Automated Hydroponics System

Senior Design 1 Report

University of Central Florida

Department of Electrical Engineering and Computer Science

Dr. Lei Wei

<u>Group E</u>

Chris Lopez - Electrical Engineering

Luiz Alves - Computer Engineering

Jarrod Pearman - Computer Engineering

Hardik Patel - Electrical Engineering

Table Of Contents

List of Figures

List of Tables

1.0 – Executive Summary

In this document, we will be explaining the concept and implementation of our home automated plant grower. The fundamentals of this project are based on the basic idea of hydroponics, which by definition, is the process of growing plants in sand, gravel, or liquid, with added nutrients but without soil (oxford dictionary source). Now hydroponics has been around for plenty of years. In fact, the first documentation of hydroponics came about in the 1930s by W.F.Gericke from the University of California at Berkeley. Since then, hydroponic systems have grown to an industry of its own where its largest market is in Europe and is vastly growing in Africa.

For this project we will be creating our hydroponics system with water with added nutrients for plants to flourish. Furthermore, we will be building an automated home hydroponics system that the average consumer would be able to set up on their own. A big reason for this is that a great amount of plant growers would benefit from this system and could even invite newcomers to start growing on their own. There will also be a small display for the user to interact with and change parameters to the user's liking. This allows for more variety of plants to grow and even some experiments conducted by the user if they want. More intuitively, an app will also be developed to aid in the process of setting up the system at home and will be capable of setting preset parameters for all kinds of plants to grow. This makes it even easier for the consumer to grow their desired plant. Couple this with hourly updates on the plants progress, a current parameter reading, the consumer will be able to monitor their plant from start to finish and yield amazing results.

In order to achieve our goal of a hands-free hydroponics system, many sensors come into play to make sure the plants are receiving the nourishments they need. The most important factors are the sensors to monitor the water contents for proper nourishment and light sensors to monitor light intensity for plant growth. That said, pH sensors, CO_2 sensors, water sensors and temperature sensors are needed to ensure the most important components are met for plant growth. Since the purpose of our project is to make an automated hydroponics system that can adapt to different home environments, additional sensors and modules are needed to protect the growing process. Humidity sensors, fans, heaters and a clear film enclosure will be included to increase the survivability of the plants. All these components and sensors will be monitored by a microcontroller that will be programmed to keep the measurements in the desired ranges. Since the project design is modeled after the Nutrient Film Technique, pumps are needed to propel the water around the system and administer pH fluid to balance the water contents. Therefore, the user doesn't have to do anything on their part. The only thing the user could do is replace the water reservoir if it gets too dirty over a very long period of use. All in all, a completely automated hydroponic system with every sensor needed for proper plant growth is going to be implemented into our version of hydroponics and deliver an easy to use product that every consumer can use and maintain.

2.0 – Project Description

2.1 – Project Motivation

Over the years, most people seem to think that the most effective way to grow all plants is with water and soil. While this has been the norm throughout history for agriculture, there are better methods out there that can yield better results. Not only better results, but also make it easier to automate so that there are less mistakes in the process of growing plants. This is where hydroponics comes into the picture as it can grow a plant to its full potential and have the process automated for the user. In early methods of hydroponics, the user would have to manually monitor the nutrients in the water and check up their plants daily. On top of this, some research would need to be done to take proper measurements and adjust them to the plant's specifications. This is quite troublesome for the average person and could shy them away from ever trying to grow plants. That said, this would allow people who aren't great with plants to have a little growing station of their own and would be great for people who want to.
The main motivation of this project can be broken down to 3 components, ease of use, portability, and optimum plant growth. Easy of use for the average user to set up and start growing their desired plants. As mentioned before, this lifts a majority of the burden that usually comes with growing plants. All the while, we want our users to constantly be updated on their plants and be able to watch the plants grow from start to finish. An app will be used to update the user and breakdown the settings for them to understand. Hopefully, this could eventually lead to an increase in plant knowledge that could be translated to more exotic plants or expansion to other hydroponic systems.

Another aspect we want to focus on is the portability of the hydroponic system. While size isn't everything, we do want this system to be able to fit through all types of doors and be as mobile as it can be. However, we also want to be able to fit a decent amount of plants in the hydroponic system so that the users can have the option of growing different kinds of plants that fit within the same environment criterion.

The last facet we want to stick by is providing the most optimal plant growth possible and being within the other conditions above. That said, we want all the plants that are growing in our hydroponic system to have enriched water for the plants. In order to do so, many sensors and pumps are needed. This will keep values in ranges for the plants and will administer any fluids that are needed in the system. Since this will ensure ideal conditions for the plants, the user will see desired results. This could even go for plants that bear any flowers, fruits or vegetables. All in all, the growth of plants in our hydroponics system are going to be heavily monitored so that they can give excellent results to our users.

2.2 – Goals and Objectives

The main goal of this project is to make a self-sustaining hydroponic system that can pre set certain plant information to grow and relay that information to the user through an app. To accomplish this goal, certain aspects of the project need to be met.
Usability – A convenient hydroponic system is the main goal as we want the user to do virtually nothing but learn from the growth process. We would also want the user to interact with the system by phone and on the system via buttons to modify parameters. By using an app's graphical user interface to communicate with the system, the user can modify parameters and keep a logged report on the system performance.

Sustainability – The hydroponic system has to be self-sustaining and be able to grow plants in an optimal setting for maximum growth potential. Should there be any interruptions in the development, the user will be notified for them to correct the issue.

Robustness – Since we want the hydroponic system to be able to handle a variety of plants, it has to adapt to their needs to grow properly. This will be done with various sensors to monitor the water nutrient contents and surrounding parameters like temperature for maximum growth potential.

Maintainability – The user shouldn't have to do anything in order to keep the system up and running. The most the user would have to do is replace the water reservoir if it gets too dirty after multiple repeated usages.

Portability – While the system won't be the smallest out there, we want a mid-size device that can hold a decent amount of plants and fit through a regular sized door. The system should also be easy to move around and place a minimal amount of strain on the user to move around.

Scalability – There are plenty of plants that require a long time to grow from start to finish and our goal is to have a system that can function throughout the plants life cycle.

2.3 – Requirements Specifications

Power/Energy

- Sustainable Battery: That can run the microcontroller and any other components needed to keep the system self-sustainable for a decent amount of time in case a dedicated power connection is unavailable. This would be useful if the power were to go out for a small amount of time or if a power surge were to happen and keep the system running smoothly.
- Hydropower Implementation: use the flow of the water in the pumps to generate power to be stored in a battery. This could be used to power other electronics in

the system to save energy or charge external devices like cell phones. This would also be used to charge and sustain the battery mentioned above.
- Hard wall connection to a standard power outlet for consistent and reliable power. This will be the main source of energy and the recommended use case.

Microcontroller
- We will need a microcontroller in order to manage and control important input data.
- Will need to connect to various sensors for readings to manage the system.
- Connection to water pumps and lighting.
- Be power efficient and easy to develop with.
- Flexible design that gives us room to easily add and remove features as we go via the pins available on the microcontroller.
- Connect to the internet with Wi-Fi module for android.
- Have database integration for app development to store preset plants information.

Sensors
- CO2 Sensor.
- PH Sensor.
- Temperature Sensors.
- Sensors to measure light/brightness levels.
- Nutrient sensor.
- Water level sensor.
- Potential other sensors for plant and water quality.
- Ultrasonic sensor to measure distance between lighting and the plants to control lighting height.

Hardware
- Various Pumps for water flow and to administer pH liquids.
- Lighting Equipment.
- Various plumbing for water flow and connections.
- Water Reservoir(s).
- Construction to hold the plants.
- Materials for the physical construction of the overall project.
- Plumbing to empty water from the system easily.
- Nutrient Adder. We want to be able to add nutrients to the water in the system in order to provide plants with their requirements.
- Some waterproofing design for electronic protection.
- Water turbines with additional energy.
- Lift/pulley for lighting: We are adding lifts to the lighting such that the lights raise with the height of the plants so that lighting levels remain constant and predictable.

Software

- Software for the microcontroller to monitor plant vitals.
- Keep basic and focus on functionality.
- Have user interaction with the hydroponic system
    - Use LEDs
    - Buttons/Switches
    - LCD Display
    - Phone App
- Android App
    - Allow the user to monitor system vitals.
    - Provide plant upkeep information.
    - Provide easy interaction with the system.
    - Being able to update parameters in the system.
    - Support many versions of android.
    - Connect over the internet with our microcontroller.
    - Use a database to store data of plant parameters.
    - Implementations include mobile development software.

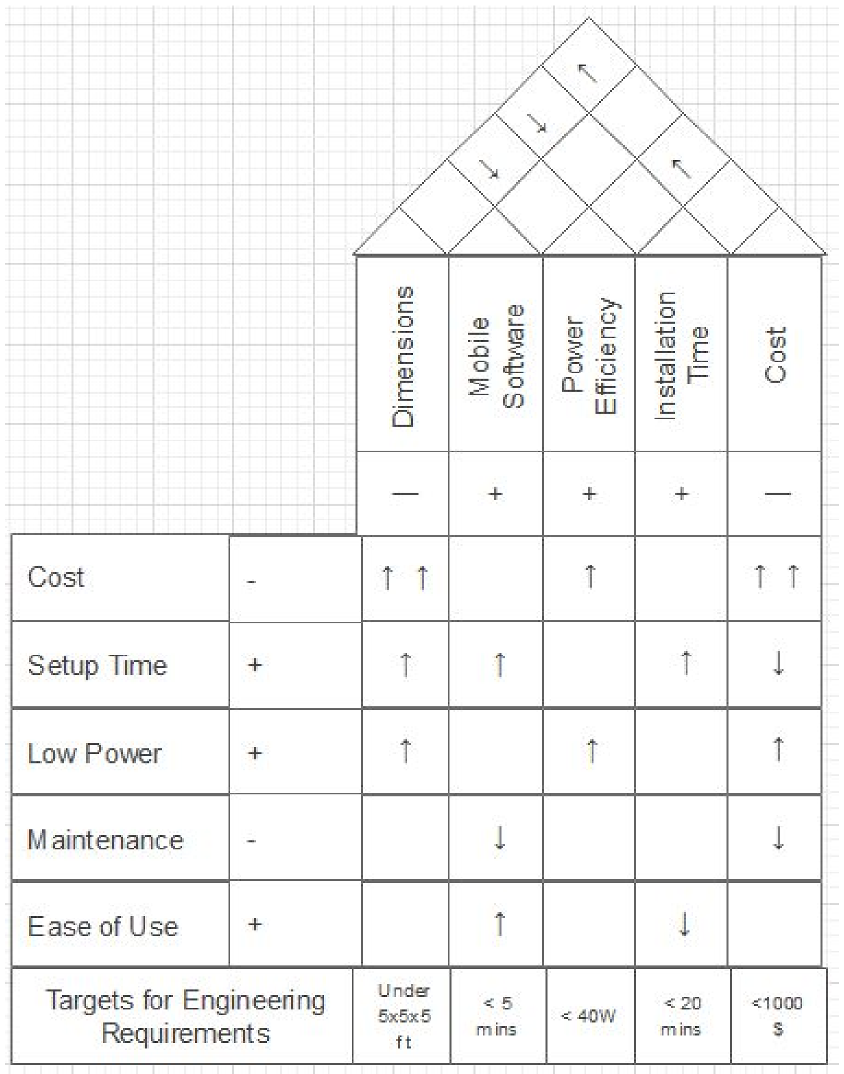
Project Constraints

- It is important to keep the cost of the project within a reasonable budget.
- Power and Energy efficiency.
- Time to work on the project.
- Smaller form factor to comfortably fit in a home.

2.4 – Quality of House Analysis

A house of quality analysis is meant to draw connections between the consumer and the development process. This keeps the project in question on track to satisfy the customers wants and fulfill their requirements. It does this by understanding the customers needs and priorities certain aspects of the project. This way, engineers can allocate resources based on the priorities on the project. On top of this, engineers can draw some requirements based on this data and start to gain some structure on how to proceed with the project.
In our diagram below, we analyze the dimensions, mobile software, power efficiency, installation time and cost of the project. We also cross reference these parameters with the cost, setup time, power usage, maintenance and ease of use of the customer.

- + = Positive polarity
- - = Negative polarity
- ↑↑ = Strong positive correlation
- ↑ = Positive correlation
- ↓ = Negative correlation
- ↓↓ = Strong negative correlation



		Dimensions	Mobile Software	Power Efficiency	Installation Time	Cost
		—	+	+	+	—
Cost	-	↑ ↑		↑		↑ ↑
Setup Time	+	↑	↑		↑	↓
Low Power	+	↑		↑		↑
Maintenance	-		↓			↓
Ease of Use	+		↑		↓	
Targets for Engineering Requirements		Under 5x5x5 ft	< 5 mins	< 40W	< 20 mins	<1000 $

Figure 2.1 - House of Quality

2.5 – Types of Hydroponic Systems

There are plenty of hydroponic systems to consider when designing your own system. There is the drip system, ebb-flow, nutrient film technique, water culture, aeroponics and the wick system. Now all of these come with their own advantages and disadvantages. When picking out which type of system we want to model ours after, they have to fit within the constraints we set in place for the project. Below we discuss different hydroponic models and what they have to offer.

2.5.1 – Drip System

One of the more common hydroponic systems is the drip system. It allows for more versatility and is very easy to expand to accommodate more plants. There aren't many parts in this system, therefore it's easy to build and execute. At its core, the drip system works by dripping solution across the surface of a growing medium where the plants reside. Since the solution is being dropped, there isn't a need for large amounts of water. The solution is dropped by drip lines that can have varying lengths and can also be elongated to cover more ground. Because a growing medium is used to soak up the solution, it can retain more moisture for the plants. This is especially beneficial to larger plants that need more space to grow. Basic components that are used for this type of system are listed below.

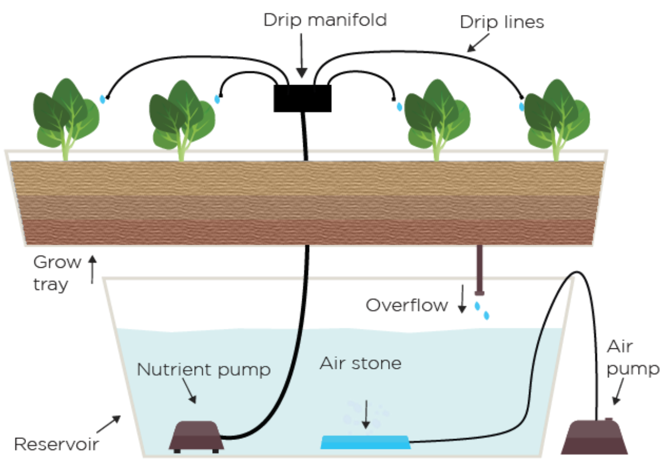


Figure 2.2 - Drip System Example

- A container for the plants.
- A reservoir to hold the nutrient solution.
- A fountain or pond pump to pump nutrient solution.
- A timer to turn the pump on and off.
- Tubing to reach the plants and drip solution back down.
- Drip emitters.
- Growing medium to hold plants.

The drip system operates by having the nutrient solution, which is kept in a separate container or reservoir below, pumped upward into the top container where the plants and growing media reside. Through the drip lines, the nutrient solution drips down onto the plants and growing media. This saturates the growing media and any excess water will flow back down into the reservoir below via tubing. Dripping is generally monitored in a timely manner to make sure a good amount of saturation is reached and to not use more solution than needed.

There are two types of hydroponic drip systems, recirculating/recovery and non-recirculating/non-recovery. In a recovery system, the remaining solution that wasn't absorbed by the plants gets cycled back into the system. As the plants absorb the solution over time, changes in pH and other nutrients occur. The water levels of the system also change as they are absorbed. Because of this, the user has to monitor the systems parameters and adjust them accordingly.

A non-recovery system would have the opposite, meaning there is no recycling of the nutrient solution. This is primarily used for commercial growers as they optimize the distribution of water across the growing media. This means that the growing media is almost perfectly saturated and there is very little runoff of water. Since water use is conservatively used, the reservoir of solution wouldn't need to be replaced often with new solution. This makes it easier to maintain as you wouldn't need to constantly monitor the solution since it will be replaced with correct levels.

2.5.2 – Ebb- Flow/ Flood & Drain

The Flood and Drain technique is another popular type of hydroponics that can be built with common materials and is affordable to produce. The design for this type allows it to thrive in indoor and outdoor environments as the water system is concealed. The basics of this system is that the plant's roots are drenched in the nutrient solution from a pump and it later drains into the reservoir. This gives the plants the time to soak up the solution and to get the oxygen it needs to grow.Basic components that are used for this type of system are listed below.

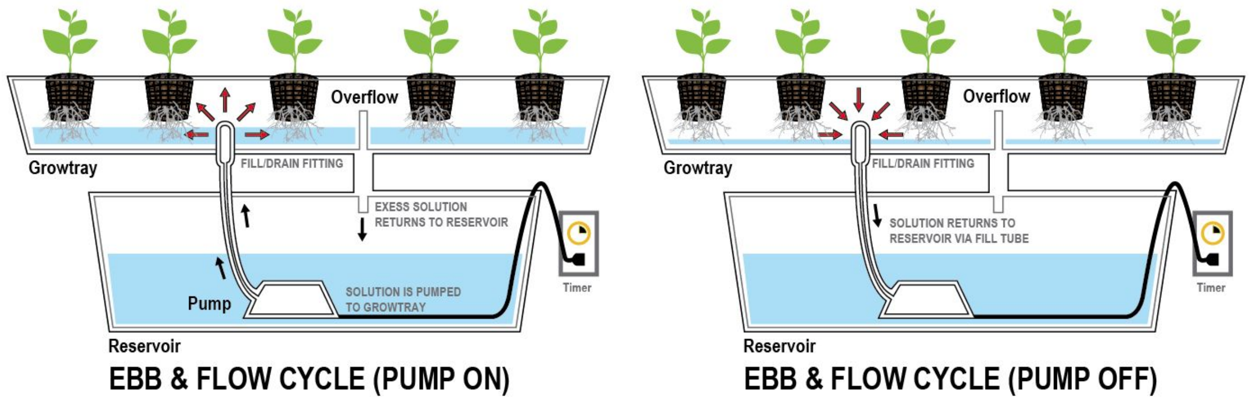

Overflow
Growtray
FILL/DRAIN FITTING
EXESS SOLUTION RETURNS TO RESERVOIR
Pump
SOLUTION IS PUMPED TO GROWTRAY
Timer
Reservoir
EBB & FLOW CYCLE (PUMP ON)
Overflow
Growtray
FILL/DRAIN FITTING
SOLUTION RETURNS TO RESERVOIR VIA FILL TUBE
Timer
Reservoir
EBB & FLOW CYCLE (PUMP OFF)

Figure 2.3 - Ebb & Flow Example

- A container for the plant's.
- A reservoir to hold the nutrient solution.
- A fountain or pond pump to pump nutrient solution.
- A timer to turn the pump on and off.
- Tubing to reach the plants and drip solution back down.
- An overflow tube to set the height of the water level.

- Growing medium to hold plants.

How the complete system works is that a container, or more in some designs, is filled via a pump from a reservoir of nutrient solution. The container fills up to a certain amount to completely coat the plant roots in nutrient solution. In order to prevent overflow, a drain hole is made to catch the excess water and flow down into the reservoir. Then the filled reservoir is drained and goes back down from the same pump it came from. The whole process is in timed intervals to give the plant a sufficient amount of oxygen and nutrients. This process is then repeated for the duration of the plant's growth.

There are many ways to build a flood and drain system. As mentioned before, it can be with many materials that can be filed and drained. For example, you can use buckets, water bottles and trash cans as structures for the plants to reside in. That said, multiple structures can be used to house a plant each or a singular structure can be used to house all plants. All the structure(s) needs is for it to be connected to a water network that can pump and drain the nutrient solution.

2.5.3 – Nutrient Film Technique (N.F.T)

The Nutrient Film Technique is the design type we are going to be using for our project. This is mainly because the design is easy to work with and is more suited for home growing. The biggest change to this technique is that water flows from one side of a channel or tube to the other while touching the roots of all the plants. This differs from the other techniques by touching each plant in a sequential order and not all at once. Basic components that are used for this type of system are listed below.

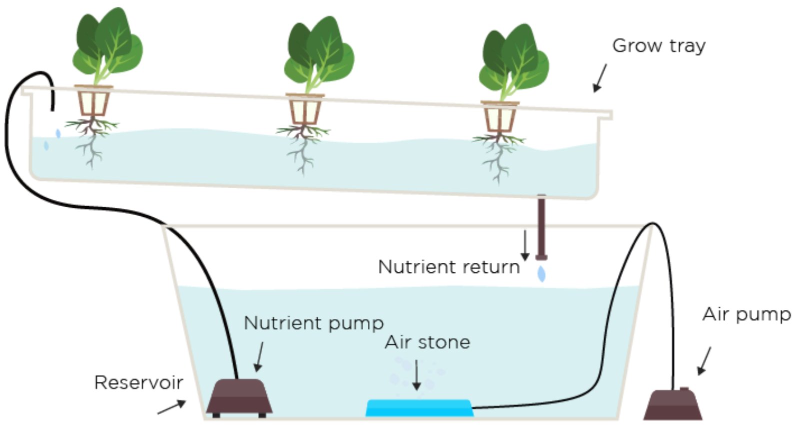


Figure 2.4 - Nutrient Film Technique

- Growing tubes or channels for the plants.
- Containers for the plant's.
- A reservoir to hold the nutrient solution.

- A fountain or pond pump to pump nutrient solution.
- Tubing to reach the plants and to flow solution back down.
- Growing medium to hold plants.

