9.3 Water Pump

The testing for the water pump was minimal as we just wanted to ensure it was capable of dispersing water at a rate that was desired in a container that would be similar in size to the water reservoir that is going to be implemented. We filled a container with water and plugged it in and were able to observe that the water was being pushed by the pump throughout the entire container. The adjustable speed also worked as intended as we could visibly tell the water was being dispersed at a quicker rate.

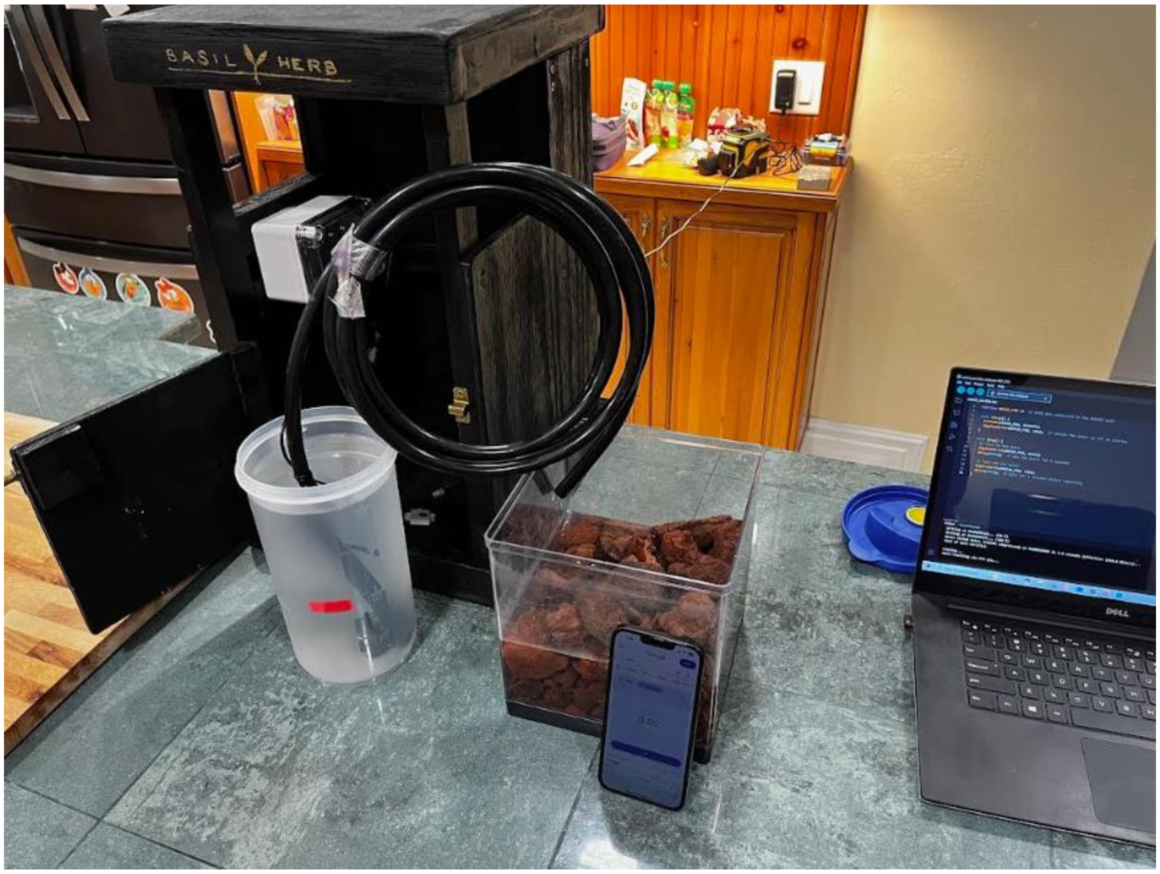


Figure 9.3. SD2 Updated Water Pump Demo

9.4 LED Lights

To ensure the proper functionality of the WS2812B RGB LEDs, we conducted a series of tests by connecting the LEDs to a breadboard (using a MEGA 2560 board), which was then interfaced with a Microsoft Surface Pro. The primary objective was to verify that the LEDs received sufficient voltage and could accurately adjust to our desired RGB values within a 5% FSR. Using a controlled test environment, we varied the RGB values through Arduino on the Surface Pro, monitoring the output with a multimeter and a light sensor. The results confirmed that the LEDs not only received the appropriate voltage levels but also responded accurately to the color adjustments, maintaining a precision within the 5% FSR threshold. This testing process validated the LEDs' capability to provide the precise lighting conditions necessary for optimal plant growth.

9.5 SD2 Plans for Hardware Components

As of SD1, we have done minor testing on a few of the sensors, water pump, LED lights, and the MCU. The plan going forward is to continue to test parts as we get the rest of them in to ensure they will be applicable to what we need in our design. This will be done during the break right before SD2 starts and continue into the start of the semester.

At the start of SD2, we will be integrating all the parts available and begin testing them in an environment that will be closest to what our product is simulating to ensure they can handle the stress. The most integral part to be tested will be the hydroponics system and how to fit that in our enclosure to not make the product too heavy.

Chapter 10 - Administrative Content

In this section we outline the administrative aspects of the system. This section addresses the budget and financing, providing a detailed bill of materials and cost analysis. Additionally, it includes project milestones for Senior Design 1 (SD1) and Senior Design 2 (SD2), highlighting key deliverables and timelines. A table of work distributions is also presented, indicating the primary and secondary responsibilities of team members. This chapter aims to provide a comprehensive overview of the project's administrative planning and resource allocation.

10.1 Bom

The budget we have been allotted for this project is $1000. This is imposed by our sponsor Dr. Michael Pape who has graciously offered us this much for our design. We managed to stay way under the budget.