How the Nutrient Film Technique works is that the nutrient solution gets pumped up from a reservoir up to a channel or tube. The nutrient solution then flows from one side of the channel to the other. This lets the solution reach the roots of the plants for growth. In order for the nutrients to flow from one side to another, the channel has to be angled and the nutrients have to be placed at a higher side of the channel. This lets gravity push the water down. Since water only runs along the bottom, oxygen inside the channel is available for absorption. Because the channel is usually enclosed, moisture develops that helps the growth process. After running through each plant, the nutrient solution flows back down into a tube and back into the reservoir. There are 2 main questions when building a nutrient film system. The angle of the channel and how deep the water should be when going through the channel. The angle determines the rate of change in the system. It is said to have only a 1-inch rise over 35 inches of channel. This may not seem like enough to not have a consistent flow of water, but less is more in this case. The deepness of the water should also not exceed more than an inch. The plant's roots act as a natural wick and travel upwards along the root.

2.5.4 – Water Culture

Water culture is the second easiest technique in this list as it doesn’t require two separate containers to flow water from. It consists of a reservoir, containers for the plants and some growing media to keep the plants in place. The simple design consists of the plants resting on top of the reservoir and having the roots are submerged in the reservoir. This allows the plants to absorb the nutrients and no moving are required to achieve this. Since the water isn’t moving around the system, it is generally used for plants that need a long time to grow. With all this in mind, it is an easy set up that is more suitable for beginners to learn about hydroponics. Basic components that are used for this type of system are listed below.

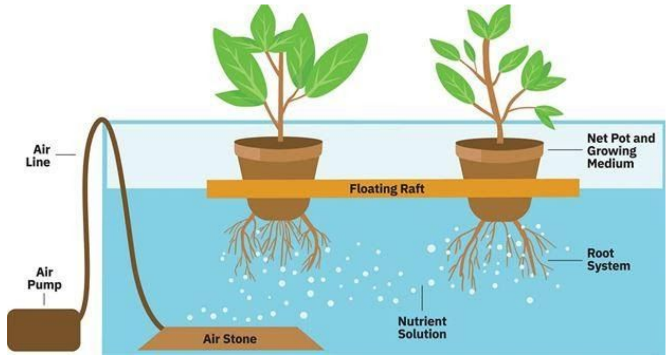


Figure 2.5 - Water Culture Example

- Containers for the plant's.
- A reservoir to hold the nutrient solution.
- An air pump.
- Air line/hose.
- Air stones to create the small bubbles.
- Growing medium to hold plants

While the design is simple to reproduce, there are a few areas that need attention for it to work properly. Firstly, the plants need some sort of way to get oxygen. Since the roots are constantly submerged all day, no oxygen ever reaches the roots. To fix this problem, an air pump and an air stone can be used to create air bubbles on the surface of the reservoir. There are other ways to create air between the plants and the solution. Any contraption that drops water into the surface that breaks the surface tension of the nutrient solution would suffice as well. It would create small pockets of air that would be distributed across the surface of the solution.

2.5.5 – Aeroponics

Aeroponics is the more complex type of the hydroponics systems as it is less direct with its watering method. In aeroponics, the roots of the plants are left dangling in the air and receive nutrients by small droplets of water via misting. While this method is possible for home growing, it is generally for larger systems to take full advantage of the misting. Basic components that are used for this type of system are listed below.

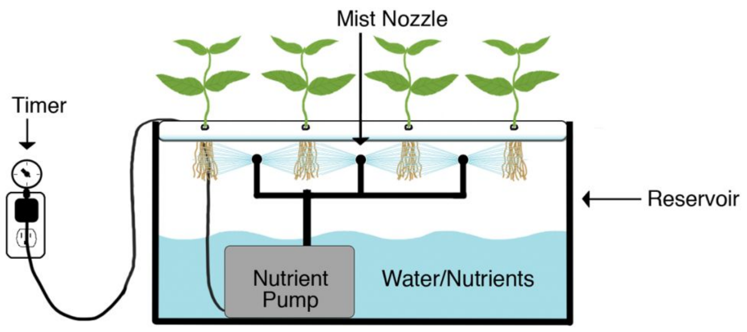


Figure 2.6 - Aeroponics Example

- Growing tubes or channels for the plants.
- A reservoir to hold the nutrient solution.
- A fountain or pond pump to pump nutrient solution.
- Tubing to reach the plants with mister heads.
- Enclosed growing chamber.
- A timer to turn the pump on and off.

Aeroponics starts off with a large reservoir where water fills the bottom portion. A pump is then used to push the nutrients up into nozzles and mist the roots that are in midair. Since water has adhesion properties, it will bind itself and coat the roots of the plants. This all takes place in one container, so the water droplets that fail to bind simply fall back down to the solution resting on the bottom. This eliminates the need for drains, a two-container system and an air stone. However, the misting nozzles can clog up over time, causing improper misting or no misting at all, and require some maintenance for plants that take longer to grow.

There are two types of aeroponic systems, low-pressure and high-pressure systems. Low-pressure systems are the most common subtype. The main reason is that it is cheaper to build and it is less messy than high pressure systems. Since the system is low pressure, each added nozzle greatly reduces the pressure of the spray for each nozzle. Because of this, low pressure systems are commonly found in home grows. Low pressure systems also create larger droplets due to their low pressure. Therefore, it’s best to have the nozzles positioned higher around the roots of the plants.

High pressure systems require a much higher amount of pressure, around 80 psi, to properly mist the plants roots. This is needed to atomize the water into the smallest droplets possible. The high pressure also allows the mist to reach deep into the roots from under them. The biggest drawbacks to this is the cost and it’s harder to have it functioning properly. The pump design would actually have to be replaced by an accumulator tank and compressed air. Otherwise a pump would wear out quickly and cost even more in the long run. More precautions would also be needed to make sure the roots are being reached by the mist.

2.5.6 – Wick System

The wick system is the easiest hydroponic system for it has no moving parts and doesn’t require any electricity to run. That said, it doesn’t use any pumps or air stones to move water or air. Its main feature is that it utilizes wicks to transport nutrients from a reservoir up to the plants in a two-container system. Despite being a two-container system, the plant container usually sits on top of the reservoir container with a hole cut out to minimize the distance the water has to travel through the wick. This system is very simple and is generally the go to hydroponic for kids and beginners alike. Basic components that are used for this type of system are listed below.

- A container for the plants.
- A reservoir to hold the nutrient solution.
- Wicking media like coco coir or vermiculite.
- Wicking rope or felt.

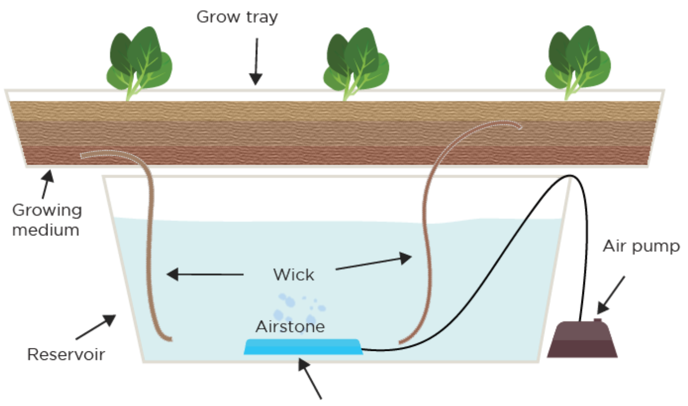


Figure 2.7 - Wick System Example

The wicks are made of absorbent material and are very porous. This is needed to transport the most amount of water possible since there isn't a pump to push it up and it's going against gravity. The wicks also need resistance from rotting as it would compromise the system over time. Since this hydroponic system uses wicks, it needs to dispel the water into a medium for the plants to absorb. Therefore, the plants are usually resting in absorbent growing mediums like vermiculite.

The biggest disadvantage of the wick system is that it isn't ideal for large plants. This is because the wicks don't supply a ton of water to the plants. Fruit bearing plants are especially going to struggle in this type of hydroponic system. Another disadvantage is that the nutrient will also take longer to reach the plants. This affects plants that require a high amount of nutrients. Lastly, plants absorb nutrients and water at different rates and a wick can't distinguish which is needed more for the plants. Therefore, there is no way to control the ratio of nutrient to water that is being delivered to the plants. This can lead to nutrient build and can be toxic to the plants. To mitigate this, the growing media and reservoir should be flushed with water to be replaced with a new solution.

3.0 – Research related to Project Definition

This section focuses on the research we have done on our project. Here we will discuss five main topics in the research and design process. In the first section we discuss our research on existing projects and products that we used as reference in order to learn and better understand the project and help guide our own design decisions. In the second section we discuss technologies that are relevant to our project and that helped show us the technologies we should implement. In the third section we discuss all the different components and parts that we will need and discuss a few options for each along with any other relevant research and information. Next, in section four, we discuss possible architectures and related diagrams that we can use for the project. Lastly, in section five, we discuss the design choices we have made. This involves our decisions on components such as the MCU and various sensors. All this along with any other information on our research and design process will be provided through this section of the report.

3.1 – Existing Similar Projects and Products

Research was done on similar projects and products that will help us aid in building our own hydroponics system. First, similar projects done by past groups by students at UCF and other universities were looked at to get some motivation to see what has been done in the past and what new can be added that can improve the technology. Some projects helped us to brainstorm ideas that were used to build our own system while others helped us resolve some issues that could possibly cause problems in the design and development stage. For example, the structural design for our hydroponic system was aided by one of the past projects.

3.1.1 – Energy Sustainable Hydroponics with Automated Reporting and Monitoring

This project was done by UCF students during the Spring/Summer of 2015. This project uses Nutrient film technique (NFT) hydroponic system where a water pump is used to control the flow of water, which passes through the roots of the plants in a timely manner as shown in Figure 3-1. The goal of this project was to make the system more user friendly by implementing a touch screen interface. This project implements a single board computer. The requirement for this project was to be able to read sensor data on a normal web browser and be able to send control data back through the MCU board through UART. The single board computer chosen was BeagleBone Black. It has the most open documentation and supports an open-source software and hardware platform. The system is built so minimum user input and care is needed. Everything from nutrient levels, water quality, water circulation, and lighting have been automated so a person can have his own hydroponics system without having to worry about time. It has a touchscreen LCD screen to set pH level, temperature, humidity, light and tank fill level. It also uses solar panels to power the system which is cheaper and more beneficial to the environment. While we have not implemented solar panels due to the complexity of the project, it is well worth the effort and time due to its benefits.

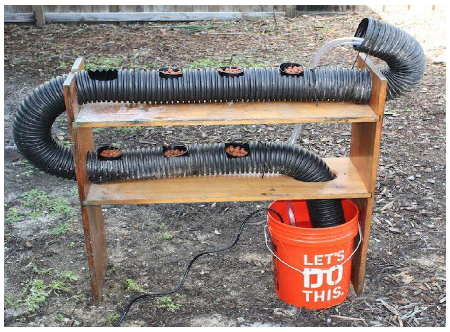


Figure 3.1-Energy Sustainable Hydroponics

3.1.2 – Waterwise Smart Hydroponics System

This project was done by UCF students in Spring/Summer 2016. The architecture design of this project, shown in Figure 3-2, looks very similar to our hydroponics system. We have also implemented a cabinet style structure at the bottom to hold the water tank and other components. The top will have PVC pipe that will provide support to the plants. This system is designed to be small enough for use in a housing environment such a studio apartment. It implements an LCD panel to let the user see the status of the plant. The project also implements an Android based application which will display various information about the system including the power system and the sensors which we are planning to implement in our own hydroponics system.

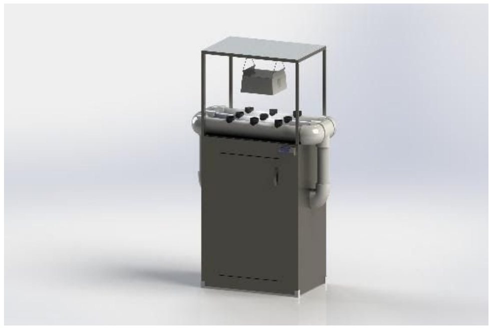


Figure 3.2 - Waterwise Hydroponics

3.1.3 – LeafAlone Hydroponics System

This project focused on building a household/backyard hydroponics system with a focus on low maintenance, energy sustainability, and great growing condition. This personal hydroponics system, shown in Figure 3-3, addresses issues such as inconvenience, cost, and prerequisite knowledge. The design used in this project is known as deep-water culture. It requires daily pH, nutrient level, water level, and temperature level testing. This project automates all these processes so a person with a busy lifestyle can own his own hydroponics system without having to worry about lengthy setup and daily maintenance. This project used an Arduino Atmega328p microprocessor. A web interface is implemented so a user can specify the plants wanting and can also monitor the sensor data. The web interface is connected to a microcontroller running the system. The plant specific settings and thresholds for each sensor have been calibrated into the microcontroller, eliminating any research the user needs to do. The electrical components have been powered by solar panels. This system also implements a 12V battery so that the microcontroller can have steady power supply even when the panels are not exposed to the sun. This project also includes a camera, nutrient dispenser, and a Wi-Fi module.

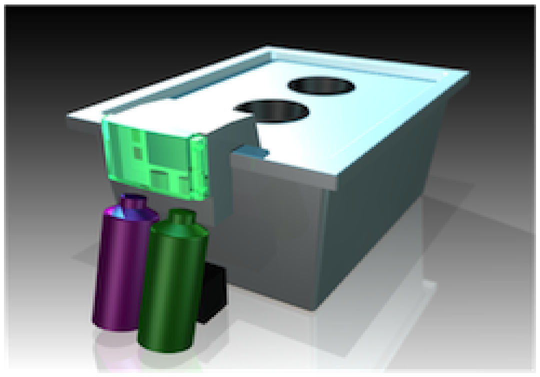


Figure 3.3- LeafAlone Hydroponics

3.1.4 – Niwa Hydroponics System

The NIWA is the first smartphone-controlled hydroponics system. The root system is supported using an inert growing media called rockwool. Rockwool is a lightweight hydroponics substrate made from spinning basaltic rock into fine fibers which are then formed into a range of cubes/blocks. The Niwa maker kit is available for hobbyists looking to make their own hydroponics system. The kit comes with electronics, actuators, and instructions on how to build. The NIWA one comes in 3 sizes: Niwa mini for $389, NIWA standard for $419, Niwa premium for $459, as shown in Figure 3-4. The small one grows products like herbs and small veggies. The standard size grows vegetables like tomatoes and peppers. The premium size can fit any home size and can also be used as home décor. The Niwa pro is also available and is a large-scale system of up to 50m^2. It comes with automation, 24/7 monitoring, and remote control making the system the most productive.

The Niwa system's core is powered by Spark board. The Spark board is controlled by an ARM Cortex M3 processor and the Wi-Fi module is a Texas instruments CC3000 network processor. The Niwa controls the light array, irrigation, ventilation, and climate control all through a mobile app. The app needs input of which plant is to be grown and it controls the rest of automation based on presets for that plant. The Niwa is like our product in a way that it has an app interface to automate the system and it is also powered through the AC power coming from the power outlets.

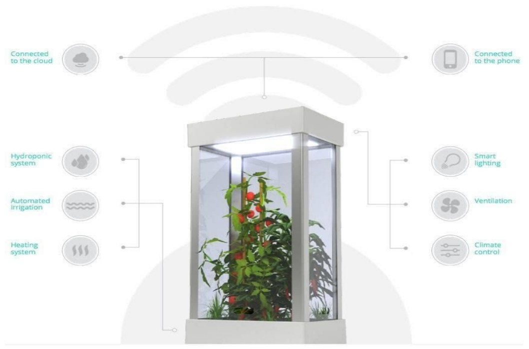


Figure 3.4- Niwa Hydroponics

3.1.5 – Cloudponics

This hydroponic system, as shown in Figure 3-5, allows the control of your growing plants by a remote app on your smart device. The app allows the users to monitor and automate the growing plant with nutrient dosing, pH sensing and climate control. This system monitors the humidity, light intensity, air temperature and water level. It uses a peristaltic pump to control the pH level and the nutrient level. This system allows for a peace of mind about your plants getting the daily requirements even when you are away from your home for a long period of time. This system automatically adjusts the climate and water levels for optimal growth. It uses 3 peristaltic pumps to allow the user to pump custom designed nutrients to the plant depending on the stage of your plants. By experimenting with different types of nutrients and the mix for your plants, the user can change what the plants need during different growing cycles.

EASY TO USE
Grow like a pro with Cloudponics.
Select what you want to grow and the system will nurture it from seed to harvest.
Follow your grow process on the app.

SAVE MONEY
Never again pay high prices for your medicine.
Cloudponics is a solution that pays for itself with the first grow.

Autonomy
You will be able to leave on a 4 week vacation without worrying about your plants, but you can check on them from your app anywhere in the world. Cloudponics gives you the peace of mind you need.

TRACEABILITY
By home growing your plants with Cloudponics you know for sure that it´s healthy.
The experience of watching your plants grow before your eyes and then consuming them is waiting for you.

16 OUNCES
Grow up to 16 ounces of your favorite medicine in just 3 months!
Yes, that's 5 grams per day.

BEST OF BOTH WORLDS
By merging Artificial Intelligence and skilled human growers that daily check your plants, our Personal Grow Trainer service allows your plants to grow under optimized conditions tailored specifically for them, improving the grow recipes constantly.

Figure 3.5- Cloudponics Hydroponics

3.1.6 – Growtronix

Growtronix is a fully automated in-door hydroponic control system. Their focus is for the user to be able to monitor and control every aspect of their system through a smart device like phone, laptop etc. It is fully customizable and up to 500 add-on sensors or controls can be added to the system. It has a variety of sensors which inform the users about the condition of the plant. Some of the sensors used are temperature sensor, humidity sensor, co2 sensor, pH sensor etc. The Growtronix base system is priced at $599.00 includes all the essential hardware and software for setting up a basic control system. It includes Growtronix Software Suite, Humidity/Temperature Sensor, controllable power outlets unit, Cables, and USB interface Module. The user then can add additional add-on hardware items like sensors, detectors etc. It has many more functionalities.

3.2 – Relevant Technologies

3.2.1 – Communication Technologies

Our smart hydroponic system will allow the end user to monitor the sensor readings and take proper course of action if something goes wrong. It will require the end user to have some form of wireless communication system. There are two technologies that have been researched to serve this purpose: Bluetooth and Wi-Fi. Either of these will be able to take the data from the microcontroller and be able to bridge the communication .0sdto the phone application. Different sensor readings such as pH level, water level, nutrient level will be available for the user to monitor.

Bluetooth

Bluetooth is a wireless technology standard for exchanging data between fixed and mobile devices over short distances using short-wavelength UHF radio waves. The IEEE standardized Bluetooth as IEEE 802.15.1. It operates in the 2.4 GHz ISM band and its fundamental purpose is to connect devices which are short distances from each other such as mobile phones and laptop computers. It is achieved by embedded low cost transceivers into the devices. Bluetooth can connect to 8 devices simultaneously and each device offers a unique 48-bit address from the IEEE 802 standard with connections being made point to point or multipoint.

Bluetooth network consists of Personal Area Network that can connect a minimum of 2 and maximum of 8 devices. It works in a slave-master configuration. A master device initiates the communication with other devices. The master devices govern the traffic between itself and the slave devices associated with it. Slave devices are required to synchronize the receive/transmit timing with that of the master. Thus, the microcontroller will be required to send the data, collected from the sensors, to a device with Bluetooth capability.

Wi-Fi

We will begin our discussion by talking about RF Transceiver. RF Transceiver is a device which includes both a transmitter and a receiver which works at the Radio Frequency. It is used to convert IF Frequency to RF Frequency and vice versa. It is a key building block for wireless and cellular operation. It is made up of RF mixers, amplifiers, pads and other RF components.

Now, we will dive into IEEE 802.11 protocol which specifies the set of media access control and physical layer protocols for implementing wireless local area networks. It uses high frequency radio waves on the unlicensed FCC frequency bands. Some of these bands are 2.4, 3.6 or 60 gigahertz. It is a digital communications module created and maintained by IEEE LAN/MAN standards committee. It consists of two layers of Open System Interconnect (OSI) model which provides the necessary protocols for the wireless systems to talk to each other. The two layers consist of media access control (MAC) and physical layer.

Layer 2 of the OSI model is the data link layer. It is a protocol layer that transfers data between nodes in the wide area network or local area network. Data link layer provides functional and procedural means to transfer data between network entities. It assigns the MAC address and performs encapsulation of network layer data packets into frames. The media access control determines who can access the media at any one time.

Physical layer is the first and lower layer in the OSI model. It consists of the electronic circuit transmission technologies of a network. It transmits raw bits rather than logical data packets over a physical data link. Physical layer provides an electrical, mechanical, and procedural interface to the transmission medium. Some of the major functions and services provided by the physical layer are bit-by-bit delivery, modulation, line coding etc.

Now, we will talk a little bit about Wi-Fi antennas. Antennas is the medium used to transmit and receive radio-frequency energy. It radiated radio waves when supplied with electric power and vice versa. Wi-Fi enabled devices have built-in antennas which enable radio communication with other Wi-Fi enabled devices. Some antennas are mounted externally as seen on Wi-Fi routers.

3.2.2 – Lighting

In this section, we will discuss three types of lights that are commonly used in hydroponics. Using an artificial lighting system over the sun has a few benefits. By using electric lighting, we can provide the right amount of light to the plants based on the ideal growing conditions. Also, this way the plant can receive light 24/7 without having to worry about good weather. The three types of lights that hydroponics gardeners used are Compact Fluorescent Lighting (CFL), High Intensity Discharge (HID), and Light-Emitting Diode (LED). Each has its own advantages and disadvantages with different sizes, prices, and quality levels.

High Intensity Discharge Lighting

HID lights have been used in hydroponics for a long time. They come in two different types, High Pressure Sodium (HPS) and Metal Halide (MH). HID lamps are a type of gas-discharge lamp which produces light by sending an electrical discharge between two electrodes and through a plasma, or ionized gas. An additional gas is used which serves as an easy way to classify between the major types of HID lamps: Mercury, sodium, and metal halide. These lamps are best known for their long-rated life span and high efficiency conversion of electricity to light.

The basic technology of this lamp has existed for over 300 years. The invention of the gas-discharge lamp is credited to Francis Hauskbee who demonstrated the technology in 1705. During that time, the lamp was filled with air but later discovered that light output can be increased by filling the lamp with noble gases such as neon, xeon, argon or krypton. Modern HID lamps have been able to improve the light output further through experimentation in gas mixtures and improved electrodes. In modern HID lamps, an electric arc is sent between two tungsten electrodes which are housed in an arc tube constructed of quartz as shown in Figure 3-6. An arc is created by the initial surge of electricity which heats the metal salts creating plasma.

Metal Halide lamps are used if price is a constraint and these systems give the plants the full spectrum of light they need. Metal halide lighting system consists of four parts: bulb itself, a reflector hood, and a ballast. The downside of HID lamps is that it produces a lot of heat compared to LED lights. This heat can cause the plant to damage if there is no cooling system in place. The heat can be controlled by using a reflector that is vented to a ductwork section that blows the hot air outside using a fan. But this requires extra time and effort which is not needed in an LED lighting system which produces little to no heat. Further, HID lamps are very inefficient and can cause the electric bill to skyrocket.

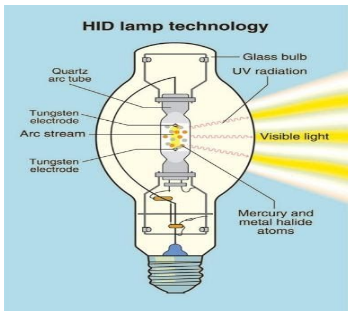


Figure 3.6 - HID lamp

Fluorescent lighting

This technology is like HID lamps. It is a low-pressure mercury-vapor gas discharge that uses fluorescence to produce visible light. An electric current in the gas excites mercury vapor which produces short-wavelength ultraviolet light causing a phosphor coating on the inside of the lamp to grow, as shown in Figure 3-7. Fluorescent lighting is good for green leaf plants like lettuce, spinach and herbs. They are also cheaper and last longer than HID lamps. The main disadvantage is it is not very powerful and doesn't have the full spectrum of lighting needed to grow plants with flowers. These lamps are high in demand for those interested in growing juvenile plants because unlike HID lamps, the worry of scorching the plants or removing the hot air doesn't exist. Many times, growers will use fluorescent lights for juvenile plants and light once the plants are mature enough and need extra light to produce flowers or fruits.

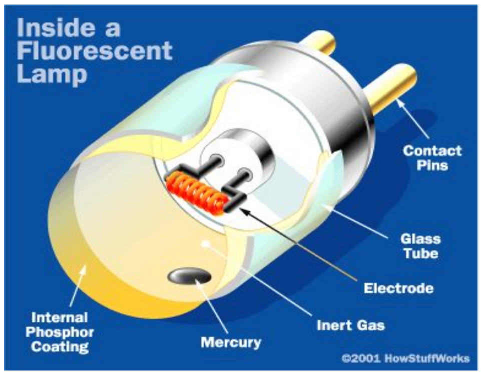


Figure 3.7- Fluorescent lamp

Light Emitting Diodes (LED)

LED is a semiconductor light source that emits light when current flows through it. Electronics in the semiconductor recombine with the holes, positively charged, releasing energy in the form of photons. The color of light is dependent on the band gap of the semiconductor. The band gap of a semiconductor is the energy required to excite an electron up to a state in the conduction band where they are free to move around. LED bulbs produce very low heat and can cover the full spectrum of light needed for plants to grow. They only tend to last a very long time compared to any other kind of bulbs. It is also very energy efficient and can be customized for small or large gardens. The only disadvantage is it is expensive to purchase up front but saves a lot of money in the long run.

3.2.3 Battery Storage

Battery capacities are measured in Amp-hour (Ah) which describes the amount of current a certain battery can supply for a certain number of hours. It is important to choose a battery which will provide enough power in case of a power outage. The characteristics of different batteries in respect to their environment, deep discharging abilities, total size and their capacity. Choosing the right battery is critical for reasons such as safety, cost and reliability of the power supply system. We will talk about three main types of batteries:

Lithium-ion Battery

Lithium-ion Batteries are commonly used in portable electronics and electric vehicles. In these types of batteries, lithium ions move from the negative electrode through an electrolyte to the positive electrode during discharge, and back when charging. They used an intercalated lithium compound as the material at the positive electrode and typically graphite at the negative electrode. These batteries have a high energy density and low self-discharge. There is a safety hazard since it contains a flammable electrolyte.

Lead-Acid Battery

Lead-Acid Battery was invented in 1859 by French physicist Gaston Plante. This battery can supply high surge currents and the cells have a relatively large power-to-weight ratio. This battery is dependable and inexpensive on a cost-per-watt base. The grid structure of the lead acid battery is made from a lead alloy. Lead acid is heavy and is less durable than nickel- and lithium-based when deep cycled. Lead acid batteries have a moderate life span and work well at cold temperatures.
The lead-Acid battery self-discharge is low being between 2.5% and 5% per month and depends on the state of charge, composition of the electrodes, and other factors. The longest lasting lead-acid battery in terms of charges and discharges for cycling

applications is the tubular plates battery, which can last for 1800 cycles during a 50% depth of discharge. Other types of batteries such as Gel battery or AGM battery lasted for 650 cycles and 450 cycles.

Nickel Metal Hydride Battery:

The chemical reaction at the positive electrode uses nickel oxide hydroxide (NiOOH). The negative electrodes use a hydrogen-absorbing alloy instead of cadmium. A NiMH battery can have two or three times the capacity of an equivalent size NiCd and its energy density of a lithium-ion battery. It was originally designed to replace nickel cadmium battery technology because of studies that prove how detrimental cadmium is to the environment. These batteries can get 1000-2000 cycles in their lifetime that makes them a good battery for applications that require cycling like portable electronics. These batteries are used for hybrid electric vehicles like its use in Toyota Prius.

3.3 – Strategic Components and Part Selections

Now we will discuss the specific components and parts we will need to execute this project. We need components that will achieve the goals of our project and will provide us with flexibility. We also want to have components that are easy to work with and are all compatible with each other as this will be one large interconnected system. Our research on this process continues below in the following part specific sections.

3.3.1 – Microcontroller

The microcontroller (MCU) is the central and key component of our system. A microcontroller will be the brains of our system that will handle all the information needed by connecting and controlling all the sensors and various power systems together. Here we outline various requirements and design choices we will consider in our search for an MCU along with some options for MCUs that we considered for the project. There are many things to consider when picking a microcontroller. For this project we will need to consider several specifications that a microcontroller must support/implement in order to be successful in achieving the project's goals. Some things we are considering include the number of pins, storage and system memory, clock speed, supported software, power efficiency and consumption.

The number of pins on the microcontroller is very important. These pins are what will allow the microcontroller to connect to and communicate with the various sensors and components of the project. The number of pins dictates the number of subsystems and components we can attach to the MCU. There are often many models of a certain type of MCU so it's possible that if an MCU that fits the needs is lacking in pins there is a bigger model with more pins. There are also components that can be added to MCUs to increase the pin count but this has the potential to add complexity. This isn’t out of the question for this project and is an option we considered in the process.

We also care about the system's overall efficiency with power consumption. We need the system to be powerful enough to handle all the computation and run all the various components without wasting too much power. We want to conserve energy and use power in an effective and efficient manner.

The systems clock speed and storage/memory are some other major components we consider in the design. We want the system to be able to run the software and handle all the components we need to control the system. Clock speed can lead to faster execution and computation. Although clock speed is the simplest way to increase the MCU effectiveness in speed it’s important to note that we understand that clock speed is not everything. Different designs can still be better even with lower clock speeds depending on the design of the system such as the number of stages in the data path. We also care about the system's storage and memory. We need the system to have enough storage for the software/code we are running along with we need fast memory such that data is transferred as quickly as possible.

Outside of what we have discussed, it's important that the microcontroller we use for this project supports all the various basic functions and protocols we should expect from an MCU. Or at least have everything we will need to execute this project. This includes features such as communication protocols, programming languages, and architecture.

Outside of just MCUs we also consider MCU development boards and kits. Many manufactures such as Texas Instruments produce development boards which include MCUs along with other various features such as I/O ports, extra pins, built in modules for systems such as Wi-Fi and Bluetooth, and other input devices such as LEDs and physical switches and buttons. Using a development board over a MCU by itself can allow for easier implementation and save us time and energy in the development/design process.

Texas Instruments MSP430

The first microcontroller we considered is the MSP430 by Texas instruments. Simply being that we had used this board through multiple courses in our curriculum and we have become more familiar with it. There are many different versions of the board with different ranging clock speeds, memory, and extra features. The MSP430 can have up to 512KB of memory along with GPIO pin options up to 87 pins, depending on the model. The CPU runs at 16 MHz and supports many communication protocols that would be applicable to this project. The models we will most likely have access to The MSP430 microcontroller family also features support for many different peripherals that can be added in to implement different features such as support for specific types of digital and analog sensors as well as specific systems for measuring gas and water. The MSP430 is also a low power option as it has features in place that can allow us to run the system in different power modes to effectively save power when needed.

Texas Instrument C2000

We continue with the Texas instruments family of products by looking into their line of real-time MCUs called the C2000 series. This microcontroller is based on a Dual-core C28x architecture with a frequency of 200Mhz and features up to 169 GPIO pins. The processing power compared to the MSP430 is definitely greater with its higher frequency and with it featuring up to 1.5MB of flash memory (split between the two cores) there is potential for more intense tasks. The C2000 supports four independent 16-bit ADCs which can provide efficient and precise management of multiple analog signals which can help ultimately increase system throughput. Along with this the C2000 supports many of the protocols needed to make the sensors work together and read and measure information. The C2000s embedded real time analysis and diagnostic module allows the analysis and debugging capabilities which can help save us both time and potentially money. Overall, the C2000 is a more powerful option compared to that of the C2000 which has support for more GPIOs than that of the MSP430, overall leading to larger support for sensors and peripherals. It’s possible that all its more advanced architecture, dual core CPU, and various innovations may prove more difficult to work with as well as potentially be overkill.

Texas Instruments Tiva-C Arm Cortex-M4F

We then moved onto a third microcontroller by Texas instruments that is based on the Tiva-C series. This processor is based on an ARM Cortex-M4F processor core. It operates at 120-MHZ with 150DMIPS performance. Supports 1024KB flash memory and has two 12 bit analog-to-digital converters. This is a more powerful option compared to the MSP430 but isn’t, on paper, better than the C2000 series we discussed before. It does use an ARM processor which has its own benefits and has a large catalog of information for the Tiva C series microcontrollers. There is support for up to 140 pins based on configuration which much like the C2000 should provide us with plenty of GPIO pins to work with for all of our sensors. One benefit of this being based on ARM is that the efficiency is great which helps reduce the number of system instructions and save energy doing so. This seems like a more advanced option compared to the MSP430 but not as complex as the C2000.

Atmel AtMega328

Another popular microcontroller manufacturer is Atmel. We saw that many past projects had used Atmel's MCUs and chose to look into their product line of MCUs. The first we came upon was the AtMega328. The AtMega328 is an 8-bit RISC based microcontroller that has 32KB of ISP flash memory and 2KB of SRAM. There are 23 GPIO pins/lines. The CPU speed is 20 MHz and with the instruction implementation reaches 1 MIPS per MHz which should provide a good balance of power consumption and overall performance. The device supports all expected protocols much like the other microcontrollers and much like the MSP430 supports many power saving modes leading to a power efficient design. The AtMega328 is the MCU used in many Arduino Uno boards and has a lot of documentation much like the MSP430.

Atmel AtMega2560

Another option in the Atmel lineup is the AtMega 2560 which is also an 8-bit RISC based microcontroller. The design is very similar to that of the Mega328 but there are increased specifications. There is now support for 256KB ISP flash memory,86 GPIO pins/lines. The frequency is less than that of the Mega328 at 16Mhz but this is made up for in just overall MCU processing power including its 8KB of ram versus the Mega328s 2KB. With this design the device achieves a throughput of 16 MIPS at 16 MHz Overall this is just a much more powerful version of the Mega328 we looked at before which could lead to more flexibility in the design and implementation. This chip much like the Mega328 also is used in Arduino boards but specifically the Arduino Mega.

Atmel SAM C Cortex-M0+

A third option from Atmel we noticed was their own ARM based microcontrollers which now gives us another option if we want to focus on a design that is ARM based. There are many ranges of options for these processors, all being 32-bits but for the sake of this project we focused on the lower complexity models as a result of the fact that we most likely won't need all that computation power. The specific model we looked at is the SAM C21 which are 32-bit processors that run at a core clock of 48MHz. The SAM C has 256KB of flash memory and 32KB of SRAM. There is support for up to 84 GPIO pins for sensors and peripherals. This is a more powerful option compared to the Mega328 and Mega2560 and has the benefits of the ARM architecture. If Atmel seems like the environment that we are leading towards and the other options aren’t enough for our design. The SAM C line of microcontrollers is definitely a viable option.

Brief Summary of Options

So far, we have discussed six different microcontrollers across two different manufacturers. There are many versions of each MCU with slight variations in each. When deciding on a controller it is important to look at the different variations in each model to make sure that it has all the required features. We have some more power conservative options which are very efficient such as the MSP430 and the Atmega328 while we also have more powerful options such as the Texas instruments C2000. Picking a good microcontroller for this project will entail balancing the systems performance vs the systems power usage. Along with this we need to consider the amount of documentation and information on the MCU that is online for reference and design. This will be useful when implementing it on a custom PCB. Out of technical specifications and documentation we need to consider availability and our own experiences to see which will be best for our group. Maybe going with a more powerful MCU despite its power usage is worth it since it could allow for less overall system complexity and leave more room for us to work with.

3.3.2 – pH Sensors

One thing we will need for the hydroponics system is a way to monitor the water's pH. Which will play an important role in maintaining the water quality of our hydroponics system. The pH level of a liquid solution is a measure of how acidic or basic it is. Scientifically speaking it is the measure of the concentration of hydrogen ions in the solution. pH is represented by levels on a scale from 0 to 14. A pH measurement below 7 indicates the liquid is considered acidic while a pH measurement above 7 indicates a liquid is considered basic. If a pH measurement is 7 then the liquid is considered neutral. A lower pH value indicates a higher concentration of hydrogen ions in the liquid solution. The importance of pH in hydroponics is a result of its effect on the availability of nutrients in the water, such as iron. Different plants will grow best at different pH ranges based on their specific nutrient requirements and how those minerals react at specific pH levels. Different minerals and nutrients in water can chemically react with their environment differently based on the concentration of the hydrogen irons. These reactions on the environment can change the state of nutrients that the plant relies on, potentially changing the nutrient to a state that is unusable to the plant. So, the plant requires the pH be at a level that keeps its required nutrients in their correct states to be used by the plants. With all of this in mind, a pH sensor is a key component in hydroponics and as a result will be important to this project. We need a sensor that will measure the pH of the water such that we can take that measurement and handle it accordingly. This will help make sure that the plants can get their required nutrients properly. A few things to consider when picking a pH sensor are compatibility, cost, and whether the sensor digital or analog. It's important to ensure that a sensor will be compatible with whichever microcontroller we pick. Along with this we will focus on the decision of using either a digital or analog sensor and weigh the benefits of each. Lastly cost will be considered within reason, meaning that if an objectively worse sensor is the most affordable it can be worth the extra cost since a potentially more efficient and easier to work with could save us time and money somewhere else in the long run.

Atlas Scientific EZO-pH Embedded Circuit

This model pH sensor is a digital sensor that supports both UART and I2C protocol for communicating with the microcontroller. It is compatible with any off-the-shelf pH probe/electrode. It requires calibration when setup and needs calibration once a year to maintain reliability. It can read a full range of pH values from 0.001 to 14.000. The pH readings have an accuracy of +/- 0.02. Its reading can either be temperature independent or dependent. There is support for both single reading and continuous reading modes. The data format produced by the sensor is in ASCII. With this sensor and with a compatible probe we could effectively and reliably measure the pH of the water in our system. There is a good amount of documentation on this sensor along with there being built in support for different models of Arduino and raspberry pi. This is useful since some of the microcontrollers we are considering are natively used in many Arduinos so having that support should make connecting it to our microcontroller an easier process.

Atlas Scientific Gravity Analog pH Sensor/Meter

This pH sensor is made by the same people as the previous one but rather than being based on providing us digital signals it reads and provides analog signals. In this case rather than getting an actual reading of the pH we would get a voltage in which we would convert to the pH using a formula provided in this sensor's datasheet. This sensor gives a range of accuracy of 0.1 to 14.0 with an accuracy of +/- 0.2. This sensor does not require calibration and much like the other sensor supports most types of probes that can be connected to the sensor. There is also a decent amount of documentation on this design and some examples of Arduino implementations for reference.

Reland Sun Liquid PH0-14 Detect Sensor Module

This is a pH sensor made by the brand Reland Sun. This module outputs an analog signal for the microcontroller to interpret. The analog signal can detect pH ranges between 0 and 14 while having a power consumption less than 0.5 W. The system's response time according to sellers such as AliExpress confirms it to be less than or equal to 5 seconds. This module is a much more affordable option compared to the other 2 atlas scientific options but that doesn’t come without some tradeoffs. There is little to no documentation that can be found for the device. No schematics or diagrams and there is less information on it. This would require reliance on user used info from troubleshooting. This may not be an issue if we can get help from others on how to interpret the sensor configuration such that we could attach it to our microcontroller.

pH Sensor considerations

For the pH sensor we are left with a few options. We have both digital and analog options. Both digital and analog can be useful for the scope of this project and it will primarily come down to design decisions, our microcontroller, and cost. Atlas scientific gives us some appealing options which are relatively well documented and have clear support for microcontrollers we considered. Despite this the sensors are relatively costly compared to our third option from Reland Sun. Reland suns option is another analog option and has a much lower up-front cost but would require ordering from vendors such as AliExpress and waiting potentially two months for it to arrive. This along with the lack of documentation makes this a more difficult decision then it potentially could be. So, we as a group will weigh cost, functionality, documentation, and flexibility when choosing from these sensors for the hydroponics system.

3.3.3 – Temperature and Humidity Sensors

One thing that will be important to measure for the sake of the health of the plants is the ambient air temperature. We need to ensure that the temperature of the air for the plants is within a feasible range for the plants to prosper and grow. Some plants have evolved based on their native climates and prefer warmer temperatures while other plants that originate from colder climates. Now today, as a result of genetic modification and specialized farming, some plants have become more adapted to growing across a broader range of temperatures. Despite this there is still enough of a variation for temperature to be an important metric and we also want the system to support plants that may not be as flexible and do require very specific temperatures. When picking a sensor, we will focus on a few parameters. We want the sensor to support a large range of temperatures such that we can read both low and high temperature environments.

We also want the sensor to be accurate such that it supports precise measurements of at least +/- 1 degree. We also will consider its response time of the signal along with what available communication protocols are supported.

Atlas Scientific EZO-RTD Resistance Temperature Detector

This is a Temperature sensor made by Atlas Scientific. This sensor only reads temperature and we would need a separate sensor in order to read humidity. It supports a range of -126 degrees Celsius up to 1254 degrees Celsius. This is a large range and should be plenty enough for the project, if anything its range is overkill for the scope of hydroponics. It has an accuracy of +/- 0.10 degrees Celsius and has a response time of 1 reading per second. The sensor supports outputting temperate readings in Fahrenheit, Celsius, and kelvin. There is support for UART and I2C protocols with initial support and documentation for use with Arduino and raspberry pi. If we use microcontrollers based in both these development board designs, we can use this documentation and support to more easily implement it into our project. Overall, this temperature sensor has a lot of features and options as well as being very accurate and supporting a wide range of values. Despite this, its cost is relatively high compared to some other sensors on the market as well as its temperature accuracy and range may be overkill for some projects. This is a case of if we need the extra accuracy and features in order to make the cost worth it.

SMAKN DHT22 / AM2302 Temperature and Humidity Sensor

This is a sensor module made by SMAKN that incorporates the DHT22 sensor from adafruit. This sensor supports temperature readings within a range of -40 to 80 degrees Celsius with an accuracy of +/-0.4 degrees, which should be plenty of scope for the sake of this hydroponics system. The sensor also can take humidity readings between 0% up to 100% with a 2-5% accuracy. The circuit supports up to a 0.5Hz sampling rate which translates roughly to once every 2 seconds. The sensor output is digital and can be read from the 3 pins extending from the board. The entire sensor module should output both temperature and humidity readings which can be connected to the microcontrollers GPIO pins. This is a low-cost option such that we get two sensors in one module as well as the overall sensor cost is lower than other options.

3.3.4 – Water Temperature Sensors

Outside of air temperature another thing to consider is the temperature of the water. The temperature of the water can have a large impact on the performance when it comes to plant growth in a hydroponics system. Many things that are important to the plant are dependent on temperature. One of which is the amount of dissolved oxygen in the water. As water temperature increases the amount of dissolved oxygen in the system decreases. This can directly affect plant growth and a plants ability to absorb nutrients from the water. This is the result of the reliance of plants on using oxygen for aerobic respiration. For a hydroponics system this is primarily done through the roots from the nutrient solution in the water. When the plants don't get enough oxygen, the roots become more permeable and can take in less water which in turn leads to less nutrient intake. We also must consider that if the water is too hot bacteria and fungus can form which can be harmful to the plants. Cold temperatures also hinder plant growth and can cause plants to effectively shut down and stop taking in nutrients. As a result of these effects, it can be important to monitor the temperature of the water as we want to make sure the temperature is within a range that sets up the plants for the best growth potential possible. The project design is primarily focused for indoor use so the extremes of this should be as big of an issue but it can be important to monitor nonetheless in case of sudden fluctuations.