Table 10.1. BOM

Sub-system	Cost
Sensors, MCU and Motor	$101.97
Enclosure	$76.58
PCB Parts	$284.56
Lighting	$17.44
Software	$87.65
Miscellaneous	$91.70
Total	**$659.90**

10.2 Distribution of Work

Table 10.2. Distribution of Work

Creol	**Tasks**
Justin Press	Integration of LED lights
	Color Sensor Selection and Integration
	Light Sensor Selection and Integration
Electrical Engineering	**Tasks**
Justin Boyd	PCB Design
	Water Pump Selection and Integration
	Voltage Regulation
	Water Level Sensor Selection and Integration
Computer Engineering	**Tasks**
Logan Voiselle	MCU Selection and Integration
	Temperature, pH and Humidity Sensor Selection and Integration
	Communication to Web App
Computer Engineering	**Tasks**
Casey O'Donahoe	Website Design and Management
	Software Design and Management

10.3 Milestones

10.3.1 Senior Design I

Table 10.3. Major Milestones for Senior Design 1

Task	Timeline	Status
Pick Project Idea	05/14/24 - 05/17/24	Complete
Project Research and Writing 10 Page D&C Assignment	05/18/24 - 05/31/24	Complete
Research Components	06/01/24 - 06/06/24	Complete
Review and Upload D&C Report - Start Draft for 60 Page Report - Meet with SD Professors, Sponsor and Reviewers - Start Ordering Parts for Testing	06/07/24 - 06/20/24	Complete
Research, Testing Parts, and Finishing 60 Page Report	06/21/24 - 07/07/24	Complete
Review and Upload 60 Page Report	07/08/24 - 07/12/24	Complete
Finish 120 Page Report and Film Mini Demo Video	07/13/24 -07/23/24	Complete

10.3.2 Senior Design II

Table 10.4. Major Milestones for Senior Design 2

Task	Timeline	Status
Review and Revise 120 Page Document and Start Working on PCB	08/19/24 - 08/26/24	Comple te
Finish Custom PCB and Begin Building Prototype	08/27/24 - 09/15/24	Comple te
Finish Building Prototype and Work on Presentation	09/16/24 - 10/15/24	Comple te
Test Prototype and Work out Problems	10/16/24 - 11/03/24	Comple

		te
Finish Final Product and Practice Presentation	11/04/24 - 11/19/24	Comple te
Finalize Report and Upload Everything on Website	11/20/24 - 11/25/24	Comple te
Final Presentation	11/22/24	Comple te

Chapter 11 - Conclusion

In conclusion, project BASIL HERB is a positive step forward for the cultivation of indoor herbs. As previously stated in our executive summary, the main goal of this project is to provide an advanced growing medium that is capable of growing herbs in the most optimal conditions possible. This design will enable any consumer to grow their own fresh herbs without having to constantly worry if the herb is receiving the necessary care. BASIL HERB ensures a plentiful harvest without the use of chemicals so the consumer can feel good about enjoying their own fresh herbs.

The main components of the project being the sensors, hydroponic system, optics design, and user-friendly web interface are combined to create a complete environment capable of optimal growth and real-time monitoring. The sensors will be constantly monitoring the environmental conditions to ensure the plant is always in its most optimal growing state allowing its growth to flourish. The hydroponics system is working in conjunction with the pump to ensure the herbs are always receiving the necessary nutrients to all parts of the roots causing accelerated growth. The photonics system uses LED grow lights emitting optimal frequencies on a timer to simulate a regular day and night cycle which allows the herbs to photosynthesize naturally. The color sensor is responsible for monitoring the leaves of the plant and detecting any deficiencies as they arise so the consumer will be able to know exactly what's wrong with the plant before it's too late.

The data from each sensor flows to the web application so the user can monitor its condition. The web application works with multiple systems at a given time and can perform under traffic stress. The multiple systems the web application is working with, the MCU, Database, OpenAI ChatGPT and its own user interface, this system has the chance of creating a complex web of systems interacting with one another, and the goal is to do that as efficient as possible for the end user. This is achieved by creating a clean and easy to use user interface using AngularJS, a complex backend leveraging ExpressJS all hosted on a NodeJS server. This is supported by an Open API Key from OpenAI to leverage ChatGPT for data analysis, and all of these are stored within a MySQL database instance stored in the cloud (AWS Lightsail). This is deployed on an AWS MEAN Stack web server that was created with a linux box. Should any problem arise, the web app will notify the user so that appropriate action can be taken to fix the issue. All the components are working together in perfect harmony to keep the herbs healthy and growing optimally.

This document contains our entire design process including the research of integral parts, the testing and integration of our software and hardware components, and the constraints that come with creating this product. The comprehensive planning and research done by our team have positioned us well for the successful implementation and operation of the BASIL HERB system. Moving forward, the knowledge and experience gained from this project will contribute to the future of sustainable and efficient herb cultivation, offering a reliable and user-friendly solution for optimal herb growth.

Appendix A - References

[1] Organic Trade Association. (2020). U.S. Organic Industry Survey 2020.

[2] American College Health Association. (2019). National College Health Assessment II: Reference Group Executive Summary Spring 2019.

[3] MarketsandMarkets. (2020). Smart Greenhouse Market by Technology, Type, Component, and Region - Global Forecast to 2025.

[4]"Hydroponics."*Hydroponics|NationalAgriculturalLibrary*, www.nal.usda.gov/farms-and-agricultural-production-systems/hydroponics.

[5] SITNFlash. "Hydroponics: The Power of Water to Grow Food." *Science in the News*, 4 Oct.2019, sitn.hms.harvard.edu/flash/2019/hydroponics-the-power-of-water-to-grow-food/.

[6] Velazquez-Gonzalez, Roberto S., et al. "A Review on Hydroponics and the Technologies Associated for Medium- and Small-Scale Operations." *MDPI*, Multidisciplinary Digital Publishing Institute, 29 Apr. 2022, www.mdpi.com/2077-0472/12/5/646.

[7] AG, Sensirion. "Sht31-Dis-B." *±2% (0-100%RH) Digital Humidity and Temperature Sensor*, sensirion.com/products/catalog/SHT31-DIS-B.

[8]"APDS-9301."*BroadcomInc.*, www.broadcom.com/products/optical-sensors/ambient-light-photo-sensors/apds-9301.

[9] "Arduino Mkr Wan 1300 (Lora)." *YADOM Must-Have Supplier of IoT Sigfox Lora M2M NBIOT*, yadom.eu/arduino-mkr-wan-1300.html.