DS18B20 Waterproof Temperature Sensor

This is a pre-wired and waterproofed sensor. The sensor is good up until 125 degrees but is best to be kept under 100 degrees as a result of the cable jacketed material. The signal output is digital and as a result there isn't any signal degradation over the long distance of the wire. The temperature precision is +/- 0.5 degrees and allows for up to 12 bits of precision. Can be used with a single digital pin and allows connection of multiple to the same pin since each one has a unique 64-bit ID designated at the factory to differentiate them. A 4.7k resistor is required as a pullup from eh DATA to VCC line when implementing the sensor. The sensor is based on the Dallas 1-Wire protocol, which is considered complex and can be difficult to parse the communication in code. Despite this, there are examples online and we feel like we would be able to follow along and figure it out for the sake of this project. There is also an adapter module designed for the Arduino that we could implement if it made connecting it to our microcontroller simpler along with simplifying the PCB design. Overall this is an affordable option that has a simple design that doesn't use up too many pins and is accurate enough for the sake of this hydroponic system.

3.3.6 – Light Sensors

All plants require some form of light in order to execute the process needed for photosynthesis. Certain plants require different amounts of light intensity along with different periods of lighting. Some plants prefer short periods of light with long periods of darkness while others prefer long 18-hour periods of sunlight per day. In order to make sure plants are receiving the proper amount of light per day along with the correct intensity of light we will need a light sensor. The light sensor we need for hydroponics will be used to measure the lights intensity in order to ensure the overall amount of light at any point in time. Along with that we will use this data in order to keep track of day/night cycles for specific plant types. When choosing a sensor, we want it to be precise and support a large range of light intensity readings and would prefer data output to be easily transferable to data in our code for the microcontroller.

Tenstar Robot BH1750 Light Intensity Illumination Module

The BH1750 is a light intensity sensor board. It has a built in 16-bit AD converter which allows us to directly output a digital signal. This digital signal is in Lux (Lx) which minimizes the use of calculations in our result which in turn can make our readings more accurate. The sensor supports a range of readings between 0 and 65535 Lux.
The sender’s readings have a precision of 1 Lux and can be performed on an overall wide range of brightness. The sensor is based on standard NXP IIC communication protocol and runs on a voltage between 3 and 5 volts. There is a good amount of documentation online along with sample C code for connecting it to the microcontroller.

Adafruit TSL2591 High Dynamic Range Light Sensor

Adafruit’s TSL2591 light sensor is an advanced digital light sensor and supports a large range of light measurement applications. This sensor supports a large range with 188 uLux all the way up to 88,000 Lux and can be configured for different gain/timing ranges. This sensor features both infrared and full spectrum diodes. This allows the system to measure infrared, full-spectrum or human-visible light. This sensor has a large dynamic range of 600,000,000:1 and operates over I2C communication. There is even a built in ADC which allows this sensor to be used with digital inputs. Power efficiency is also great with a low current draw which can be good for low power implementations drawing as low a 5 uA when power-downed.

RobotDyn Photosensitive Light Sensor - Analog and Digital

This light sensor module is based on a light dependent resistor. Generally, the resistance will decrease when the ambient light intensity increases. This translates to mean that the module will be HIGH in bright light and LOW in dark lighting. This particular module can output both analog and digital signals. The analog signal will output the real-time output voltage of the photoresistor. The digital output will output high and low signals based on reaching a certain threshold. This threshold can be adjusted via a potentiometer on the sensor. The range of Lux is not specified for this sensor but it is assumed it supports a typical range that would be usable for this project. Reviews and users report successful usage with the sensor so we feel it’s a safe option to consider. Despite this the sensor is cost efficient and could be tested for its limits once we used it.

3.3.7 – Water Level Sensors

Water is one of the main components in a hydroponic system. It is the medium in which the roots of plants get all of their required nutrients. Water will naturally evaporate overtime and if the water runs out, the hydroponic system essentially breaks. Low water leads to plants not getting the nutrients they need and can lead to drying out and slow down plant growth. With this in mind it’s important to make sure there is enough water in the system for the plants. We need to implement a way to track that the water level/volume is within a working range and have a way for reporting this feedback to the rest of the system. To do this we need a water level sensor. This sensor will need to make sure that the water level/volume doesn’t drop below a certain point. This ensures the roots stay submerged. The sensor will be in the systems water reservoir and will report once the water drops telling us the system needs more water. The specific level at any given time isn’t necessarily needed. The main function is to track if the level drops below a specified threshold.

CQRobot Contact Water/Liquid Level Sensor

This is a photoelectric water/liquid sensor that operates based upon using optical principles. This particular sensor is designed to work with Arduino and raspberry pi which should be transferable to any microcontroller we pick. The sensor has a DIP switch which allows the voltage level for the high output to change between 3.3 volts and 5 volts. The low output is less than 0.1 volts. The sensor can operate over a large range of temperature environments and has no specified measuring range, according to the manufacturer documentation that is no measurement range limit. The sensor has a detection accuracy of +/- 0.5mm. The sensor design is a module which connects to our microcontroller, then this module connects to a probe which goes into the reservoir where the water is located.

eTape Liquid Level Sensor

Another sensor option would be the eTape liquid level sensor by adafruit. This is a solid-state sensor which outputs a resistance that varies with the level of the fluid. The sensor operates based on the hydrostatic pressure on the sensor’s envelope caused by the liquid it is immersed in. This results in a change in the distance between the top of the sensor to the surface of the liquid. The resistive output is inversely proportional to the height of the liquid. This indicates that at lower liquid levels the resistance is higher while when the liquid is higher the resistance output is lower. This sensor comes in different lengths which can be used for different depths and operates within standard temp ranges between 15 and 150-degrees Fahrenheit or -9 to 75 degrees Celsius. The sensor has an actuation depth of nominal 25.4mm which is roughly 1 inch. The sensor outputs a reference range of 400-2000 ohms with an accuracy of +/- 20 %. The sensor works as a resistive divider / analog out so it is best to read on an ADC pin on a microcontroller. The sensor uses three wires with one connecting to a 3 volt or 5-volt pin, one to the ground, and the third where we measure our output for the analog diver.

3.3.8 – TDS Embedded Conductivity Sensors

A major key factor in hydroponics is how the plants get their nutrients. In hydroponics plants get their nutrients through the water in which their roots are submerged. As a result, nutrients must be supplied to the water in most cases to ensure the plants are getting their required nutrients. Over time the nutrients in the water can be used up by the plants and will require the system to be replenished. As a result, it can be important to monitor the nutrient levels in a hydroponic system. One way to do this is to measure the electrical conductivity of a solution. This is a measure of a solution's ability to carry an electrical current. In a solution there are many different types of dissolved solids that can contribute to its ability to carry a current. The primary contributor to a solution's conductivity is dissolved salts. This is a result of the fact that when these salts dissolve in a solution they dissociate into positive and negative ions which can both effectively carry a current. The dissolved salts that are applicable to hydroponics and to growing plants include calcium, phosphate, nitrate, sulfate, and potassium. Each of these are added to the water to form our nutrient solution which can act as sort of a fertilizer for the plants in our system. Knowing the nutrients in our system and how they support the conductivity of a solution, if we can effectively measure a system electrical conductivity, we should be able to infer information about the amount of nutrients in our system. In short, we want to measure the systems TDS, total dissolved solids. The more dissolved solids in the water should indicate that there are a higher amount of dissolved salts and as we have discovered this indicates higher conductivity. For this hydroponics project we want a sensor that can read us either the TDS or directly tell us the waters conductivity. We will use this information to keep track of a system's nutrient levels. If the levels are low, we want to indicate that more nutrients need to be added to the system. If the nutrients are high, we want to make sure there aren’t excess amounts in the system. If the nutrient levels are low the plants could experience nutrient deficiencies and have slow growth rates. If the nutrient levels are too high there is a chance the plants can be burned or even killed by the amount of salts and minerals in the solution. Considering we want to make sure our TDS levels are within a safe range and we need to read TDS or conductivity we need a specific type of sensor. Generally, the way a sensor for measuring TDS is composed with three components. Sensor module which handles the connection to the microcontroller and interfaces with 2 probes. These two probe sensors use something called an amperometric measurement. These types of sensors operate by applying alternating voltage to the two electrodes. The amount of current carried between these two electrodes is based on the salts present in the solution. We measure the current carried which can indicate the conductivity and in turn tell us the TDS of the solution. There are some other technologies which even use more probes or have combined the two-pole design into 1 probe. We will be looking into the one and two probe designs for the sake of this project as they have greater simplicity and require the least amount of electrical components

Atlas Scientific EZO-EC Embedded Conductivity sensor

This is a high accuracy digital sensor that supports UART and I2C communication protocols. This sensor has high accuracy and works based on the two-probe design. It does support probes of any brand as long as they are K0.1 – K10. This sensor can be used to read 4 specific numeric values depending on the use case. It can ready conductivity in uS/cm, TDS in ppm, salinity in PSU, and specific gravity which only applies to salt water. For the sake of hydroponics, we are only concerned with the conductivity and/or TDS readings of the sensor. The sensor also has a response time of 1 reading per second which is plenty for the sake of this project. The readings are accurate within +/- 2% of actual value. The sensor module can read values within a range of 0.07 to 500,000 uS/cm. There is a good amount of documentation and has Arduino examples online. This is a powerful yet expensive option that provides precise readings.

DFRobot - Gravity: Analog TDS Sensor/Meter

This sensor measures the TDS value of a liquid in a relatively affordable fashion compared to other sensors on the market such as the atlas scientific once we previously discussed. This sensor uses a single probe called the TDS pen in which it normally can be bought with. This saves on the number of parts needed and with the lower cast can save us up to 75% the cost compared to our atlas scientific option. This sensor supports input voltages between 3.3 volts and 5.5 volts and outputs between 0- and 2.3-volts analog voltage. This in turn makes it compatible with either 5 volt or 3.3-volt control systems. This sensor reads TDS on a range of 0 to 1000ppm with an accuracy of
+/-10%. This sensor design uses a single probe with 2 needles to measure the conductivity in which it then reports a TDS reading to the output signal. The sensor is designed around systems like the Arduino and should make implementation with our

microcontroller a smoother process. This along with examples and documentation should make it a viable option, especially with its relatively low cost.

3.3.9 – Dissolved Oxygen Sensor

Oxygen along with CO2 is a key element that plants use in order to respirate and commence photosynthesis. In a hydroponic system plants, most get the oxygen they need from dissolved oxygen in the water, through their roots. Often when people think about plants, they think about how they intake carbon dioxide and produce oxygen, which is true, but plants also need oxygen in order to respirate. It just happens that plants produce more oxygen through photosynthesis then they personally consume. Oxygen is one of the key nutrients that plants need to grow. The amount of oxygen needed can vary depending on the type of plants, if you are trying to make them flower, if they are in a vegetative state, etc. This is often a result of the root size changes and requirements for the root size. Lack of oxygen in a system can lead to wilting of the leaves and make the roots less permeable which can lead to less nutrient intake and potential buildup of toxins. Despite oxygen being an important part of the process, plant health is often determined through pH and TDS, with dissolved oxygen being used less often in the business as a result of cost vs benefit. Dissolved oxygen sensors often cost way more then the TDS and pH sensor systems which together can accurately give hydroponic farms an idea of how plants are doing without investing in a dissolved oxygen sensor system. As a result, we are considering implementing a dissolved oxygen sensor as it helps monitor the plant's health and with it being used less in the industry it could be a differentiating feature in our hydronic system. As a result of the expected high cost we will consider our options and will choose to implement or not after getting other features designed, planned, and implemented. We will discuss a few options below and overall consider our options when implementing it. If we aren’t able to afford the cost to implement this sensor monitoring feature for this hydroponic system, it would definitely be something we would want to implement given more time and financial support.

Atlas Scientific EZO-DO Dissolved Oxygen Embedded Circuit

One option for a dissolved oxygen sensor is Atlas Scientific EZO-DO embedded dissolved oxygen circuit. It requires a specialized galvanic probe which is the most expensive part of the sensor system. This probe consists of a special membrane called PTFE, and anode and a cathode. Oxygen molecules will go through the membrane and reach the cathode in which a voltage is produced in mV. If no oxygen is found the output will be zero. The output increases with the amount of oxygen. This sensor supports temperature, salinity, and pressure compensation and works with both UART and I2C data protocols. Data output is in ASCII and the system operates between 3.3 volts and 5 volts. The sensor can read within a range of 0.01 to 100+ mg/L and 0.1 to 400+% saturation. The sensor readings output with an accuracy of +/- 0.05 mg/L with a response time of 1 reading per second.

DFRobot Gravity series Analog Dissolved Oxygen Sensor Kit

This is an analog sensor from DFRobot for measuring dissolved oxygen. It is designed to be compatible with Arduino which leads to compatibility with a range of many microcontrollers. This sensor also works a galvanic probe that it comes with out of the box along with the main sensor module. The entire kit comes with everything needed to get started. The probe has a filling solution within it in order to work and can be replaced periodically. This period is about a month and would potentially be costly over time. The membrane cap that holds it together also needs to be replaced depending on the water clarity such as clean vs muddy water. The overall system can run on a range of 3.3 volts to 5 volts which is compatible with most microcontrollers. The analog output of the sensor is 0 to 3 volts and connects to a microcontroller’s ADC. The sensor can detect a range of 0 to 20 mg/L with a response time of up to 98% full response, within 90 seconds.

3.3.10 – Distance Sensor

For this project there are certain cases where we need to measure distance. One of which is monitoring the plant height and keeping track of growth. This is mostly because we are considering implementing a lift for the lighting system we implement. Such that the lighting height can be adjusted based on the height of the plants are certain stages in growth. This would mostly be done to help control intensity and control the amount of lights with the plants as well as giving the plants the proper room to grow. However, we have also considered other design options in which this sensor is not required for this case and we could potentially implement this design without the need for a distance sensor. So, it will come down to need, availability, and cost if we choose to implement this sensor. If cost isn’t an issue it may be worth implementing for the sake of accuracy. There is also the possibility that a distance sensor could be implemented elsewhere.

HC-SR04 Ultrasonic Module Distance Sensor

If we find that we need an ultrasonic sensor in our distance and money is available in the cost, the HC-SR04 is a popular option that is low cost. This sensor module emits an ultrasound at 40,000 Hz which then travels through the air. If it collides with an object or obstacle on its path it will then bounce back to the module. Now the module considers the speed of sound and the travel time it took for the emitted signal to bounce back and calculates the distance that object or obstacle is away from the sensor. The sensor requires a voltage of 5 volts and a current less than 2mA. The module has an induction angle of up to 15 degrees. The detection range is between 2cm and 250cm and has measurements with an accuracy up to 0.3cm. There is a good number of examples and documentation on the product. With its low cost and relatively simple design, if we feel that this is required or is a better alternative to some other designs we have considered, that this is a viable option.

3.3.11 – LCD Screen

One method of interface we are considering for this project is an LCD display incorporated into the physical main device. The purpose of this display would be to display the hydroponic systems current data. The display would show information about the system, its main focus being that you can get an understanding of how the system is doing at a quick glance. There is no planned functionality as far as controlling the system through the LCD or controlling anything directly through the system. The controlling of the system is currently planned to be purely implemented through the android application. The data and information will be displayed on the LCD at all times to provide real time information about the current state of the system. Exact graphics design and implementation will depend on the display and implementation. We could display all the information at once, but that could be cluttered. We could have a rotating data design which rotates through various screens which each show a specific type of data. Another option is to have a reduced amount of information displayed at all times. Here we would only display the most vital information that a user at a quick glance can see that a system is in good state or not. We would omit data that is more specific and less vital on a day to day basis.

When picking an LCD for this project there are some limitations and features, we need to consider. First thing is, what level of quality do we want this LCD to have. Can it be a relatively simple LCD module that is small and supports a row by column design. Such that each position in that matrix would correspond to a character these usually backlit and are character LCM LCD displays. Such that for example a 16x2 LCD display module would support 32 total characters, 16 per row. For simply displaying numbers and text to the user this may be enough, the difficulty would be in fitting the information on this screen in a clean and user-friendly way. On the other end we would have a full-fledged HD display with up to 1080p display. Having a 1080p screen would simply be overkill for the sake of this project so if we went down this route, we would probably max out at 720p. These HD displays would be physically larger as well as visually clearer and appealing to the user. They would also provide us with the flexibility to make a custom interface which would allow us to fit more information on the screen while keeping it clean and user friendly. A large component to factor in when picking between an LCD character LCM module and a full-fledged LCD HD display is how it would connect to the rest of our system. The simpler and less complex character LCM displays can be more directly connected to a microcontroller. We have experience with this from embedded systems and digital systems in which we worked with Texas instrument microcontrollers. We used these character displays for timers and for displaying basic data. These displays don't require too much graphical power and hence allow for almost direct connection to the microcontroller. They can be powered and run from the MCU and there are often libraries which will make controlling the individual character segments easier to do. These types of displays work by being composed of many zones. Each zone has several segments which can be turned on or off. In the code we use bit masking to turn on and off individual segments per zone. Once we do this for all the zones we can come to an actual message. If we do a larger more complex HD display the libraries and the coding becomes more complicated. With this, however, comes more flexibility in design. There are LCD displays which have graphical flexibility that go from 120px120p all the way to 1920x1080p. The higher the resolution and the larger the display the more power and energy is needed to run the display effectively. This can reduce the systems overall efficiency and performance but the increased functionality may be worth it. Some of these more advanced displays can be connected directly to the microcontroller as they are designed for systems such as the Arduino, some may require some, in between, modular interface for increased power and energy regulation. These displays cost a decent amount more then the character segment designs cost as much as 5 to 10 times as much depending on display quality. There are many small screen option LCDs out there but if the screen is so small making graphic visually appealing and even just straight recognizable gets difficult. So, we would want a larger display, which increases cost and power needed to run. Despite this, it may be worth investing. The Arduinos larger and complex displays require an external extension module, so we would have to consider with our custom PCB whether we could connect and run the LCD properly. If the libraries are there and we can make the LCD work with our PCB, we can just use an LCD display and work with the given libraries. If there are power issues or if we wish to make the LCD display and what we do with it more complex, it may be worth having an extra external system outside of our microcontroller and its PCB. One option is to have a separate Arduino dedicated to running the display and have the connect to our custom PCB for controlling it. This way we would offload running and handling the LCD from the microcontroller and let the microcontroller focus more on the sensor input and controlling other subsystems. Another option would be to use a single board computer like a raspberry pi. The raspberry pi is very powerful and can run a full OS such as Linux. It would be plenty to run anything graphical we want for the user interface and could lead to more advanced features. With a raspberry pie there is also the built-in benefit of wireless functionality being built in so there is the potential for connecting the display over the web and implementing more advanced features.

Overall, for the LCD we are looking at simple character array displays and proper LCDs. If we go with a smaller, lower res screen that can run directly off our PCB we will do so, if we find that we want greater functionality and more powerful and complex display we may dedicate a whole subsystem to the display. This would be done with an Arduino or even something as powerful as a raspberry pi. This decision will be done after parts are chosen for the microcontroller and could change as problems appear.

3.3.12 – Air stone and Air Pump

In the Dissolved oxygen sensor section, we discussed the importance of oxygen in the water. We also discussed that monitoring it with a sensor is expensive and may not be necessary for monitoring the overall water quality. Despite the cost for monitoring it, it is possible and cost effective to increase the oxygen levels in the water. This is done by breaking the water's surface tension, which creates more area for the water to interact with the air, which in turns allows for more oxygen to be absorbed into the water. In aquariums you often will see bubbles coming out of an air stone or even some little toy chest that opens and blows bubbles. These actually have a purpose. In aquariums it is used to increase the oxygen levels by breaking the surface tension of the water. As the bubbles rise up and hit the surface it breaks the surface tension, creating more area, and aiding in the overall oxygenation. This helps with fish as fish need oxygen to breathe through their gills. Practically this can also be achieved by having flowing water, waterfalls, and high amounts of circulation. Any sort of water movement that breaks the surface and the water from being stale aids in oxygenation. For the sake of this project it will be important to keep the water from getting stale to ensure proper oxygenation of the plants and to keep dissolved solids in the water from settling. If the water flow is high, there is a natural amount of splashing from potential waterfalls and the water is circulated well, then it is most likely safe to assume the system is in a good state. If there is an issue in keeping the water moving, we could implement an air stone. An air stone is a high porous rock that when connected to an air pump can diffuse air. When placed underwater an air stone will create bubbles of different sizes, which can depend on the quality of air stone that will float to the surface. The bubbles that reach the surface break the tension and create more surface area which provides more space for the water to interact with the air and absorb oxygen from our atmosphere. Underwater the bubbles rising will break up any still water and keep dissolved solids from settling and increase the overall circulation of the system. A feature like this won't be required for this project and will largely come down to how our overall water flow is working out in our end system. Despite this, adding an air stone is cost efficient and has almost no negative effects. As a result, it may be worth adding to the system anyways, just having it for safe measure. Any low-cost air pump would suffice, depending on water volume.

3.3.13 – Nutrient and pH Peristaltic Pumps

Whenever we get a sensor reading that indicates some sort of water quality imbalance, we potentially need to handle it. This will be useful when first setting up the system and when we need to monitor and maintain it once it’s up and running. There are certain things we have no control over such as the environment and the local water supplied to us for our system. Across the country, and even the world, our water sources can vary. This all can have an impact when the system is first set up. Water from a well will be different from city water. Water is often referred to as being hard or soft. Hard referring to a high amount of dissolved minerals and soft water being low amounts of minerals with sodium being the primary ion left. In Florida our water is generally considered to be hard. The majority of our water comes from the aquifer and can change from across Florida. South Florida such as Miami and west palm beach has harder water than that of Orlando or Tallahassee. As a result, when the hydroponic system is initially set up, we need to account for the water quality we have access to and adjust it for the designated plants the user wishes to grow. This means we need to be able to control the amount of nutrients in the water, monitor it, and potentially add or remove any. There are chemicals and filters that can be used for removing the extra amounts of dissolved solids, along with simply draining and putting fresh water, but for this section we will focus on how we can add nutrients and modify our water's pH.