[10]"ArduinoMKR1000WIFI."*ArduinoOnlineShop*, store-usa.arduino.cc/products/arduino-mkr1000-wifi?gad_source=1&gclid=CjwKCAjwnK60BhA9EiwAmpHZwzUDqOmsvKBiTDV1Eu5ALqkoYOFaW7RV7GxYCInzJbwh0AVy-Hp64BoCNaoQAvD_BwE.

[11]BH1750FVI :SensorICS, www.mouser.com/datasheet/2/348/bh1750fvi-e-186247.pdf.

[12]Digital-OutputRelativeHumidity&TemperatureSensor/Module, cdn-shop.adafruit.com/datasheets/DHT22.pdf.

[13]DS18B20-ProgrammableResolution1-WireDigital ..., www.analog.com/media/en/technical-documentation/data-sheets/ds18b20.pdf.

[14]ESP32-S2SeriesDatasheet, www.espressif.com/sites/default/files/documentation/esp32-s2_datasheet_en.pdf.

[15]ESP32-WROOM-32(ESP-WROOM-32)Datasheet, www.mouser.com/datasheet/2/891/esp-wroom-32_datasheet_en-1223836.pdf.

[16]ESP32WROVERE&ESP32WROVERIE, www.espressif.com/sites/default/files/documentation/esp32-wrover-e_esp32-wrover-ie_datasheet_en.pdf.

[17]“Gravity: Analog Ph Sensor / Meter Kit for Arduino.” *DFRobot*, www.dfrobot.com/product-1025.html.

[18]Honeywell HumidIconTM: Hih6130/6131 Series With ..., prod-edam.honeywell.com/content/dam/honeywell-edam/sps/siot/en-ca/products/sensors/humidity-with-temperature-sensors/honeywell-humidicon-hih6100-series/documents/sps-siot-hih6130-6131-install-50061154-3-en-ciid-142166.pdf.

[19]“HumiditySensorBME280.”*BoschSensortec*, www.bosch-sensortec.com/products/environmental-sensors/humidity-sensors-bme280/.

[20]Industries, Adafruit. “Adafruit TSL2561 Digital Luminosity/Lux/Light Sensor Breakout.” *Adafruit Industries Blog RSS*, www.adafruit.com/product/439.

[21]“Lab Grade Ph Probe.” *Atlas Scientific*, 17 June 2024, www.atlas-scientific.com/probes/ph-probe/.

[22]MCP9808, www.microchip.com/en-us/product/mcp9808.

[23]“OPT3001.” *OPT3001 Data Sheet, Product Information and Support* | *TI.Com*, www.ti.com/product/OPT3001?bm-verify=AAQAAAAJ_____yuTDmhoa_ddlN8UkBKAVHoKJnRo0FLko6RepZuEQgvlA15YiuHaebQCxShpZlFb3qFEk2hZB2IURfK6kbg4erDNmY_X8GzwDlXdUoMCzGeHNcaOmExhnTBzGZRiwJzP_0up8c56Jbt4H5Im43hEBeLmr3le_cbzC5gEuDDD1F3Pc9vf7inEdWgnchGmBDl2_AjPTnx-B1EdAf5FeyN6WC-SKhSY0rgIR9psOEcyOXk21OMlakcbxMZFges-3_P1Gieec-_8a2sI640mnmIgh_g#description.

[24]“Ph Sensor.” *Vernier*, 29 May 2024, www.vernier.com/product/ph-sensor/.

[25]RaspberryPi4ModelB, datasheets.raspberrypi.com/rpi4/raspberry-pi-4-product-brief.pdf.

[26]“Raspberry Pi Zero 2 W.” *Raspberry Pi Zero 2 W - Waveshare Wiki*, www.waveshare.com/wiki/Raspberry_Pi_Zero_2_W.

[27]SI7021-A20-I2CHumidityandTemperatureSensor, www.silabs.com/documents/public/data-sheets/Si7021-A20.pdf.

[28]"TMP36." *TMP36 Datasheet and Product Info | Analog Devices*, www.analog.com/en/products/tmp36.html.

[29]Veml7700 High Accuracy Ambient Light Sensor with I2C ..., www.vishay.com/docs/84286/veml7700.pdf.

[30] McCree, K. J. (1971). The action spectrum, absorption and fluorescence of chlorophylls a and b. Plant Physiology, 47(6), 815-825.

[31] Carter, G. A. (1993). Responses of leaf spectral reflectance to nutrient deficiencies in herbaceous plants. Remote Sensing of Environment, 44(2), 273-283.

[32] Miao, Y., Li, Y., Gong, W., Song, S., Fan, M., & Xu, Z. (2018). Estimation of chlorophyll concentration and nitrogen content in tobacco leaves using hyperspectral remote sensing data. Sensors (Switzerland), 18(12), 4294.

[33] Peñuelas, J., Baret, F., & Filella, I. (1995). Semi-empirical indices to assess remotely sensed vegetation cover. Photogrammetric Engineering & Remote Sensing, 61(4), 251-256.