Whenever our system finds that there is a lack of nutrients in the water, potentially by the lack of dissolved solids in the water, we need to be able to add it. In Hydroponics this can be done in a few different ways, but one of the more common is a liquid nutrient solution. Generally, this is something that can be bought online or at a store that is designed specially to provide nutrients to hydroponic systems. The goal here is to minimize the users need to manually add a specified dosage of nutrients when needed. So, we will use peristaltic pumps in order to automatically deliver the nutrients when the systems detect, through our sensors, that there is a lack of nutrients in the system. This will be automatically done and require the user to have to manually interact with the system less often. This means the user would only need to refill the nutrient solution storage whenever it runs out. We see the importance of having the proper nutrients in hydroponics when we discuss the TDS sensor.

When our system finds that the pH is out of range, for the specific plant that its growing, we also need to fix that somehow. In both hydroponics and aquarium systems this is generally done a couple ways. There are some materials that increase hardness such as shells and rocks while some dry wheat and such can help soften the water. The issue with these for a hydroponic system is that they are solid physical pieces of matter and can be more difficult for maintenance of the user. Another option would be two liquid solutions that are common in the field called pH UP and pH DOWN solutions. Their respective names say exactly what they do. pH UP will help increase the pH of the water in the system, pH down will decrease the pH of the water in the system. For this we will need two peristaltic pumps, one each. If we find that the pH is low, the pump for pH UP turns on and adds a solution to the water which will in turn increase the pH. When the pH is high a separate pump will turn on and add the pH DOWN solution to the water. As a result, this will help keep the systems pH levels within the defined range for the plants. This will save the user the manual need to measure the amount of solution and to annually add it whenever needed. Keeping the pH within in range is important as it discussed in the pH sensor section, is vital to the plant’s growth and development.

The peristaltic pumps that we will be using to implement these systems aren’t the same as any aquarium or pond filter water pump. Peristaltic pumps are a type of positive displacement pump. Every one of these types of pumps features a flexible hose or tub, which provides the path for moving the fluid. Peristaltic pumps are based on a pumping principle called peristalsis, which is based on alternating compression and relaxation of the hose or tube. By doing so the pump draws content towards the pump at one end of the tube and propels the product out the other end of the tube, away from the pump. There is a type of rotating shoe, roller, or disc that passes along the length of the hose or tub that creates a temporary seal between the suction and discharge ends of the pump. As the rotor turns this sealing pressure will move along the tubing, forcing the product away from the pump and into the discharge line. Here the pressure is released and the tubing recovers creating a vacuum, which draws the product out, releasing it to our destination, and sealing the tub preventing any backflow. As a result of this process more content from the inlet side of the pump will be brought up towards the pump, and the process continues. This creates a self-priming positive displacement process. The decision to use these kinds of pumps comes down to a few things. They are relatively low cost, have no metal contact with the liquid as a result of the polymer/elastic tubing which prevents contamination, and because of simple maintenance compared to other pumps. The reduced maintenance is a result of the lower complexity of the pumps parts with minimal values, seals and glands. This makes the cost of repair low while making troubleshooting easier when trying to find the source of any issues we run into.

3.3.14 – Wireless / WIFI Transmission

In order to have our microcontroller communicate and operate in such a way that we can control it through an android app, we need to connect to the internet. Microcontrollers do not natively support wireless data transmission and in order to have the system communicate with the android app we need to transmit sensor data and other system data to our database and webserver. As a result, we will need to connect the microcontroller to some separate embedded system module that handles the wireless network connection. The module's job will be to take data read from the sensors that the microcontroller receives and send them over the internet to our database. Wi-Fi modules are relatively low cost, proportionally to the capabilities they allow, and will be a key part to making this hydroponic system smart and worthwhile. Many brands make Wi-Fi modules such as Texas instruments and come with many varying features and implementations. Connecting the module to the microcontroller can vary based on design and implementation. Some may require external extra modules in order to safely connect and power it. There may be a need for an external power source rather than powering off the PCB and it may require the connections to the microcontroller to be stepped Up or Down. If needed, other tools and modules will be researched based on the Wi-Fi module we decide on.

As far as features go, we need the module to cover a minimal amount of work. All it is required to do is transmit the data, it won’t need to control any other part of the hydroponic system. It will be our database/web server communication device and that’s it. The Wi-Fi module needs to follow the 802.11 b/g/n communication standard such that it can properly communicate with other devices and routers. We also need to ensure the module we pick supports a large range such that it can properly connect to and transmit data to the router and as a result over the internet. We need to ensure the Wi-Fi module supports two-way communication such that we can read data from the sensors in our system and control the system remotely by sending data from the database to the system. The android app should be able to effectively transmit commands to the microcontroller if we chose to implement that. The exact features of the android app aren’t entirely written in stone and are open to flexibility. Potentially we make the system controllable locally and make the android app simply for monitoring. Despite this we want the features to be there so that our design can be flexible such that we can adapt the systems features as we go. Finally, we want the module to both be cost and power efficient. There are many different models of Wi-fi Modules out on the market by many manufacturers. The specifics on features can vary but most are based on 3 volt or 5-volt designs. Like discussed, some may require some signal stepping in order to connect to the microcontroller but this can be done through design or by external modules. We will be considering many different Wi-Fi modules online. One of which we are considering is the ESP8266 which is popular among other designs and supports many of the features we will need. We will be sure to consider other options in order to make sure we have the best option for our design.

Another Option – Single Board Computer

One way we could connect to the internet and communicate with our database with a single board computer. For this design we would use a raspberry pi. The raspberry pie would be connected to the microcontroller. The microcontroller would use the raspberry pi in order to connect to the internet and commence all communication. The benefit of a design like this is that the raspberry pi natively supports many wireless communication protocols that are already built in. So, we would be able to communicate over Wi-Fi using the raspberry pi as well as there would be support for Bluetooth in our design if needed. There is also the added benefit that, like we discussed in the LCD section, that we could also use it to run the LCD user interface. The main negative to this design would the overall cost and power usage increase over a generic Wi-Fi module

3.3.15 – System Power

The plan for powering this hydroponic system is through any standard home 120-volt wall socket. There are no current plans to make it run off self sufficient battery power. The overall power cost and component power draw will just be too much for any cost-effective battery system for this design. Along with the fact that generally, a hydroponic system isn’t meant to be moved around and is meant to stay in one place, hence being connected to a wall outlet at all times won’t be a large negative. We plan on implementing a small battery for the microcontroller and sensors such that any power surges and power inconsistencies don’t mess with the system. Primary a small battery connected to the microcontroller will help keep the power signal stable and fluctuate less. There are some components we will need in order to convert the 120-volt wall outlet power to be useful to our system, but specifics on the design will be discussed later on in the report. Research will be done on what exact components are needed, one thing we know we will need is an AC to DC converter in order to make the AC power from the outlet into DC to be used by the various components.

3.3.16 – Other Various Components

There components below are just some components that may potentially be needed in order to implement the various main components of the system such as our sensors and pumps. These aren't components we are defining as needed but are potentially helper components to help the system connect and work together. There may be more that aren't shown below that we will discover with further design/research and implementation.

Logic Level Shifter

A logic level shifter could be used for situations in which we need to connect components to improper matching voltages. Such as, when we need to step up a 3.3-volt connection to a 5-volt pin or a 5-volt connection needs to be stepped down to a 3-volt pin. This can help when there are many components that aren't directly compatible and we need to modify the signal in order for them to communicate.

Relays

Relay modules could be used if we find that there is a need to control the signals ON and OFF state. Relays can act as a switch and allow us to connect and disconnect signals. Such as we want to turn on and off a pump. The microcontroller could connect to the relay and tell it whether or not we want the signal on or off. This can help control power to such binary devices that have only two states.

3.4 – Android App Development

In order to develop an app for our hydroponics system, a good android development environment is necessary. This will determine the ease of making the app and determine the learning curve in order to develop an app. Therefore, we would like to work with a development software that most of our team are somewhat familiar with and can contribute to.
One thing to consider when developing an app is if we want to build a native app or a hybrid app. The main key differences are programming languages used to develop the app and the platforms that can run on. Native apps are generally tailored to either android or iOS platforms, but a few development environments have cross-platform

capabilities. They are also coded in specific languages like Java, which is a language that half of the team is already familiar with, Kotlin and Swift. Hybrid apps are cross platform that can have a single codebase for both android and iOS. That's possible since it is developed using JavaScript, HTML and CSS. That said, its more like a mini version on a web application that emulates mobile software in usage.

3.4.1 – Hybrid App Development

Hybrid apps are very versatile in terms of development. Since it is most commonly built using JavaScript, HTML and CSS, it is regarded as a website that is put into containers. That said, APIs and frameworks are needed in order for the app development to function. Other third-party packages can also be downloaded to further develop the application. A huge advantage to using hybrid development is that a single code base is used for platforms like android and iOS. This minimizes development time if we wanted to have this app for both platforms and more. Hybrid apps would also have no integration issues when integrating with other apps, which could be useful when expanding the app's capabilities in the future. Offline support is also possible in hybrid apps. This means that with poor or no connectivity to Wi-Fi, there can be function within the app that will still work. On the other hand, constant internet connection is required for certain plugins to work, which can affect device features. Another feature of hybrid apps is that they are scalable with app versions. Therefore, it would be much easier to release updated versions of the app should we make improvements in the future. That said, it would also be easier to maintain and test the app before release.

Visual Studio - Xamarin

Xamarin is developed by Microsoft and is able to develop native apps for both platforms. Since it's developed by Microsoft, apps are created with C# and .NET. Since its part of the .NET framework, a number of features can be used like asynchronous programming and lambdas. It also offers cloud services to test apps, web services and some other 3rd party integrations. Since this is developed with C#, it can more closely emulate the performance of native applications like Java for Android. It's also worth mentioning that Xamarin is capable of full hardware support like native applications. This includes a camera and microphone for instance. Like native apps, it doesn't need specific plugins or APIs to achieve this. With the continuous support over the software, more improvements are made to give a more native feel. Another component of this development tool is the visual studio app center. This allows information on the app's performance, crash histories, and analysis on affected devices. It also can run automated tests on the UI that can detect problems at runtime.

A unique trait of Xamarin is the Xamarin.Forms. This is a simplified form of Xamarin. Android and Xamarin.iOS counterparts. That said, simple versions of apps that don't require complex animations or features can be developed with this. A good thing about it is that it compiles UI components at runtime for both iOS and Android. However, Xamarin.Forms runs a little slower than Xamarin.Android and Xamarin.iOS because of an extra abstraction layer. Despite this, it is great for fast development and would be easy to maintain for both platforms.

React Native

React Native was developed by Facebook and is mostly known for its performance to most hybrid development environments. One big appeal of React Native is the code reusability across both platforms and all the available components that can be downloaded for use. Since the community is so large, many pre-developed components are made for other developers to use. In turn, this speeds up the performance as code is written in advance. Another well-known feature is live and hot reloading. Live reloading allows the developer to add files to the application and automatically reads the app from the beginning. Hot reloading allows the developer to add files and view code changes without app recompilation. This cuts down time in developing the application.

Framework 7

Framework 7 is an open source HTML framework that uses traditional web technologies to develop apps. It was originally designed for the iOS platform, but has since expanded support to Android. It’s a simple framework that doesn’t depend on any single third-party library. It also has its own DOM library that contributes to its speed and simplicity. This framework also has a lot of the famous features to give the native feel. For example, you can easily add a swipe back feature and a pull to refresh, which are popular navigation controls in most smartphones. Framework 7 has also partnered with Vi to implement ads in the application to generate ad revenue in a simple way. Therefore, we have the option to place advertisements in our application with ease.

Ionic

Ionic is developed with standard web technologies like its hybrid peers. It’s a pretty new technology and was first developed with Angular from google. That said, it utilizes WebView, which is the devices browser instance. It also works in conjunction with Apache Cordova to convert the web technologies to mobile applications. Because of the WebView, most device features like the camera are blocked for use unless Cordova Bridge plugins are used to gain access to the device’s operating system. There are plenty of UI components available to use in its UI components library, which give fast prototyping of apps. Another nice component is the testing convenience. The app can be tested right from the devices browser and doesn’t require a testing device. One drawback is that it can be difficult to maintain with the plethora of plugins that are available and its performance isn’t the best considering the competition.

Titanium Appcelerator

Appcelarator is entirely JavaScript based and is capable of building cross-platform native apps. This is a great SDK for developers who are familiar with JavaScript and would make JavaScript technologies, like JSON formats for data transfer, easy to use. A big appeal is the rapid prototyping that it has to offer with simple syntax in coding. It also offers some native UI components and device functionality like GPS, camera and file system. Additional UI components include native navigation bars, menus and dialog boxes. Another impressive feature is Hyperloop, Hyperloop allows the user to access native code within JavaScript and use 3rd party libraries. Couple this with Angular and Vue.js integration, more possibilities open up in developing mobile apps. There is also the Appcelarator Cloud Services (ACS) integration that can expand the apps capabilities by being able to notify the user of status updates, push notifications and photo uploads. The addition of Google's V8 JavaScript engine only makes the performance of the app better and gives off a more native feel.

3.4.2 – Native App Development

Native apps are very specific to a platform. For example, Java and Kotlin are used to develop Android applications. While Objective-C and Swift are used to develop iOS applications. This usually results in developers developing both technologies to support both markets, which is more time consuming. However, the whole team has android so developing for iOS isn't a huge issue. Another huge advantage native apps have is the increased performance on phones. Since it's platform specific, the SDK tools will run faster than traditional web technologies. This will improve the customers experience as UI will be more seamless to the user and make the customer experience more enjoyable. The platform specificity also allows full use of device features like the camera, GPS and microphone. These could be used for future features that could be integrated with the hydroponics system. There will also be less bugs to work out since the programming language is made for the platform. Added data security is also a great advantage with native apps. This is more important to financial companies, but can be a great feature to have for our hydroponics system. UI effects and other animations are also better in native apps since they are supplied in SDKs and not from potentially unreliable plugins. In terms of database connectivity, native apps are able to connect to different databases without extra plugins or tools like hybrid apps. This would simplify the database connectivity portion of the project and could cut down on development time.

Android Studio

Android Studio is a common development environment that uses Java or Kotlin to make Android applications. A perk of this software is that it can apply changes to the app without restarting the app. It can also emulate the application so you don't have to load it into the device every time. This helps with code edits and debugging the app in a

faster way. There are also testing tools with JUnit4 and functional UI testing frameworks like Espresso Test Recorder. Similarly, there is a layout editor that can visualize changes in the layout of each component in the app and assist in finding the right place to put said components.
Another nice feature is the build automations and customizable build configurations. Version control support is also supported with connections to GitHub and Subversion. This is a great element in Android Studio for team development and would speed up the development process. There is also Firebase and Cloud integration. This gives us analytics and detailed instructions that can be used to improve our app. Furthermore, it can integrate services like Google Cloud Endpoints to maintain API used in the app development.

RAD Studio

RAD Studio is a native environment that uses a single code base to develop ads for android and iOS. While it may have cross-platform capabilities, it compiles natively to the platform you want. It is also hard to reverse engineer the code for the platform of your choosing. You can code with either C++ or Delphi to design the app. While C++ Is a familiar programming language for most, Delphi would require some learning to utilize its features. There are features with the FireMonkey framework that allows for high end UI design. With many flexible libraries, there is no shortage of possible UI. Another great feature is the ARC for mobile that manages memory at compile time and optimizes it. This leads to better running code that would surpass the performance given by Java and C# in other environments.

3.5 – PCB Design:

Printed Circuit Board (PCB) manufacturing is the process in which a board is fabricated that is used to support and connect various electronic/electrical components to one another. Various components such as inductors, capacitors, resistors, and other devices are soldered on the board. First, the PCB board needs to be designed using a design software before being sent to a manufacturer. We will discuss some design software and the advantages and disadvantages later. Once the printed circuit board is laid out in the software, it is sent to a PCB manufacturing company to bring the circuit schematic to life. Some of the PCB manufacturing companies will be discussed in this section.

3.5.1 – PCB Design Software:

There are various kinds of PCB Design software and many PCB vendors offer up their own software to use for free to entice the user to use their service. Some of the PCB software out there is free while others are premium which you must pay for. When selecting a design software, it is best to use some of the latest tools with available support and community. We will discuss a few types of popular design software:

Autodesk Eagle

Eagle is arguably one of the most well-known schematic and PCB design software. It contains a schematic editor, for designing circuit diagrams and a PCB layout editor for designing PCBs. It provides component placement, PCB routing, a comprehensive library content, and much more. There is a free version of this software available which includes 2 schematic sheets, 2 signal layers, and an 80cm^2 board area. Eagle is the industry standard because it offers a full suite of options and has direct connections to the company's electrical component database. This allows designers to easily choose components which are in stock and available for order and generates a bill of materials automatically.

KiCAD

It is a cross-platform and open source electronic design automation suite. It includes a schematic editor for creating and editing schematic designs, a PCB Editor for making a professional PCB layout with up to 32 copper layers, and a 3D viewer which can be used to inspect the design in a 3D form. Unlike Eagle, this software is completely free. It is available for Windows, Mac, and Linux. It has an extensive community to ask questions and if help is needed.

DesignSpark PCB

It is an easy to learn environment with a schematic capture and PCB layout tool. It is totally free to use and comes with an excellent schematic capture, PCB editor for designing unlimited number of PCB layers, a part and library creator, 3D views, and many more features. It has an extensive library and a community willing to help.

3.5.2 – PCB Vendors:

After the PCB board layout is created with the help of a PCB design software, the components are selected and sent to a PCB manufacturing company. The reason for using a manufacturing company is because the process of manufacturing complicated printed circuits onto substrate material is quite difficult and thus a PCB manufacturing company with the right tools can perform it with ease. There are numerous PCB vendors to choose from for manufacturing PCB for this project. The main requirement for choosing a vendor is the cost and the time it takes for the job. Other factors like the PCB quality and ability to meet our requirements are also important.

One of the main constraints for our project is the financial budget. Since we are paying for the project ourselves, it is important to use the cheapest vendor but also keeping quality in mind. Also, it is also important to use the components which are necessary and as inexpensive as possible. The other constraint to keep in mind is that our project needs to be able to grow a plant which can take up to weeks or months, which means getting the PCB in a timely manner is important. We would need to get the PCB done in the first month of Fall semester so that way we can have it as soon as possible. It is important to keep in mind that creating a PCB is one of the important requirements for this senior design project. Some PCB vendors that are researched and taken into consideration are discussed:

4PCB

This is North America's third largest printed circuit board manufacturer. It offers free design software PCB Artist design software to be used. It offers a low cost of a 2-layer PCB board for $33. There is no minimum amount of boards required to order. It is very useful for engineering students who need to build their project prototype inexpensively. The turn time is maximum of 5 days for a standard board.

Sunstone Circuits

This company has over 45 years' experience in PCB manufacturing and operated out of the USA. The company also provides a free design software PCB123. It can create anything from simple PCB boards to highly complex RF/Microwave boards. The 2-layer PCB board starts around $32 and usually takes from 1 day to 3 weeks.

Imagineering inc.

The production facilities are in Illinois. It specializes in rigid and flex double-sided and multilayer PCBs. The turn time is 5-6 days. The cost of a 2-layer board is around $25-$30.

Express PCB

This is another manufacturer that provides PCB fabrication service. They charge $51.00 plus shipping for three mini sized PCBs. This vendor is expensive and has a slow turn time.

PCBWay

This company is based in China. It has no minimum requirement. The company is also able to take returns and offer refunds if the products don't work. It offers a price of $30 with shipping and the turn time is around 3-5 days

3.6 – Databases for Mobile App Development

SQLite

SQLite is a very popular database that is used by many well-known companies like Apple, Adobe, Facebook, Google and Microsoft. Since it’s a popular database, there are plenty of features that are fully tested and are in constant development for reliability. It’s a small database that is compact and very reliable. It's also a relational database, so tables are created to manage the data in the database. It is mostly used for web browsers and front-end mobile apps.

Some features it includes is that it’s serverless. This means that the database has direct access to storage files and doesn’t need another server to accomplish the same task. SQLite also doesn’t require a server to operate and is also contained in one library that is embedded in the application. This eliminates the setup time of a server and simplifies the process. Creating an instance of a database is easy with little margin for error. It also can also be used for many platforms like Android, iOS and Windows. The database is also fully ACID-compliant, which stands for Atomic, Consistent, Isolated and Durable. This comes in handy for safe transactions for multiple users on multiple threads. Furthermore, it has additional commands via extensions that are not found in SQL. REPLACE, ATTACH, ON CONFLICT commands are a few clauses that are not found in the SQL database to allow for more customization within the database.

ORMLite

ORMLite is a simple database that simplifies complex SQL commands. ORM stands for Object Relational Mapping and works with Java objects in the database. There is a flexible query builder and provides abstract database object classes. ORMLite supports multiple databases that are cross-platform like MySQL, SQLite, and other databases for expandability. This allows for more features to be used that can handle more unique databases.

While ORMLite center around Java, there are a lightweight C# extension that can be used with the .NET framework. Therefore, it can have more possibilities with interfaces for the database. ORMLite also has support for indexes and text blobs, which is useful to organize data. It can also auto generate SQL code to create tables and input data. It’s important to note that there is no Java database connector for Android app development. Therefore, it makes direct calls to Android database APIs to use SQLite databases. However, these are still native calls to the Android OS. Since ORMLite is a larger program, it is also slower than other databases like SQLite.