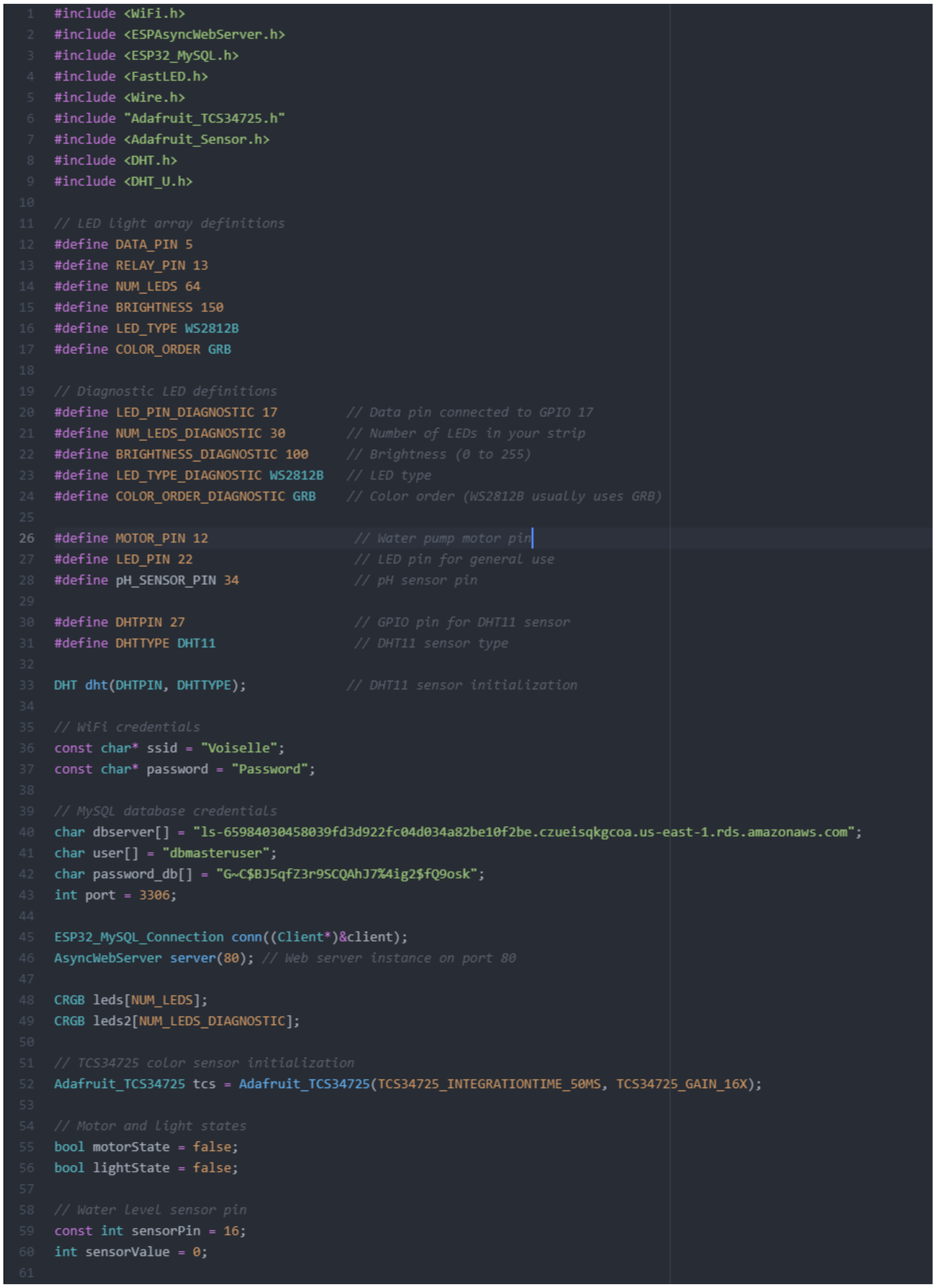
[34] Thenkabail, P. S., & Morton, C. M. (2018). Hyperspectral indices for ecological applications and high-throughput phenotyping. Trends in Plant Science, 23(2), 172-182.

[35] Zhang, J., He, Y., Wang, R., Feng, H., & Fu, Z. (2018). Spectral reflectance-based detection of phosphorus deficiency in flue-cured tobacco using multivariate analysis. Journal of Integrative Agriculture, 17(10), 2423-2432.

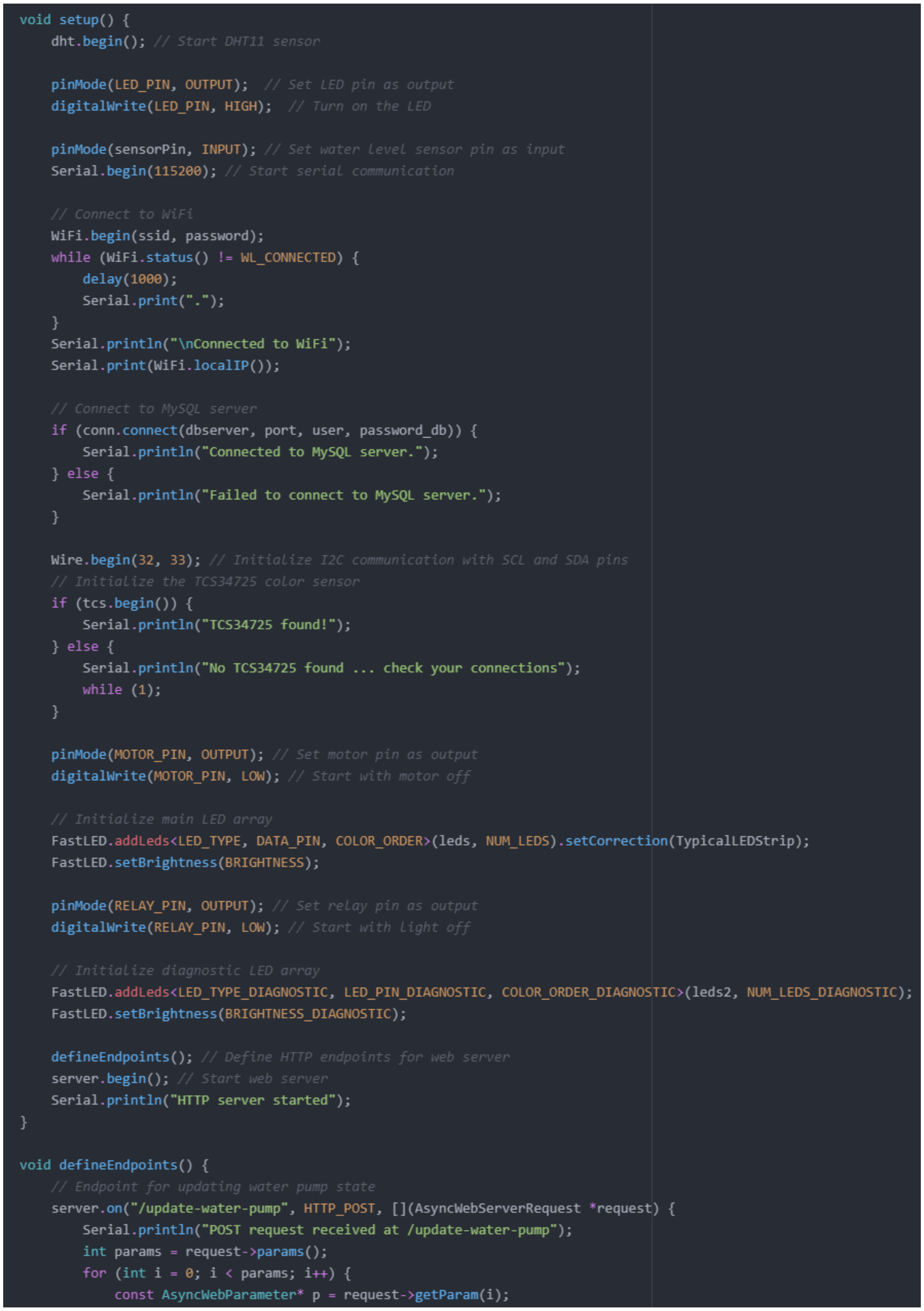
[36] Fogg, B. J., Soohoo, C., Danielson, D. R., Marable, L., Stanford, J., & Tauber, E. R. (2003). How do users evaluate the credibility of web sites? Stanford Persuasive Technology Lab. https://doi.org/10.1145/997078.997097