Berkeley DB

Berkeley DB is a high-performance database that also provides reliable storage of data. This is due to its developments as it is written in the C language, which is one of the fastest languages in the world. There are also other versions of Berkeley DB that are written in different languages like Berkeley DB Java and Berkeley DB XML, which is written in C++. It has API bindings for many programming languages like C#, Java, PHP, Python, Perl and many more. Therefore, it has plenty of APIs for all those languages for Android and iOS mobile development.

One feature of Berkeley DB is the data management. It has configurable locking for concurrency and a library for data management in embedded systems. It can store data in in-memory, disk or both. There is also strong encryption for the transactions to keep data safe. In order to keep the safe, useful transactional features are implemented. That said, it has deadlock detection, plenty of recovery options and log file archival. Berkeley DB is also ACID-compliant to keep transactions safe.

ObjectBox DB

ObjectBox DB is an object-oriented database that specializes in mobile database management. It is easy to program with as it revolves around writing less code to have a database up. It does this by transferring enterprise features into an XML interface that is easy to work with. One thing to note is that this database only has support for Android and Linux, but iOS development is up and coming. That said, it also supports JavaRx and Kotlin programming languages for Android development.

A strong point is that ObjectBox DB is that it doesn’t use SQLite and a medium and there is no ORM. It is also faster than SQLite, which is regarded as one the faster databases for mobile development, and a lot of other embedded databases. Since there is no ORM, it builds objects from scratch and there are no rows or columns to work with. This makes It easier to develop a database and would save time in debugging. ACID properties and MVCC, which stands for Multiversion Concurrency Control, is also supported for safe transactions and parallel threads. There is also instant unit testing that can be run on an ObjectBox database in seconds on the desktop.

Realm DB

Realm DB is a database that is compatible with numbers platforms like Xamarin and React Native. It is serverless like SQLite and provides faster performance. Data can be stored on disk and in in-memory. Realm DB can also be used for both Android and iOS development. Since it isn't an ORM, it doesn't rely on SQLite to function and is based on an object store. To accomplish this, a new database engine is used for transactions.

It is easy to create and store data in Realm DB as it uses NSObject classes. This means that @properties can be used to define data models. That said, model objects can easily be made with subclasses of RLMObjects. Another impressive feat is the speed of the queries that this database can handle. As mentioned before, it is faster than SQLite in app data storage and can handle complex queries with ease. There is also the addition of the Realm browser, which gives users the ability to administer and explore databases. Encryption is also a useful feature as it can secure data with transparent encryption and decryption. Concurrency support is also there with MVCC architecture where views are automatically updated when a transaction is fully committed from any thread.

Couchbase Lite

Couchbase Lite is a JSON centered embedded database that is designed for mobile app development. The data is stored in JSON documents and can have multiple attachments. It is able to use MapReduce to query persistent indices, or views, and can also manage them. Couchbase is able to communicate with the cloud whenever a signal is available. It also runs locally on the mobile device in a variety of languages of support. Ranging from Swift to Java to C#, finding language compatibility with popular programming languages is an easy task.

This is a lightweight database that under 1MB is space. There are some security features like 256-bit AES encryption. Couchbase Lite also allows for more options as it contains REST and native APIs for data management. There is also support for Sync Gateway, which allows for server synchronization to secure data that is being transferred between clients and servers. This will be beneficial for when we need to secure the users data if this is a desirable feature future users want implemented.

LevelDB

LevelDB is an open source database that is developed by Jeffrey Dean and Sanjay Ghemawat of Google. That said, this database is used by Google themselves in their Google Chrome software product and others. It is written in C++ and can be used for both iOS and Android platforms.

LevelDB does not support SQL Queries or indices. It doesn't provide a server or a command-line interface either. It is best used a library that has a high amount of read and write operations. That said, LevelDB is also capable of batch writes with put and delete batch operations. This makes it a great database for writing in large data. Put and delete operations are also available as basic operations that make it easier to manage transactions with the database. There is the Snappy Compression Library that offers automatic data compression that would help with memory management.

MongoDB

MongoDB is cross-platform and a NoSQL database utilizes JSON documents for transactions. It uses familiar JavaScript technologies that are already a little familiar and that makes it a compelling choice to use. It is also used by a lot of companies and users. This also means that there is a large user base, so the technology is heavily updated and secure. There are also different editions available on MongoDB that give different options on functionalities with different cloud platforms and features.

Some useful features include the Ad-hoc queries which support additional searches like range and regular-expression searches. It also has indexing to outline the documents for readability. There are also load balancing countermeasures to ensure safety for if the database were to fail. This includes duplicating the data as copies to have the most up to date information and keep the system functioning. Aggregation is also a neat component as it can perform this in multiple ways including aggregation pipeline, the map-reduce function and single-purpose aggregation. Map-reduce and Aggregation Pipeline are 2 methods that are used for batch processing. However, Aggregation Pipeline has better performance than its counterpart.

4.0 – Engineering Standards

The success of any engineering project is in part related to the projects reliability, sustainability, and overall effectiveness. Any product put into the market meets these expectations by following engineering standards put in place by relevant standardization bodies. Engineering standards are detailed documents that provide requirements and specifications to help guide the design and construction of engineering projects. These requirements help ensure a viable product that is effective and safe for the consumer/user. Users should always check to see if the products they buy/use adhere to the latest standards if possible.

Our hydroponic system will contain standards that are relevant to the systems wireless capabilities, software, battery, sensors, and much more. The relevant standards of each of these areas is expanded upon in the subsequent sections and can be found along with further explanation of what they are and how they relate to the project. For the sake of user readability, only the most important standards related to the project are discussed below, as there are thousands of standards in the world and we would like to focus on the most important to how the user will be affected.

4.1 – Communication Standards

One of the main goals of the hydroponic system is to be able to monitor the analytics of the system that are related to plant growth and customize what kind of plant the user would want to grow through an Android app that would interface with the hydroponic system and have the system automatically create an environment optimal for the selected plant type. In order for our microcontroller and app to communicate, we either need to choose a microcontroller that comes with Wi-Fi capabilities on board and ready to use or find an external Wi-Fi module that can communicate with an external microcontroller through embedded communication.

4.1.1 – 802.11 Wireless

The IEEE 802 standard covers local area networks, particularly involving the physical and data link networks. IEEE 802.11 is the standard most relevant to our hydroponics system, as the microcontroller will communicate and share data to the Android Application through the use of a Wi-Fi connection (with either a 2.4 or 5 GHz connection).

4.1.2 – JTAG Programming

JTAG is a hardware interface that provides a way for a user's computer to communicate directly with the chips on a board. Developed in the mid 1980's, the Joint Test Access Group wished to make it easier to test PCB's and has since become a mainstay in the debugging, testing, and development of pretty much all embedded devices. JTAGs are common on microcontrollers such as the MSP430 and are used to reprogram the board. The IEEE1149.1-1990 standard was introduced in 1990 to provide specifications that JTAGs should adhere to, such as a standard interface for instructions and test data. The latest version is IEEE1149.1-2013, which similar to IEEE1149.1-1990, defines four pins (Test Clock, Test Mode Select, Test Data-In, Test Data-out, with an optional fifth Test Reset), which communicate with systems that follow the boundary scan protocol. With boundary scan, testing of interconnects is possible due to how users are able to read and write single bits on individual pins of individual chips on a board. A JTAG is not a connector, but the presence of the necessary signals mentioned above that make it possible to easily debug and test microcontrollers that use this standard.

4.2 – Software Standards

The software in the microcontroller portion of the hydroponic system project will need to be able to reliably collect, store, and transmit data to a database that will be able to move that data to the corresponding Android App for easy reading by the user. The microcontroller will likely be coded in the C programming Language with a basic IDE like CodeBlocks. The Android App will likely be done in Android Studio utilizing the Java programming language.

4.2.1 – C Programming Language

The C programming language first appeared in 1972, developed by Dennis Ritchie and Bell Labs, designed to provide low-level access to memory and present language that is easily understood by humans but converts efficiently into machine code. The most up to date standard of C is defined as ISO/IEC 9899:2018, released in the year 2018 and also often referred to as just C18. The microcontroller will be coded with the C programming language, providing a capable low level programming language that all members of the project are familiar with and that has an extensive community online that is able to help with troubleshooting problems.

4.2.2 – Android Studio Design & Quality Guidelines (Java)

The Android App platform does not hold developers to strict standards, but has laid out guidelines to try to ensure that applications adhere to a standard for visual and navigational patterns. The Material Design is a visual language that is inspired by the physical world and it’s textures and is meant to feel welcoming and accessible for users. Furthermore, quality guidelines are available to test Android apps and help to make sure to ensure an excellent user experience across Wear OS, TV, and Auto apps. Should we decide to make this product commercially available these are guidelines we should consider to create a satisfying user experience.

4.2.3 – Java Programming Language

The Java programming language first appeared in 1995, designed by James Gosling and developed by Sun Microsystems (now Oracle). Java is one of the most popular programming languages in use according to Github and has a syntax similar to that of C or C++. Java is highly portable, due to how Java code can run all on platforms that support Java without the need to recompile. Unlike the C programming language mentioned earlier, Java does not adhere to a distinct industry standard. In 1997, Sun Microsystems approached ISO/IEC standards body to formalize Java, but ultimately withdrew and Java remains a *de facto* standard, controlled through the Java Community Process. A de facto standard has achieved a dominant position through public and community acceptance.

4.3 – Sensor Standards

Our hydroponic system will use sensors to assess the status of the current plants conditions and ensure that they reflect the ideal conditions of the user-selected plant.

Our system will utilize pH sensors, proximity sensors, water level sensors, and more. Below are standards that are relevant to the sensors used in our project.

4.3.1 – IEEE 2700

IEEE 27000 provides a common framework for measuring sensor performance by defining specification terminology, units, conditions, and limits. The latest version of this standard is IEEE 2700-2017, which was first published in 2018 and supersedes the former 2700-2014. IEEE 2700 is intended for sensors that use a digital I/O interface to communicate their readings. The relevant sensors that fall under IEEE 2700 for our hydroponics system are humidity sensors, temperature sensors, light sensors, and proximity sensors. IEEE 2700 was created with an intent on easing system integration and time to market. This is essential to our hydroponics system, as we will be working with a limited time frame and will have less than one semester to integrate all our parts together into one working, cohesive product.

4.4 – Environmental Protection Standards

Water and electricity can be dangerous when mixed together and they both are used in our hydroponics system. The safety of the user is the most important aspect of product creation because not only does it need to be reliable and dependent, but also should the product harm the user in any way, we would be open to numerous legal actions. When cleaning the hydroponics system (which should be done periodically as debris may affect plant growth), users should disconnect the system from the power source and then reapply power once they are finished. To prevent electrical wires from creating hazardous environments, all wire routes will be placed to avoid contact with water if possible and have proper enclosure that will prevent damage to the wire.

A relay system will be considered in the project design in order to protect the user by preventing power surges. Relay systems are common in commercial power supplies so it will most likely be built into the power system we choose. Power surges occur when a high amount voltage occurs in the wiring and causes a surge in the transient voltage/current/power. This can cause overheating or short circuiting in the system, so relays are meant to detect these surges and pretty much create an open in the system so that electricity stops flowing and the components of the system are kept protected. Below specific standards are detailed that help us determine how to best protect our system from any complications caused by the combination of water and electricity in our hydroponic system.

4.4.1 – IEC 60529

The International Electrotechnical Commission (IEC) is an international standards organization with its headquarters in Geneva, Switzerland that publishes international standards for all electrical, electronic, and related technologies (“electrotechnology”).

Should the microcontroller or Printed Circuit Board in our design be located in close proximity to any water source it is important we create an enclosure that will adequately protect against any damage. Many consumers are familiar with IP water protection ratings due to the prevalent rise in “waterproof” smartphones that have come to market in the last few years. If we made an enclosure it would not need extreme protection but be able to protect against splashes of water and be able to protect against a small spill.

In 1998 the International Electrotechnical Commission created the original definition for the standard called IEC 60529, with contributions from the American National Standards Institute and the National Electrical Manufacturers Association. IEC 60529 establishes an enclosure grading system that produces an IP rating based on the product's ability to keep the environment from interfering with the operations of a product. IP ratings start with IP and are followed by two letters and an optional letter, each pertaining to a certain aspect of protection. IEC 60529 contains three tables (found below), that correspond to each of these aspects of protection with table 4-1 containing a rating on an enclosures dust protection, table 4-2 a rating on the enclosures accidental contact protection, and table 4-3 being relevant to the enclosures water protection. So with a product's IP rating users are able to learn about an enclosure's dust, accidental contact, and water protection.

First Characteristic Numeral	Brief Description	Definition
0	Non Protected	
1	Protected against access to hazardous parts with the back of a hand.	The Access probe, sphere of 50 mm ø, shall have adequate clearance from hazardous parts.
2	Protected against access to hazardous parts with a finger.	The jointed test finger of 12mm ø, 80 mm length, shall have adequate clearance from hazardous parts.
3	Protected against access to hazardous parts with a tool.	The access probe of 2.5 mm ø shall not penetrate.
4	Protected against access to hazardous parts with a wire.	The access probe of 1.0 mm ø shall not penetrate.
5	Protected against access to hazardous parts with a wire.	The access probe of 1.0 mm ø shall not penetrate.
6	Protected against access to hazardous parts with a wire.	The access probe of 1.0 mm ø shall not penetrate.

Table 4-1: Degrees of Protection against Hazardous Parts

Our final design will likely have protection to the “2” characteristic numeral in the *Degrees of Protection against Hazardous Parts* table 4-1, as by the end of senior design we will likely only have a prototype functional design and not be releasing a product to a consumer market. We will want to keep our system easy to tinker with and make any necessary adjustments, as we will have a strict schedule to do system integration.

First Characteristic Numeral	Brief Description	Definition
0	Non Protected	
1	Protected against solid foreign objects of 50 mm ø and greater.	The object probe, sphere of 50 mm ø, shall not fully penetrate.
2	Protected against solid foreign objects of 12.5 mm ø and greater.	The object probe, sphere of 12.5 mm ø, shall not fully penetrate.
3	Protected against solid foreign objects of 2.5 mm ø and greater.	The object probe of 2.5 mm ø shall not penetrate at all.
4	Protected against solid foreign objects of 1.0 mm ø and greater.	The object probe of 1.0 mm ø shall not penetrate at all.
5	Dust-protected	Ingress of dust is not totally prevented, but dust shall not penetrate in a quantity to interfere with satisfactory operation of the apparatus or to impair safety.
6	Dust-tight	No ingress of dust

Table 4-2: Degrees of Protection against Solid Foreign Objects

The design of our final hydroponics system will not worry about dust protection because it should not be left in any areas with high amounts of dust for any prolonged amount of time and will not be a consumer focused design but more of a prototype. Should we move to a consumer ready device then the enclosure should have some sort of dust protection. Because of this achieving a characteristic numeral of “1” for dust protection would be satisfactory for our hydroponic system.

First Characteristic Numeral	Brief Description	Definition
0	Non Protected	
1	Protected against vertically falling water drops	Vertically falling drops shall have no harmful effects.
2	Protected against vertically falling water drops when the enclosure tilted up to 15°.	Vertically falling drops shall have no harmful effects when the enclosure is tilted at any angle up to 15° on either side of the vertical.
3	Protected against spraying water.	Water sprayed at any angle up to 60° on either side of the vertical shall have no harmful effects.
4	Protected against splashing water	Water splashed against the enclosure from any direction shall have no harmful effects.
5	Protected against water jets	Water projected in jets against the enclosure from any direction shall have no harmful effects.
6	Protected against powerful water jets	Water projected in powerful jets against the enclosure from any direction shall have no harmful effects.
7	Protected against the effects of temporary immersion in water.	Ingress of water in quantities causing harmful effects shall not be possible when the enclosure is temporarily immersed in water under standardized conditions of pressure and time.
8	Protected against the effects of continuous immersion in water.	Ingress of water in quantities causing harmful effects shall not be possible when the enclosure is continuously immersed in water under conditions which shall be agreed between manufacturer and user but which are more severe than for numeral 7.

Table 4-3: Degrees of Protection against Solid Foreign Objects

In the *Degrees of Protection against Solid Foreign Objects* Table 4-3, as mentioned earlier we are striving for a design that is able to protect against accidental splashes and similar levels of water so we should have an enclosure that is rated to the fourth level of the table. Based on the analysis of the three tables pertaining to standard IEC 60529, our hydroponic system would aim for an ideal IP rating of IP214.

4.5 – Printed Circuit Board Standards

The IPC was founded in 1957 as the Institute of Printed Circuits and was aimed at standardizing the assembly and production requirements of electronic equipment and assemblies. IPC has since changed its name to the Association Connecting Electronics Industries as electronic boards became more complex, with packages and assemblies growing beyond bare boards and their standards being the industry standard among electronic board makers. Accredited by the American National Standards Institute (ANSI), the IPC creates standards that govern every step related to the design, inspection, testing, and documentation of printed circuit boards (PCBs).

4.5.1 – IPC-A-610 Acceptability of Electronic Assemblies

IPC-A-610 is the most widely used standard that is published by the IPC and is used around the world, with a reputation as the source for end product acceptance criteria for printed wiring assemblies. Industry professionals responsible for the quality and reliability of electronic assemblies take training that ensure they understand the topics covered in the IPC-A-610 standard which includes, but is not limited to: hardware installation, soldering criteria, jumper wire assembly requirements, surface mounting criteria for chip components, and more. Any supplier we choose to order our Printed Circuit Board from should have professionals that are well-versed in this standard to ensure we receive a board that is reliable and matches the design that we sent.

4.5.2 – IPC-2615 Printed Board Dimensions and Tolerances

Released in 2000 and meant to replace one of IPC’s earliest standards IPC-D-300G, IPC-2615 is meant to give PWB designers knowledge of how to accurately space components to avoid interference and errors. The document includes: Fundamental dimensioning and tolerancing rules, Positional tolerance, Profiles/controls and tolerancing, and more. This standard will prove very valuable to our design team as none of us has an extensive background in Printed Circuit Board design outside of some minor coursework and club projects. Containing over 100 images and illustrations, the standard is meant to be easy to follow along with and good at teaching how to build a proper functioning printed circuit board.

4.5.3 – IPC-2221 Generic Standard of Printed Board Design

Superseding standard IPC-D-275, IPC-2221 Generic Standard of Printed Board Design establishes general requirements for the design of organic printed boards and other forms of component mounting or interconnecting structures. The requirements contained in IPC-2221 are meant to establish design principles with recommendations that should be used along with the set of standards that fall under the IPC-2220 PWB Design Document set, identified in the table below.

Standard Code	Brief Description
IPC-2222	Rigid organic printed structure design
IPC-2223	Flexible printed board structure design
IPC-2224	Organic, PC card format, printed board structure design
IPC-2225	Organic, MCM-L, printed board structure design
IPC-2226	High Density Interconnect (HDI) structure design
IPC-2227	Organic board design using discrete wiring

Table 4: IPC-2220 PWB Design Document Set

Many more standards are used in conjecture with the examples listed in the table above to produce designs that are intended to mount and attach passive and active components. These include through-hole components, surface mount components, fine pitch components, and more. All dimensions and tolerances in IPC-2221 are expressed in metric (SI) units in order to be translatable across the world.

IPC-2221 provides directions for classifying board types based on usage, increases in sophistication, performance requirements, and the frequency of testing and inspections. Printed Circuit Boards are split into three categories labeled Class 1, Class 2, and Class 3 board types.

5.0 – Realistic Design Constraints

All University of Central Florida ECE Senior Design projects are designed to be completed over two semesters. Senior Design I (EEL 4914) is completed in one semester and details the project planning and parts selection. Senior Design II (EEL 4915) is done in a future semester and involves system integration, testing, and final preparations. Our group has elected to do Senior Design I in the Spring 2020 semester and Senior Design II in the Fall 2020 semester. Ideally, this would mean that anything not able to be completely done in Senior Design I such as procurement of necessary parts to be completed in the summer between.

5.1 – Economic and Time Constraints

Engineers are expected to be the innovators of society, constantly pushing the envelope on what is possible and what to expect in the future. Although this is true, what engineers create is often inhibited by time and economic constraints that are imposed by tight schedules and the reality of price cost. In our situation we have clear constraints such as being college students and having a strict allotted time period to complete our hydroponic system. Besides the obvious constraints, this section will delve into extra obstacles we have also encountered that will affect the way we discuss as a team, our time allotted for ordering parts/meeting/advising, and meet with advisors.

With the unexpected rise of the Coronavirus (COVID-19) in the United States in March 2020, many schools and universities around the country have moved to online instruction, including the University of Central Florida among them. This will have a profound effect on the anticipated schedule and planning involved with the hydroponics system and how we as a group approach our deadlines. Much of our group discussions, such as our weekly meetings, that would be face to face have been replaced with group meetings on work clients such as Discord and Slack. Any discussions scheduled with advisors have been moved to a video chatting platform called Zoom that allows us to communicate “face to face” over the web.

Beyond the challenge of not being able to meet face to face, the pandemic has had a profound toll on not only the US economy, but also the world economy as a whole. With fewer workers available to work, many part order requests are expected to arrive later than normally anticipated. Along with this, shortages are possible as parts may be out of stock for prolonged periods of time. Should part procurement take longer than expected, we luckily have the extra time in the summer semester as mentioned before to continue ordering parts.

As a group of college students when choosing our hydroponic system project we were already on a short budget, but as the virus continues to affect the US economy some cuts may need to be made if deemed too pricey. We were already planning on using practical components that have been thoroughly tested and have a strong following, and avoiding cutting-edge tech that while it may be useful, has little help available for currently on the internet and holds a higher price tag. Upon researching the price of sensors involved in the system, we found many digital options are often pricier than their analog counterparts. Should we follow a strict budget, analog components may best fit with our price range. Also, we may use a variety of suppliers for the sensors, as sticking with one company does not always provide the best price.