[GPTC] We have utilized LLM for drafting, outlining, comparing, summarizing, and proofreading purposes.

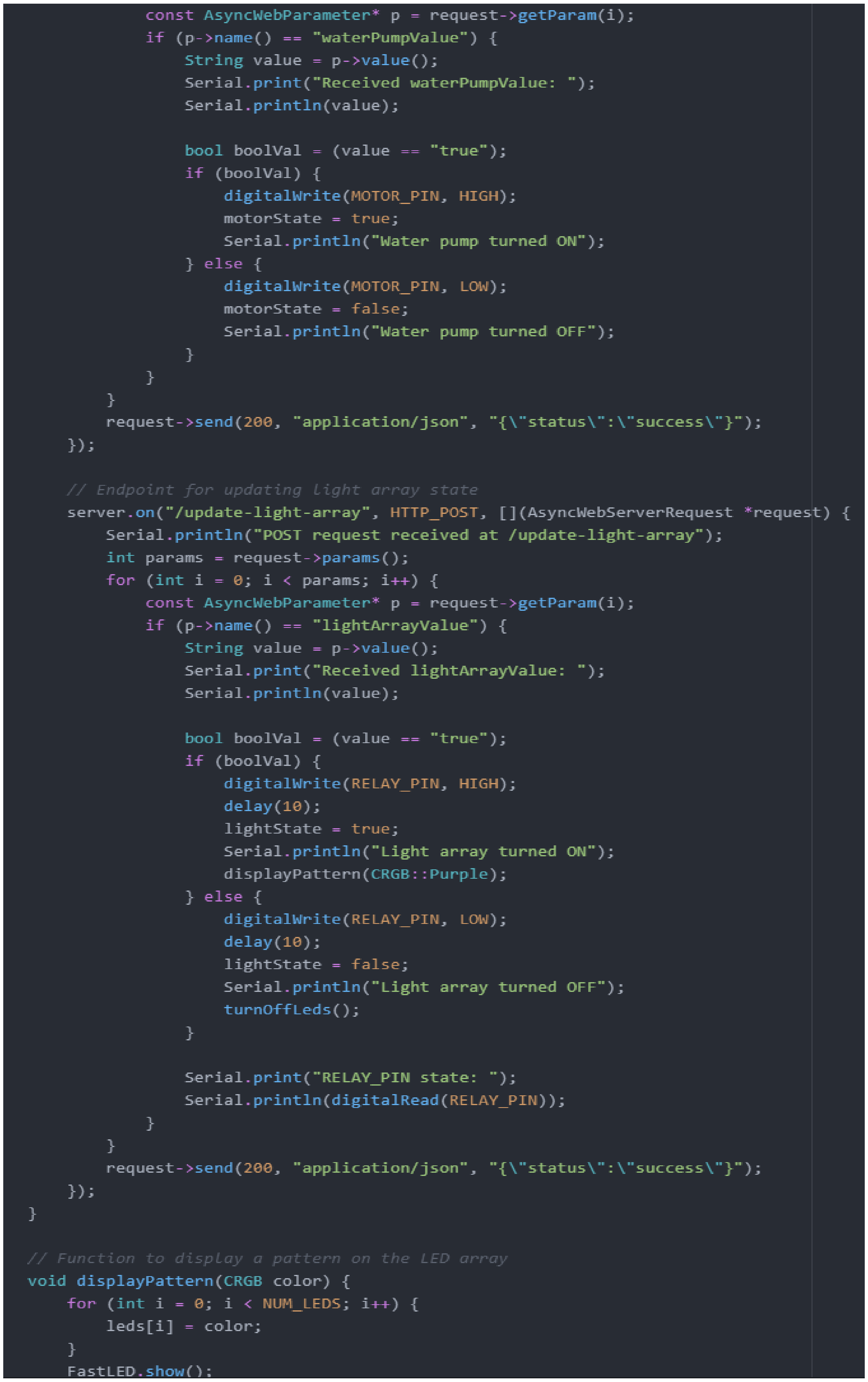
Appendix B - Code



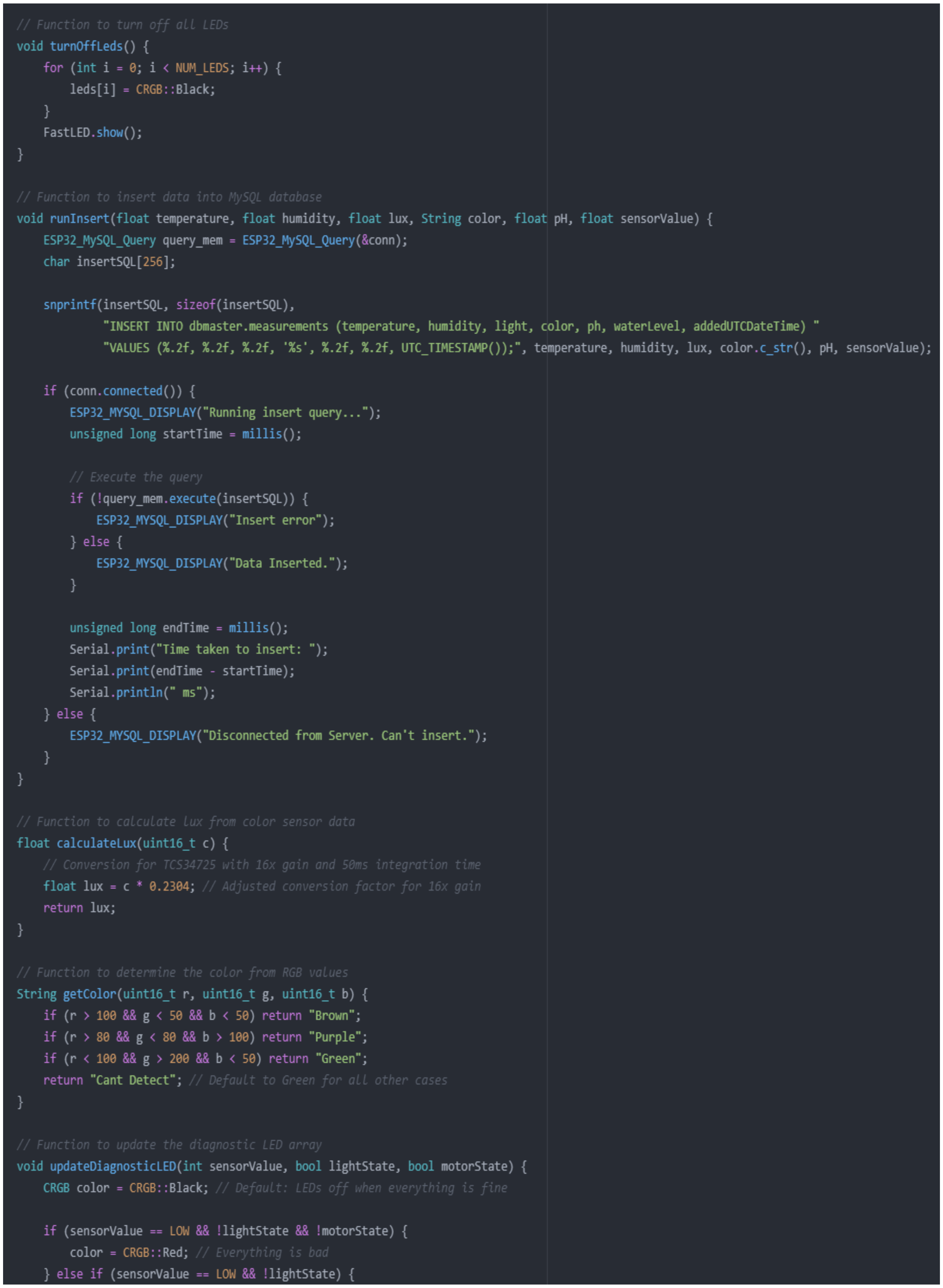
```cpp
#include <WiFi.h>
#include <ESPAsyncWebServer.h>
#include <ESP32_MySQL.h>
#include <FastLED.h>
#include <Wire.h>
#include "Adafruit_TCS34725.h"
#include <Adafruit_Sensor.h>
#include <DHT.h>
#include <DHT_U.h>

// LED light array definitions
#define DATA_PIN 5
#define RELAY_PIN 13
#define NUM_LEDS 64
#define BRIGHTNESS 150
#define LED_TYPE WS2812B
#define COLOR_ORDER GRB

// Diagnostic LED definitions
#define LED_PIN_DIAGNOSTIC 17         // Data pin connected to GPIO 17
#define NUM_LEDS_DIAGNOSTIC 30        // Number of LEDs in your strip
#define BRIGHTNESS_DIAGNOSTIC 100     // Brightness (0 to 255)
#define LED_TYPE_DIAGNOSTIC WS2812B   // LED type
#define COLOR_ORDER_DIAGNOSTIC GRB    // Color order (WS2812B usually uses GRB)

#define MOTOR_PIN 12                   // Water pump motor pin
#define LED_PIN 22                     // LED pin for general use
#define pH_SENSOR_PIN 34               // pH sensor pin

#define DHTPIN 27                      // GPIO pin for DHT11 sensor
#define DHTTYPE DHT11                  // DHT11 sensor type

DHT dht(DHTPIN, DHTTYPE);             // DHT11 sensor initialization

// WiFi credentials
const char* ssid = "Voiselle";
const char* password = "Password";

// MySQL database credentials
char dbserver[] = "ls-65984030458039fd3d922fc04d034a82be10f2be.czueisqkgcoa.us-east-1.rds.amazonaws.com";
char user[] = "dbmasteruser";
char password_db[] = "G~C$BJ5qfZ3r9SCQAhJ7%4ig2$fQ9osk";
int port = 3306;

ESP32_MySQL_Connection conn((Client*)&client);
AsyncWebServer server(80); // Web server instance on port 80

CRGB leds[NUM_LEDS];
CRGB leds2[NUM_LEDS_DIAGNOSTIC];

// TCS34725 color sensor initialization
Adafruit_TCS34725 tcs = Adafruit_TCS34725(TCS34725_INTEGRATIONTIME_50MS, TCS34725_GAIN_16X);

// Motor and light states
bool motorState = false;
bool lightState = false;

// Water level sensor pin
const int sensorPin = 16;
int sensorValue = 0;