5.2 – Environmental Constraints

Our hydroponic system should have the ability to modify the local environment experienced by the plants being grown. However, these changes should be self-contained and not have an adverse effect on the environment that surrounds the hydroponic system. The system is meant to be used inside of small apartment spaces where conventional systems may not have enough space and not have any major impact on the living area they are situated in. Because the system is meant to be used indoors there should be reliable access to electricity in order to power the system and conditions should not be too extreme where the system would need to do too much extraneous work to create optimal conditions for the plants that are growing.

5.3 – Health and Safety Constraints

Above all else, the first constraint when creating a project design should be to design around user safety. Luckily our hydroponics system does have many safety issues to worry about besides controlling the 120 VAC wall power that will power the unit and ensuring that the high voltage will in no way pose a danger to the operator of the system. The user should be completely safe while operating the hydroponic system and any concerns of power surges, harmful electricity damage, etc. should be addressed in the *Engineering Standards* portion of this document. The hydroponics system should only be affecting the local environment surrounding the plants being grown and should not have any consequential impacts that can possibly pose danger to the users environment.

5.4 – Manufacturability and Sustainability Constraints

As mentioned in section *Economic and Time Constraints* the Coronavirus has had a major impact on the production and supply of electronic components both home and abroad. This may result in a change in the manufacturers we choose to order from versus whom we originally planned and will ultimately order from. Our PCB may change suppliers along with sensors, etc. With this into mind, with early planning and some modification to project plans, our hydroponic system should be built in a similar timeframe as first anticipated.

Our current design is for the system to be able to self-sustain itself in an indoor environment. We plan to have a design compact enough to be kept in a small indoor apartment where major conventional gardening setups may not be possible. With hydroponics anyone can have a simple entry point into growing his or her own food and plants. With this in mind, our system is not tested in an outdoor environment nor meant for an outdoor environment and if used outdoors we cannot guarantee the project's results.

6.0 – Hardware and Software Design Summary

6.1 – Physical Structural Design

Our hydroponics system design was inspired by one of the past projects done by UCF students. The structural design needs to be able to support hydroponics requirements and be stable enough to support the PVC structure, water flowing, the lighting system etc. The structure will be square/rectangular shaped rolling structure which will be divided into 2 compartments. The whole structure will be around 5-6 feet tall and 2-3 feet wide. The lower section will be having a cabinet style design. It will be around 3-4 feet tall. Each corner will implement a 360-degree wheel which will make it easier to move and relocate the structure. The lower cabinet style compartment will hold things like the container for nutrient solution, PCB board, wirings and more. There will be large opening holes on both sides to deliver the water solution from the lower compartment to the upper compartment using PVC pipes. There will also be a small opening from the back to plug the main wiring into the wall outlet. Above the lower compartment will contain the main part of the hydroponic system which are the plants themselves. It will be open with small curtains surrounding it to block any outside interference such as light, temperature such as. Above that will be a ceiling type structure which will hold the lighting system.

The reservoir will sit at the bottom of the entire system serving as a stabilizer preventing the whole system from tipping over since most of the weight will be near the ground. The reservoir will hold around 15-20 gallons of water solution that will be pumped throughout the system at regular intervals. It will also house most of the electronic components such as the PCB board, sensors and more. Sensors will monitor the state of nutrient solution which will be flowing through the system and many other variables will be monitored.

PVC piping will be used for the plant system. There will be 2 rows of pipes running lengthwise across the system. Each pipe will have 3 holes which will have net cups for the plants to be placed. We decide on growing 6 plants simultaneously but that is subject to change. The plants will have adequate space in between when they grow bigger. The net cups will have hydroton, which are clay pebbles to support the plants and are one of the most popular mediums used in hydroponics. Above the plant system will be a lighting system which is a very important requirement for the plants to grow properly. The lighting system will be able to move up and down depending on the plant's growth cycle and how much light is best required.

Our hydroponics system will be designed to be a nutrient film system. Therefore, the nutrient solution will flow through the system at regular intervals. The reservoir will sit directly under the growth channels where it will flow over the plant’s roots and back down in the reservoir. The plants need constant flow of nutrient solution over their roots and for that reason the main pump is chosen that must be able to do the job. The pump must be submersible and be able to run continuously if needed.

Two pumps will be used to keep the pH level and nutrients in the water solution at the desired level. These pumps will have a more precise caliber than the main pump which runs the water solution through the system. To achieve the precision, Peristaltic pumps are the best option. These pumps use a mechanism where the fluid is enclosed in a flexible tube within a circular ring. A rotor with a few rollers attaches to its circumference compresses the flex tube as it rotates it pushes the liquid through the system. These pumps operate at 12 Volts and last a very long time.

The nutrient solution level will be monitored by the water level sensor to ensure the whole system runs smoothly. The nutrient solution level will fall overtime due to the plants absorption and some of it evaporating. When the solution level falls too low, even a small amount of pH level change or nutrient solution will have a significant effect flowing through the system. Since our reservoir won’t be connected to a water supply, we need to be able to notify the user when the water level falls too low.

Temperature sensors will be used to notify the user about the system. There will be one used to monitor the temperature of the nutrient solution flowing through the system. Another will be used to monitor the surrounding air temperature. We were able to find cheap temperature sensors easily. Since we are measuring the temperature of a liquid, we had to find a waterproof temperature sensor.

Our hydroponics system will also implement a seed germination chamber. Our goal with this project is to bring hydroponic gardening to everyday consumers with as low of a barrier to entry as possible. It is a fact that hydroponics gardening is much more successful for adolescent plants. While it is possible to start a plant in a system from the seed, it is often unsuccessful. The results are much better if the seed is germinated outside the system and the plant is transferred afterwards. For this reason, we have decided to include a seed germination chamber with the cabinet area below the system. A separate light source will be included for the germination chamber.

Furthermore, we decided to implement a housing for PCB we will be designing. Because the sensors are connected to this PCB, it is important to seal in a housing that is waterproof so that we can keep it near the water reservoir. The idea of waterproofing a PCB is very common and we can buy a premade enclosure and ensure it is watertight before adding our system.

Moreover, we decided to implement modifying the power strip so that the outlets can be controlled using relays that can interface with the microcontroller. This allows the system to stop with a press of a button on the LCD when for example the reservoir will need to be emptied and refilled periodically. Since the pumps can’t be connected directly to the microcontroller, using relays with power strips can be a way around this.

6.2 – Hardware Diagram

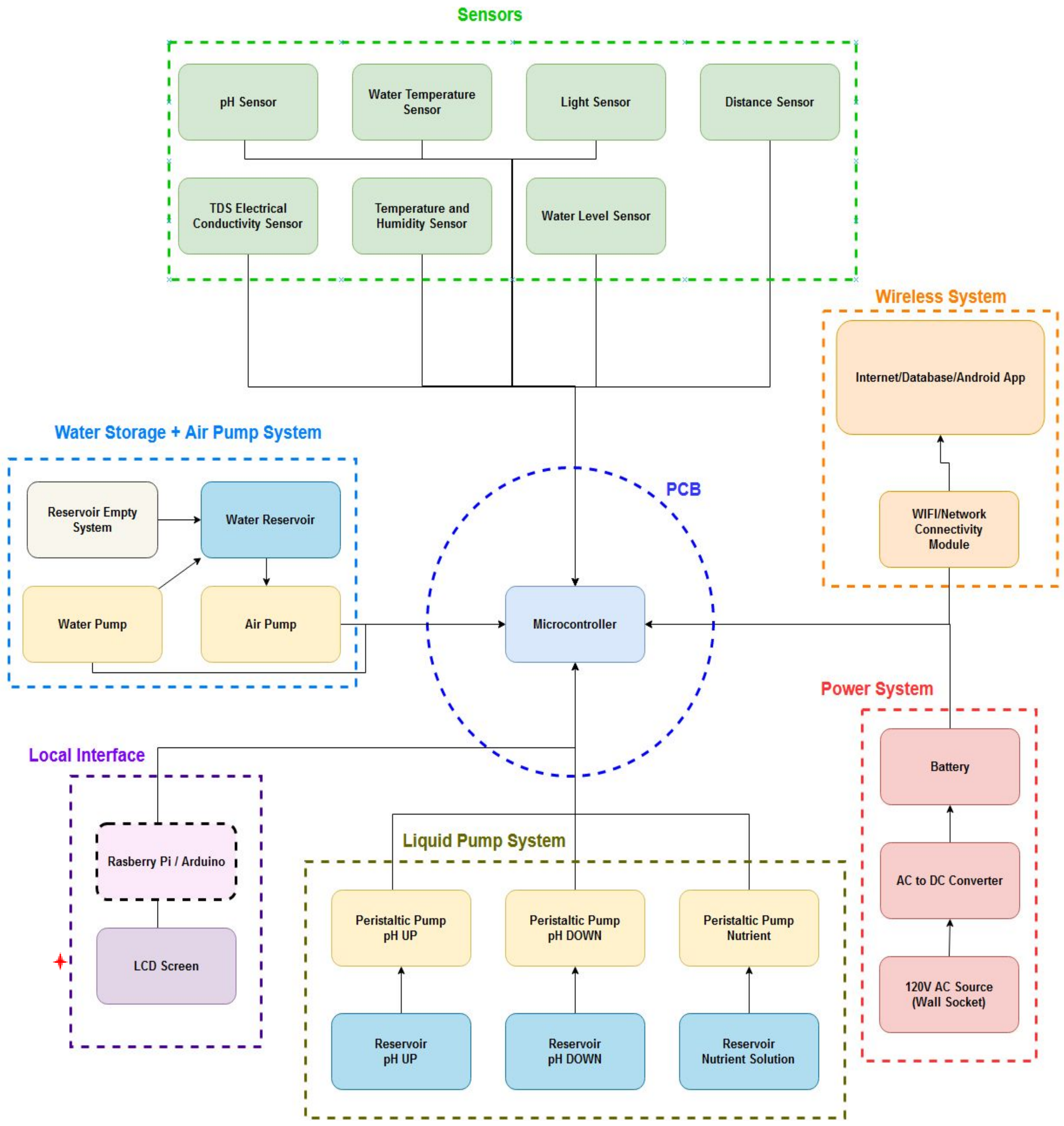


Figure 6-0 – Hardware Diagram: Contains the main components and systems

Note: The Raspberry Pi/Arduino may or may not be implemented based on whether or not we choose to have it as an intermediary interface between the MCU and the LCD or if we choose to directly connect it to our MCU in the custom PCB.

6.3 – Hardware/Electronic Subsystems

In this section we will discuss the hardware and electronics we have decided on for this project. We will focus on the main strategic components such as the microcontroller and sensors and any other key feature of the hardware design. We are still planning on being flexible and if we run into issues in testing are open to design modifications and part exchanging if needed. However, these are our initial design and part choices and we have moved to designing and planning with these parts for initial consideration. As you follow along you may notice interesting decisions and part omission or inclusion, much of this will be explained throughout this section.

6.3.1 – Microcontroller

As discussed previously in section 3 of the report, the microcontroller is a key and central component of our hydroponics system. We need it to be the brains of the majority of our component's and be the central connection between all of our sensors. We compared many different microcontrollers on the market, as discussed in section 3, and were able to come to a conclusion that we felt was best for our design. To reiterate what we were looking for in a microcontroller, we wanted power efficiency, capable processing power (this include features such as memory, ram, clock speed, etc.), Plenty of Pins to connect our various components, Decent IDE and programming support and implementation, and that there are various examples of implementation and documentation online. With all of this in mind we went with one of the six main microcontrollers that were under consideration that we discussed in section 3. We decided to go with the Atmega 2560 by Atmel. The Atmega 328 was also a close consideration but as a result of the fact that the 2560 had much greater support with more GPIO pins and increased memory we decided the Atmega 2560 was a safer, more flexible option for this project.

The Atmega 2560 seemed like a great fit for this project that balanced power and energy efficiency well along with providing more features then we expect to need. This will help make up for any unforeseen issues that arise as far as compatibility and connecting all the various strategic components of our design. There is also support for different low power modes which can aid in the overall systems efficiency and energy consumption. The clock speed is less than that of other microcontrollers, at only 16 MHz, compared to even the Atmega328 20 MHz but it makes up for it in just straight processing power and with better Memory and features. For the sake of this project the clock speed isn't as vital to the overall design as we don't need the faster execution.

Most sensors will be monitoring data and for this kind of application, where we are using sensors to monitor a systems data, the difference we could see in increased frequency wouldn't be as important as the memory and extra features this microcontroller includes. In the figure below we display a basic summary of the specifications of the microcontroller. As you can see there is support for many communication protocols along with support for many different components with a large set of GPIO pins. This will allow us to connect all the sensors we need along with any other electrical / hardware components without having to worry about having enough space for the connections on the microcontroller.

There is also the consideration for the development environment in which we can code the Atmega 2560. We saw multiple, popular options and the Atmega 2560 had some easy to work with options that should aid in making the process of implementing everything more streamline. There was the Atmel Studio IDE which is the standard for this microcontroller, as it is manufactured by Atmel, and there was the Arduino IDE. As briefly noted in section 3, the Atmega 2560 is also used by Arduino in their Arduino Mega and for their dev kits and boards they use the Arduino IDE. This means we have two viable IDE options which can be implemented to work with our microcontroller which made this a compelling option for a microcontroller. This was very important in our decision as the ease of use for programming and loading our code can save us a lot of time in the debug and setup process. With Arduino Mega using this Atmega 2560 microcontroller and with the Arduino IDE being so widely used. There is plenty of documentation and information online about how to work with the Atmega 2560, and as a result of many sensors being designed to work on the Arduino platform, there should be more seamless connections between the Atmega 2560 in our custom PCB and the various components.

For any further specifics on the Atmega 2560, refer below to table 6.1 we previously referenced. You can find information such as pin count, voltages, memory, etc. One thing to point out for this specific microcontroller design, not shown in the table, is that it is surface mounted. Meaning, it does not have pins that can be used to stick to a breadboard, but rather the chips are small and flat and designed to be soldered to a PCB and take up very little space. We will have to learn to work with this design, and have decided it is something we are willing to work with. Included below is a schematic in figure 6.1 that shows the pin layout of the microcontroller. ALong with circuit design for clock and crystal oscillator connections that will and can be used with the microcontroller in order to make it work and run. The schematic is broken into parts A and B for better readability.

Atmega 2560 Specification Summary	
Memory Size (KB)	256 KB
CPU Speed (MIPS/DMIPS)	16 MHz
SRAM(B)	8192 Bytes
EEPROM(KB)	4 KB
16-bit resolution PWM	12
ADC channels	16
Timers	2 x 8-bit, 4 x 16-bit
Operating Voltage (V)	1.8 to 5.5 volts
Pin Count	100
GPIO	86
General Purpose Registers	32
Digital Communication	UART, SPI, I2C

Table 6.1 - Atmega 2560 Basic Specification Table

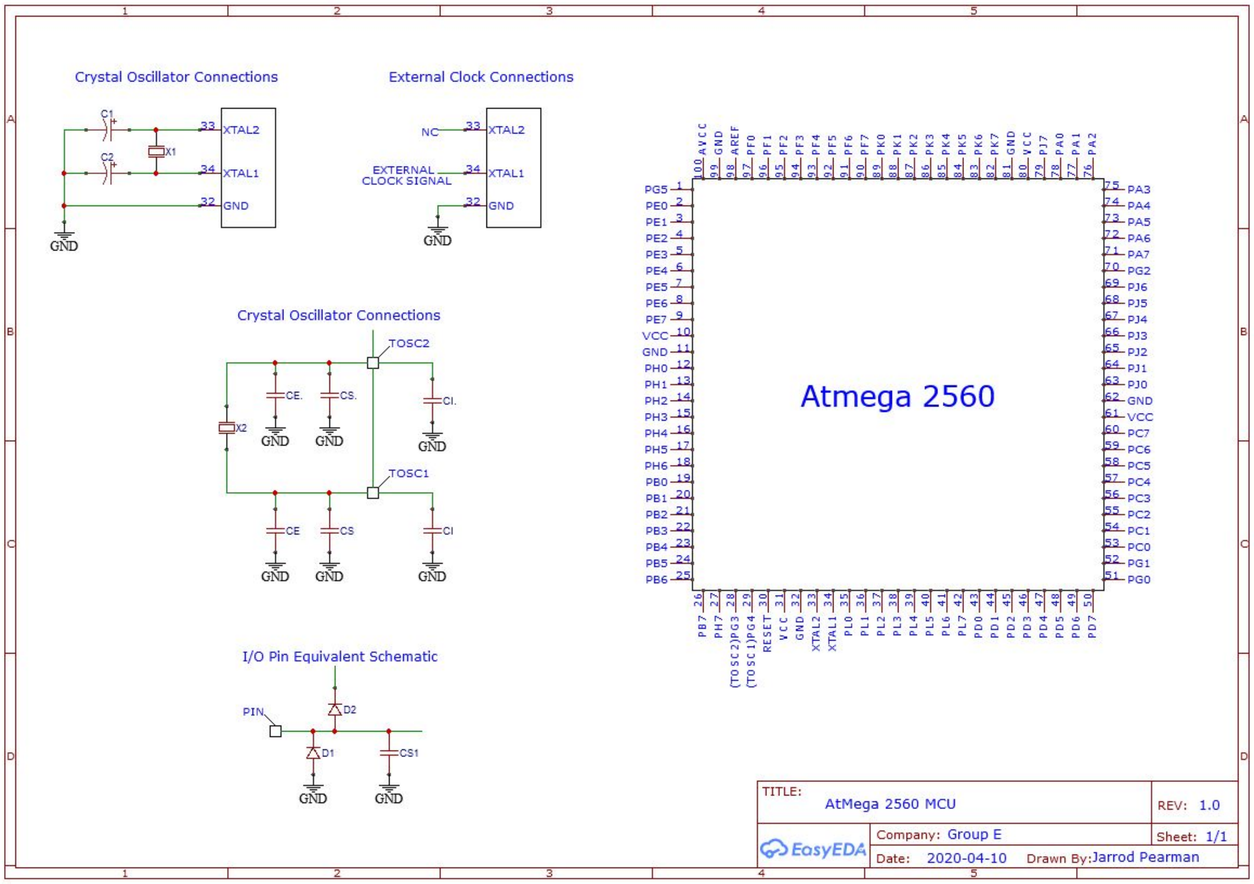


Figure 6.1 - Atmega 2560 Schematic Design

6.3.2 – TDS Electrical Conductivity Sensor

Back in section 3.3.8 we discussed conductivity sensors as a method of measuring the TDS of the water in the system. If you recall, the importance of measuring TDS can be vital in ensuring the plant's health in our hydroponics system. The TDS measurement can be a key indication in water quality as the number of dissolved solids indicates the amount of nutrient contents in the water. These nutrients are what the plants use in order to flourish and grow. Specifically we wanted to measure the dissolved salts in the water, these electrical conductivity sensors allowed us to measure this by sending a current in the water, since the current travels through the dissolved solids, the more dissolved solids will indicate and allow for greater current and indicate a larger amount of dissolved solids. In short, we will generally refer to the dissolved solids as TDS, totally dissolved solids.

After having researched a couple electrical conductivity sensors back in section 3 and after doing research on possible implementations for this project, we came to a conclusion on how we are overall going to implement this sensor. We will discuss overall implementation before moving on to the specific sensor we chose and its features. The plan is to use the Electrical Conductivity sensor to monitor the readings of the water, these will report back to the microcontroller in which we will handle, in our code, how to deal with these readings and if any potential response is needed. The sensor readings will be displayed to the user either through the android app, the local user interface, or both. Along with the user being able to monitor the readings, the systems should respond appropriately to the sensors readings. If the TDS is higher than the desired amount, this will indicate that there may be too much nutrients in the system. As discussed before this can also be bad and should be handled by doing a system water change in the main reservoir. If the TDS readings are within normal range for the current plants then we simply display the value in our various interfaces and report it to our database, no other action is needed. If the TDS is low, we will indicate this to the user along with handling the situation. This will be done by adding nutrient solution to the reservoir. This will be discussed more specifically in the Peristaltic Pump section.

Now let’s discuss the sensor we decided to use for this hydroponics system, as you may recall, back in section 3.3.8 we were comparing two different model sensors from two companies. One was from Atlas Scientific and another from DFRobot. Both sensors follow the relatively same design with a main sensor module and a probe (or 2). The sensor module is connected to the probe(s). The probes will create a voltage difference between two metal tips in which the current will flow and in turn give us our current reading, indicating TDS. This goes to the module which is our connection and handles the probes interface. This will connect to our microcontroller. Now with them having similar design it mostly came down to features and cost. The Atlas scientific had more features compared the DFR Robot while also supporting a large range of readings. There is also a greater accuracy of within 2 percent while the DFRobot has an accuracy of within 10 percent. With this improved accuracy and features, including more reading modes, there is a premium on the cost. We did consider the benefits and compared the different specs. Part of this discussion can be recalled back in section 3.3.8. Our choice ended up being the DFRobot Analog TDS Sensor/Meter. This is a result of two major components. First were its features, it measures a large range of values and could work with both 3.3 volts and 5 volts input voltages. This helps with compatibility. There is also 10 percent accuracy, which despite being the worst of the two options, is plenty for this hydroponics system as we aren't looking for that intense of accuracy. The readings are used as a general idea on the water quality in order to indicate when more nutrients need to be added. The specific number of the reading, however useful, isn't needed to be exact to the point. We just want a close enough value to get an idea on how to report to the user and handle readings accordingly. We will still display the value to the user, it may be important to note in our interface the potential accuracy. The second main reason for choosing this sensor over the Atlas Scientific was its overall Cost. The DFRobot sensor, including the probe cost around 15 dollars, in comparison to the Atlas Scientific which is over 40 dollars and requires the probe to be bought separately, which itself can be another 50 dollars. With the DFRobot being much cheaper and the specs being enough for the scope of this hydroponics system, it was a no brainer for this project. The DFRobot sensor, as seen in table 6.2 below, is flexible with input voltages of up to 5.5 volts, working currents of 3 to 6 mA with a large range of readings. For more specifics on the spec go to table 6.2 for information on dimensions, interface TDS range, and more for both the sensor board/module and the TDS probe that comes with it. Below is an included schematic , figure 6.2, drawn to show how the TDS sensor works, the overall schematic includes 1 large system of connections and 2 separate systems. Here you can see the main components that make the TDS system work and the connections to the pins that we will use to connect our microcontroller. Along with the decent specs at a low cost there was the fact that it was designed with Arduino in mind. With our Microcontroller being used in Arduinos, this should help with compatibility and make connecting everything a smoother process. Plus, there is the fact that there is more documentation and examples online for anything designed to work with Arduinos, as a result of the popularity.