```



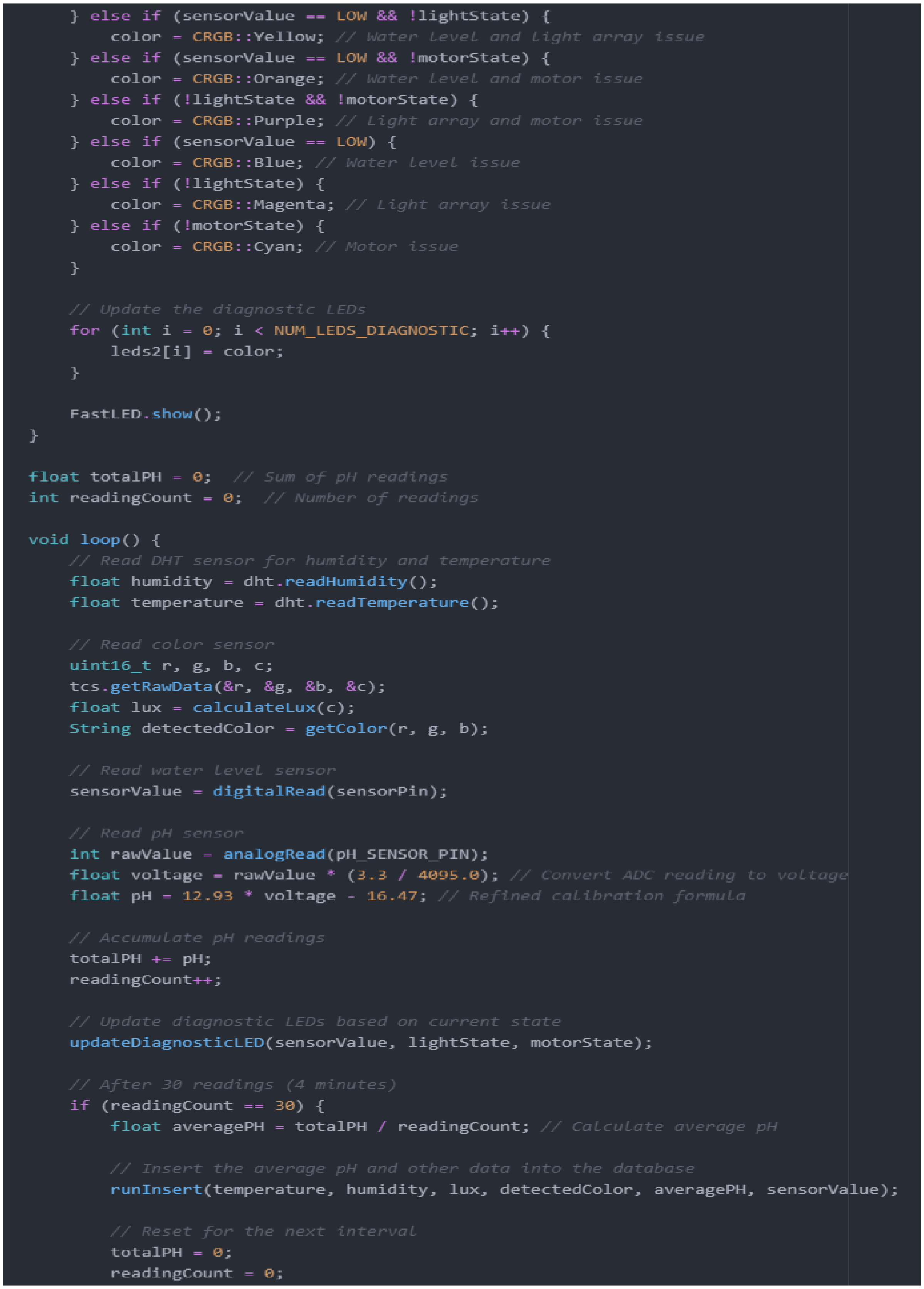
```
void setup() {
    dht.begin(); // Start DHT11 sensor

    pinMode(LED_PIN, OUTPUT);  // Set LED pin as output
    digitalWrite(LED_PIN, HIGH);  // Turn on the LED

    pinMode(sensorPin, INPUT); // Set water level sensor pin as input
    Serial.begin(115200); // Start serial communication

    // Connect to WiFi
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(1000);
        Serial.print(".");
    }
    Serial.println("\nConnected to WiFi");
    Serial.print(WiFi.localIP());

    // Connect to MySQL server
    if (conn.connect(dbserver, port, user, password_db)) {
        Serial.println("Connected to MySQL server.");
    } else {
        Serial.println("Failed to connect to MySQL server.");
    }

    Wire.begin(32, 33); // Initialize I2C communication with SCL and SDA pins
    // Initialize the TCS34725 color sensor
    if (tcs.begin()) {
        Serial.println("TCS34725 found!");
    } else {
        Serial.println("No TCS34725 found ... check your connections");
        while (1);
    }

    pinMode(MOTOR_PIN, OUTPUT); // Set motor pin as output
    digitalWrite(MOTOR_PIN, LOW); // Start with motor off

    // Initialize main LED array
    FastLED.addLeds<LED_TYPE, DATA_PIN, COLOR_ORDER>(leds, NUM_LEDS).setCorrection(TypicalLEDStrip);
    FastLED.setBrightness(BRIGHTNESS);

    pinMode(RELAY_PIN, OUTPUT); // Set relay pin as output
    digitalWrite(RELAY_PIN, LOW); // Start with light off

    // Initialize diagnostic LED array
    FastLED.addLeds<LED_TYPE_DIAGNOSTIC, LED_PIN_DIAGNOSTIC, COLOR_ORDER_DIAGNOSTIC>(leds2, NUM_LEDS_DIAGNOSTIC);
    FastLED.setBrightness(BRIGHTNESS_DIAGNOSTIC);

    defineEndpoints(); // Define HTTP endpoints for web server
    server.begin(); // Start web server
    Serial.println("HTTP server started");
}

void defineEndpoints() {
    // Endpoint for updating water pump state
    server.on("/update-water-pump", HTTP_POST, [](AsyncWebServerRequest *request) {
        Serial.println("POST request received at /update-water-pump");
        int params = request->params();
        for (int i = 0; i < params; i++) {
            const AsyncWebParameter* p = request->getParam(i);
```



```
            const AsyncWebParameter* p = request->getParam(i);
            if (p->name() == "waterPumpValue") {
                String value = p->value();
                Serial.print("Received waterPumpValue: ");
                Serial.println(value);

                bool boolVal = (value == "true");
                if (boolVal) {
                    digitalWrite(MOTOR_PIN, HIGH);
                    motorState = true;
                    Serial.println("Water pump turned ON");
                } else {
                    digitalWrite(MOTOR_PIN, LOW);
                    motorState = false;
                    Serial.println("Water pump turned OFF");
                }
            }
        }
        request->send(200, "application/json", "{\"status\":\"success\"}");
    });

    // Endpoint for updating light array state
    server.on("/update-light-array", HTTP_POST, [](AsyncWebServerRequest *request) {
        Serial.println("POST request received at /update-light-array");
        int params = request->params();
        for (int i = 0; i < params; i++) {
            const AsyncWebParameter* p = request->getParam(i);
            if (p->name() == "lightArrayValue") {
                String value = p->value();
                Serial.print("Received lightArrayValue: ");
                Serial.println(value);

                bool boolVal = (value == "true");
                if (boolVal) {
                    digitalWrite(RELAY_PIN, HIGH);
                    delay(10);
                    lightState = true;
                    Serial.println("Light array turned ON");
                    displayPattern(CRGB::Purple);
                } else {
                    digitalWrite(RELAY_PIN, LOW);
                    delay(10);
                    lightState = false;
                    Serial.println("Light array turned OFF");
                    turnOffLeds();
                }

                Serial.print("RELAY_PIN state: ");
                Serial.println(digitalRead(RELAY_PIN));
            }
        }
        request->send(200, "application/json", "{\"status\":\"success\"}");
    });
}

// Function to display a pattern on the LED array
void displayPattern(CRGB color) {
    for (int i = 0; i < NUM_LEDS; i++) {
        leds[i] = color;
    }
    FastLED.show();
```



```
// Function to turn off all LEDs
void turnOffLeds() {
    for (int i = 0; i < NUM_LEDS; i++) {
        leds[i] = CRGB::Black;
    }
    FastLED.show();
}

// Function to insert data into MySQL database
void runInsert(float temperature, float humidity, float lux, String color, float pH, float sensorValue) {
    ESP32_MySQL_Query query_mem = ESP32_MySQL_Query(&conn);
    char insertSQL[256];

    snprintf(insertSQL, sizeof(insertSQL),
             "INSERT INTO dbmaster.measurements (temperature, humidity, light, color, ph, waterLevel, addedUTCDateTime) "
             "VALUES (%.2f, %.2f, %.2f, '%s', %.2f, %.2f, UTC_TIMESTAMP());", temperature, humidity, lux, color.c_str(), pH, sensorValue);

    if (conn.connected()) {
        ESP32_MYSQL_DISPLAY("Running insert query...");
        unsigned long startTime = millis();

        // Execute the query
        if (!query_mem.execute(insertSQL)) {
            ESP32_MYSQL_DISPLAY("Insert error");
        } else {
            ESP32_MYSQL_DISPLAY("Data Inserted.");
        }

        unsigned long endTime = millis();
        Serial.print("Time taken to insert: ");
        Serial.print(endTime - startTime);
        Serial.println(" ms");
    } else {
        ESP32_MYSQL_DISPLAY("Disconnected from Server. Can't insert.");
    }
}

// Function to calculate lux from color sensor data
float calculateLux(uint16_t c) {
    // Conversion for TCS34725 with 16x gain and 50ms integration time
    float lux = c * 0.2304; // Adjusted conversion factor for 16x gain
    return lux;
}

// Function to determine the color from RGB values
String getColor(uint16_t r, uint16_t g, uint16_t b) {
    if (r > 100 && g < 50 && b < 50) return "Brown";
    if (r > 80 && g < 80 && b > 100) return "Purple";
    if (r < 100 && g > 200 && b < 50) return "Green";
    return "Cant Detect"; // Default to Green for all other cases
}

// Function to update the diagnostic LED array
void updateDiagnosticLED(int sensorValue, bool lightState, bool motorState) {
    CRGB color = CRGB::Black; // Default: LEDs off when everything is fine

    if (sensorValue == LOW && !lightState && !motorState) {
        color = CRGB::Red; // Everything is bad
    } else if (sensorValue == LOW && !lightState) {
```



```
    } else if (sensorValue == LOW && !lightState) {
        color = CRGB::Yellow; // Water level and light array issue
    } else if (sensorValue == LOW && !motorState) {
        color = CRGB::Orange; // Water level and motor issue
    } else if (!lightState && !motorState) {
        color = CRGB::Purple; // Light array and motor issue
    } else if (sensorValue == LOW) {
        color = CRGB::Blue; // Water level issue
    } else if (!lightState) {
        color = CRGB::Magenta; // Light array issue
    } else if (!motorState) {
        color = CRGB::Cyan; // Motor issue
    }

    // Update the diagnostic LEDs
    for (int i = 0; i < NUM_LEDS_DIAGNOSTIC; i++) {
        leds2[i] = color;
    }

    FastLED.show();
}

float totalPH = 0;  // Sum of pH readings
int readingCount = 0;  // Number of readings

void loop() {
    // Read DHT sensor for humidity and temperature
    float humidity = dht.readHumidity();
    float temperature = dht.readTemperature();

    // Read color sensor
    uint16_t r, g, b, c;
    tcs.getRawData(&r, &g, &b, &c);
    float lux = calculateLux(c);
    String detectedColor = getColor(r, g, b);

    // Read water level sensor
    sensorValue = digitalRead(sensorPin);

    // Read pH sensor
    int rawValue = analogRead(pH_SENSOR_PIN);
    float voltage = rawValue * (3.3 / 4095.0); // Convert ADC reading to voltage
    float pH = 12.93 * voltage - 16.47; // Refined calibration formula

    // Accumulate pH readings
    totalPH += pH;
    readingCount++;

    // Update diagnostic LEDs based on current state
    updateDiagnosticLED(sensorValue, lightState, motorState);

    // After 30 readings (4 minutes)
    if (readingCount == 30) {
        float averagePH = totalPH / readingCount; // Calculate average pH

        // Insert the average pH and other data into the database
        runInsert(temperature, humidity, lux, detectedColor, averagePH, sensorValue);

        // Reset for the next interval
        totalPH = 0;
        readingCount = 0;
```