DFRobot – Gravity Analog TDS Sensor/Meter	
Signal Transmitter Board	
Input Voltage	3.3 ~ 5.5 Volts
Output Voltage	0 ~ 2.3 Volts
Working Current	3 ~ 6 mA
TDS Measurement Range	0 ~ 1000 ppm
TDS Measurement Accuracy	+/- 10% F.S (25 ℃)
Module Size	42 * 32 mm
Module Interface	PH2.0-3P
Electrode Interface	XH2.54-2P
TDS Probe	
Number of Needle(s)	2
Total Length	83 cm
Connection Interface	XH2.54-2P
Color	Black(no-refill) or Blue(refill)
Waterproof Probe	Yes

Table 6.2 - TDS Sensor Spec. Table for Module and Probe

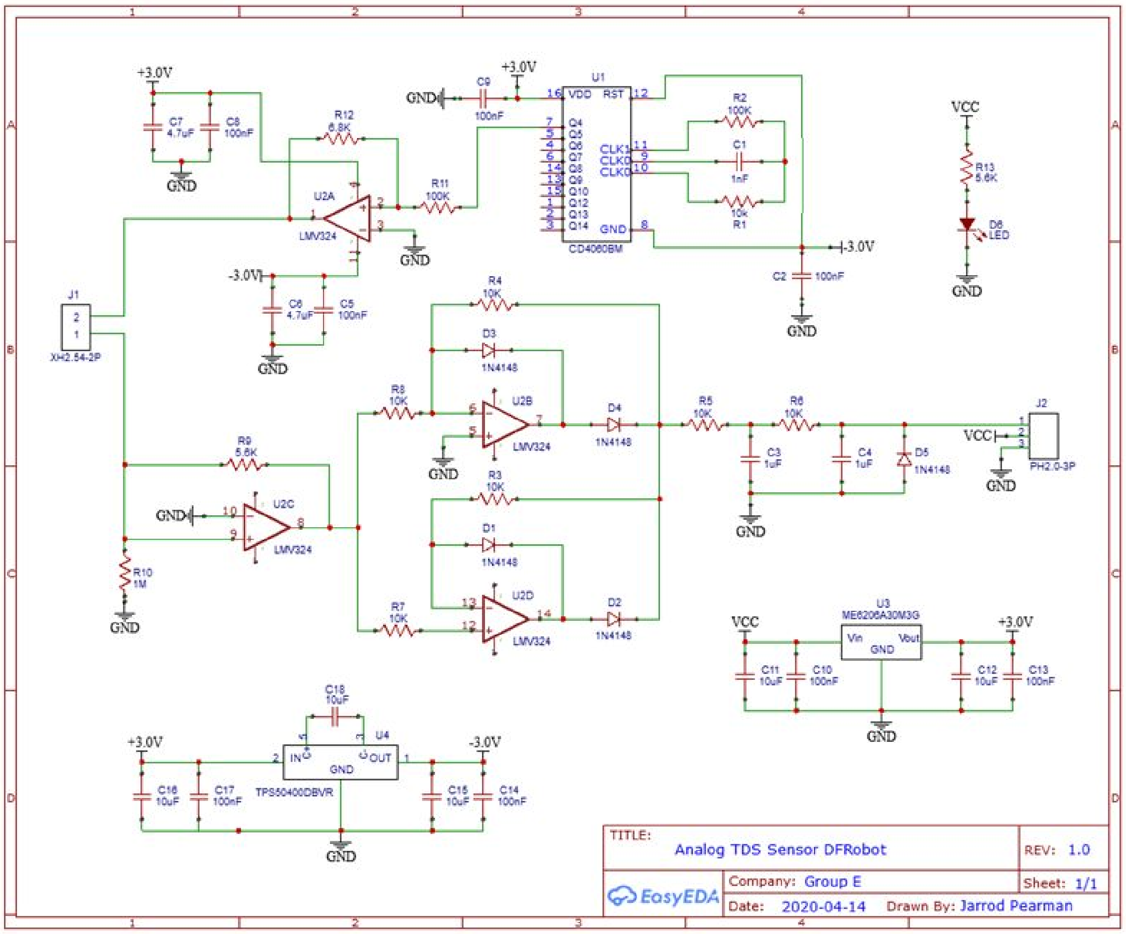


Figure 6.2 - TDS Sensor Schematic

6.3.3 – pH Sensor

We saw previously in section 3.3.2 that pH can have a large impact on the health of plants in a hydroponics system. Along with TDS as discussed in the last section. pH can be a large proponent to the water quality in our system. For this project we want to measure the pH to ensure it's within a range that is good for working with the specified plants. In general, many plants like their pH around 6, give or take 1. As a reminder, pH is measured on a range of 0 to 14 with anything greater than 7 being increasingly basic. While anything below a measurement of 7 being acidic. 0 would be max acidity and 14 would be the most basic. In Section 3 we discussed a few options for pH sensors. There were two from Atlas Scientific and a 3rd that was found on AliExpress, although after further research there seems to be many brands that produce the same sensor but with their own branding and selling it. For the AliExpress Sensor component, when we discussed it, it included all these various manufacturers selling the identical components. The Atlas Scientific Sensor options left us with a digital and or analog option. Both were accurate with the digital being most, with an accuracy of 0.001 to 14.000 as we saw back when we discussed the various sensor options back in section 3. The analog option, as we saw, was viable as well and just lost some accuracy compared to the digital version. Much like for the TDS sensor the Atlas Scientific sensors are very nice but have a cost premium. Along with this the probs is sold separately and also has a cost more than that of the original sensor module. If we need increased decimal precision and or premium accuracy such as in human health related scenarios this would be key, but for the sake of this hydroponics system we simply need to monitor the general pH levels and handle them accordingly. As a result of the increased cost and lack of exact precision, we looked to our third option. This was a sensor module + probe bundle found on AliExpress as well as some Amazon listings, some say manufactured by a company called Reland Sung another by a company Vokatta. The cost was much lower than that of the previous options while also having good reading and accuracy. The specs stood up to what we needed to achieve this project and did so affordably. This sensor can be found around 10-15 dollars on AliExpress and up to 30 dollars on Amazon.

Of the various ones on AliExpress and the various brands we went with the Vokatta one as the specifications and imagery shown provided enough information for us to get an idea of its capabilities along with there being many good reviews on the product. Many sellers didn’t provide enough information to trust the product, we need as much information as we can get to make sure this can be implemented properly with our system. As seen in the following table 6-3 below, spec wise, this pH sensor module and probe setup should be great for the scope of this project supports all pH value ranges and has low power consumption. One main concern with this sensor is the lack of documentation online, outside of the seller listing and store page there isn't many example online, at least when compared to other sensor on the market, despite this, for the much lower price we found it worth the risk as we can potentially fix and issues or confusion with our own testing. After having decided on a pH sensor we had to choose a probe, this sensor can be bought with either a black or blue probe. Both do the same job but offer a different design feature. The black probe comes as is and contains a solution that makes it work. The probe is locked closed and can't be modified. Alternatively, the blue probe can be open and have the solution refilled. This potentially allows for more maintenance and easier changing of the solution if needed. For the sake of this project however we find that the black probe will suffice. We don't want to add any extra steps for the user to clean and change the solution out of the probe. Especially considering the cost, we feel it's safer to get the black and make sure it works. If we run into longevity issues, we can replace with another black for the sake of this prototype or get the blue probe and use it along with our current sensor module. Any other specifics on the module and probe can be seen further in the table 6.3 below. You can also find a schematic of the pH sensors system in figure 6.3 below. This schematic shows the components that make up the sensor and their connections to the various pins that we connect to our microcontroller. Also you can see in the schematic there are 6 pin connections and the diagram is broken into 6 sub schematics, 2 of which include indicator LEDs. Look further for more design specifics.

As far as how we are going to use and implement this sensor, the overall design will be much the same as the TDS sensor use case. We will measure the pH readings and our microcontroller will provide the user with information of its current state. This is for monitoring purposes. The microcontroller will also look at the pH's current values and check whether it falls within a defined range for the current plants in the system. IF the pH is too high or too low, we will handle accordingly. Otherwise the microcontroller will do nothing but send the information accordingly such that it is displayed to the user. If the pH is too low then we will use one of our peristaltic pumps to add more pH UP solution to the water and if it's too high we will use another pump to add pH DOWN to the water. Specifics on how these pumps will work and on the specific pump we will be using will be discussed in a later section.

Liquid PH Value Detection Regulator Sensor	
Liquid PH Sensor Board	
Heating Voltage	5+/- 0.2 V (AC -• DC)
Output	Analog Voltage
Working Current	5 ~ 10 mA
The detection concentration range	PH0-14
The detection range of temperature	0-80 centigrade
The response time	≤ 5S
Stability time	≤ 60S
Power consumption	≤ 0.5W
Size	42mm x 32mm x 20mm
The working temperature	-10~50 centigrade (the nominal temperature 20 centigrade)
Working humidity	95%RH (nominal humidity 65%RH)
Service life	3 years

Table 6.3 - pH Sensor Spec. Table for Module and Probe

Liquid PH Value Detection Regulator Sensor	
BNC PH Electrode Probe	
PH range	0-14 PH
Zero-point	7 ± 0.5PH
Alkali Error	0.2PH
Theoretical Percentage Slope	≧98.5%
Internal Resistance	≦250MΩ
Response Time	≦1min
Operating Temperature	0-60℃
Terminal Blocks	BNC plug

Table 6.3-B- pH Sensor Spec. Table for Module and Probe

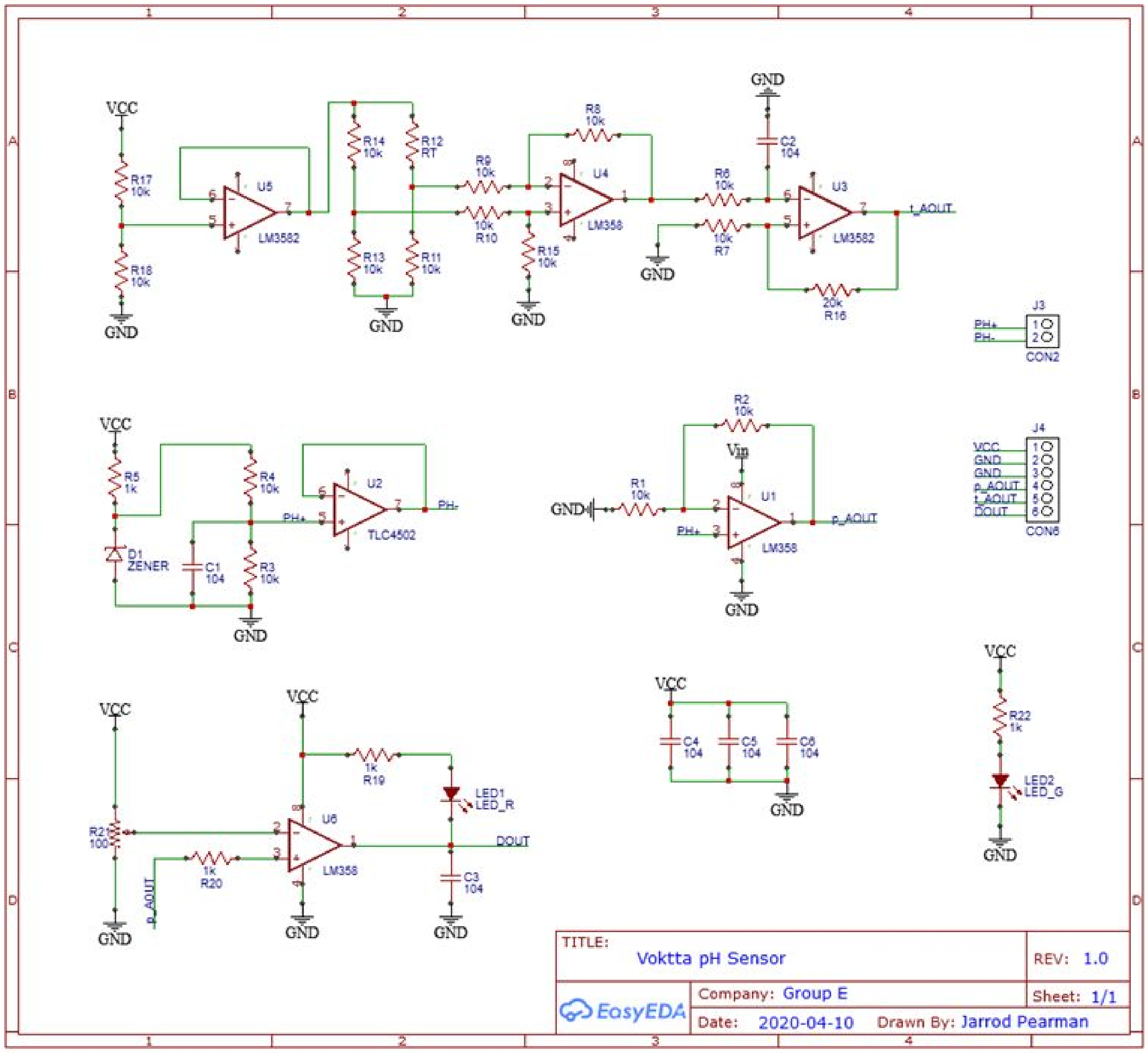


Figure 6.3 - pH Sensor Schematic

6.3.4 – Temperature and Humidity Sensor

Back in section 3.3.3 we discussed temperature and humidity sensors. One of the reasons for combining both these measurements in the same conversation is that many of the sensors made in the industry that provide temperature measurement functionality also include humidity measurements. As discussed before, measuring the temperature of the environment of the hydroponics system is important. Different plants across the world grow in their respective environments, with each being different. If you're trying to grow some mainstream plants, this may be less of an issue, but we want to ensure that our hydroponics system can support a large range of plant growth types. As a result, we need to keep track of the temperature and humidity to ensure the environment's ecosystem is correct for the respective plant. Temperature is key as we want to ensure the plants have the correct amount of warmth and heat to grow. Things to avoid are burning the leaves from overheating and frostbite from freezing temperatures. Now, these are extremes but the general idea still applies. Humidity is another key aspect that can have a large effect on the health of the plants. It can affect processes such as photosynthesis, respiration, and pollination. If the humidity is too high, this will create an environment in which mildew and various molds and fungi can grow. These buildups can take over the spores of the plant and eventually end up killing it. Another factor of high humidity is that it can reduce the plants ability to breath. With all of this reiteration on the importance of this sensor, we can move onto our decision and how we got there.

Back in section 3.3.3 we looked at two different temperature sensors, one from Atlas scientific and another by SMAKN. The Atlas Scientific one offered greater precision with a large range of temperature readings. There was also the case of initial compatibility with Arduino and raspberry pi. This would be great with cases where this large range and precise accuracy were of utmost importance, but for this project we can most likely get by with less accuracy, as once again these are just for monitoring the consensus of the systems environment. If we find an issue with accuracy, we can always account for it in our code. Keeping this in mind, the Atlas Scientific Sensor, specifically called the EZO-RTD Resistance Temperature Detector, is also more expensive than the other option options on the market. We then looked at our second option from SMAKN which incorporates the DHT22 Temperature and Humidity sensor into a module. This entire sensor module makes it connect more easily with Arduino and raspberry pi systems. Which is why it has become a popular option. It has a way smaller temperature reading range but for its low cost, going as low as 10 dollars on amazon. As seen in Table 6.4 below the temperature range for this sensor is still plenty for the scope of this project, unless you plan on growing hydroponics in extreme climates. There is also a strong accuracy, despite not being as high as the atlas, but is perfect for the scope of our design. Overall, we chose the second sensor since it included temperature and humidity sensors (unlike eh Atlas Scientific, which only included temperature), was low cost, and was popular and widely used. With this in mind the SMAKN DHT22 Sensor Module is our choice for this project and will be a safe option. Specifics on the DHT22 can be found below in Table 6.4 for information on range, accuracy, and any other specifics.

The implementation for this sensor will be relatively simple. We will get the readings from the sensor. The sensors temperature and humidity readings will be taken by the microcontroller and sent to the various modes of user interaction. One to the server and database such that the android app can connect and read the info and local data to be displayed locally to the user. The system designs are primarily planned for indoor use so we have no exact plans to add systems of climate control. So, for this sensor we don’t plan on handling the temperature and humidity automatically ourselves. The readings will be given to the user and the user can decide on how to handle it. Potentially, if time and cost allow, we will implement ways of climate controlling the plants environment. This would be done accordingly based on the readings being either high or low. This would most likely be done through a system of fans, coolers, and heaters.

Temperature and Humidity Sensor	
SMAKN DHT22	
Temperature Range	-40℃ to 80℃
Temperature Accuracy	+/- 0.5℃
Humidity Range	0-100%
Humidity Accuracy	2-5%
Sampling Rate	0.5Hz (Once every 2 seconds)
Max Current	2.5 mA
Operating Voltage	3 to 5 volts
Resolution	16 Bit
Signal Output	Digital
Dimensions	28.2mm (L) * 13.1mm (W) * 5.5mm (H).
Number of Pins	4

Table 6.4 - Temperature and Humidity Sensor Spec. Table

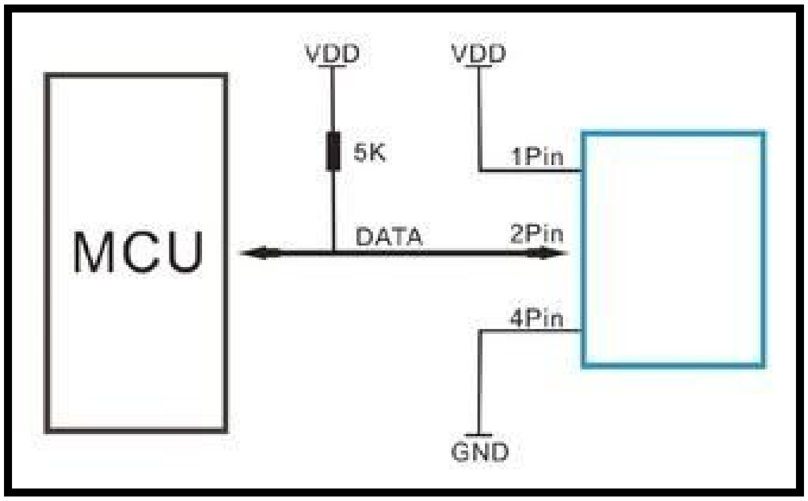


Figure 6.4 - Temp and Humidity Schematic

6.3.5 – Light Sensor

Now let’s discuss our choice for light sensor and reiterate why having a light sensor is important. Light, as we know, is a key part in plant growth. Plants use light in order to commence the processes they need to prosper and grow. As Discussed back in 3.3.6 plants can prefer differing periods of light exposure. Some may prefer long exospore periods of up to almost 20 hours, while some prefer as low as 6 hours of light per day. This can be controlled and set in software; however, we wish to keep track of the light’s intensity. Being able to set time periods of ON and OFF with lighting isn’t a difficult issue to solve, however, we wish to also account for how bright and intense this light is for this time period. If you have a rather strong light going on for the same amount of time as a weaker light. There are going to be differing effects on the plants in the system. If the light is more intense, potentially want to shorten the amount of light exposure in order to not burn out the plant. Also, if the light isn’t intense enough, we may want to increase the duration of the light that is on. Outside of this, we may directly wish to change the intensity, assuming the lighting allows for it. We want to be able to set light intensifies for different plants, and a light sensor will help us keep track of our readings and handle it accordingly.

For this project we considered three different sensors back in section 3.3.6. The options were pretty close in design and features. All seemed like viable options as far as compatibility, protocells, measurement ranges and sensors. Even with their differences we only cared if the data would suffice for this design. Such that our decision came down to practicality, information available online, reputable, and documentation. For specific, you can go back to section 3.3.6 on the three sensors considered. Our decision for this project needed up being the Adafruit TSL2591. It is a popular sensor, its low cost being under 10 dollars, and has great specs that should work for this hydroponics system. This sensor allows us to measure multiple spectrums of light including infrared, full-spectrum, and human visible light. This sensor is also low power, pretty precise, large range, and flexible in design for connecting it to a microcontroller. More information on the sensor such as the specifications are in Table 6-5 below. Along with this, there is a lot of documentation and information online about how to integrate this sensor into systems such as Arduinos. This will help greatly in learning how to implement our design. Below, in figure 6.5 we also include a schematic for the design of the light sensor that will help show how it works along with any potential connections. Using the schematic you can see the various pin connections and circuit logic that makes the sensor work without a microcontroller along with various small components that are used to make up the sensor.

Light Sensor	
Adafruit TSL2591	
Lux Range	188 uLux up to 88,000 Lux
Voltage Range	3.3 to 5 V
Dynamic range	1 to 600,000,000 Counts
Interface	I2C
Address	I2C 7-bit address 0x29 and 0x28
Dimensions	19mm x 16mm x 1mm / .75" x .63" x .04"
Temp. Range	-30 to 80 *C

Table 6.5 - Light Sensor Specifications

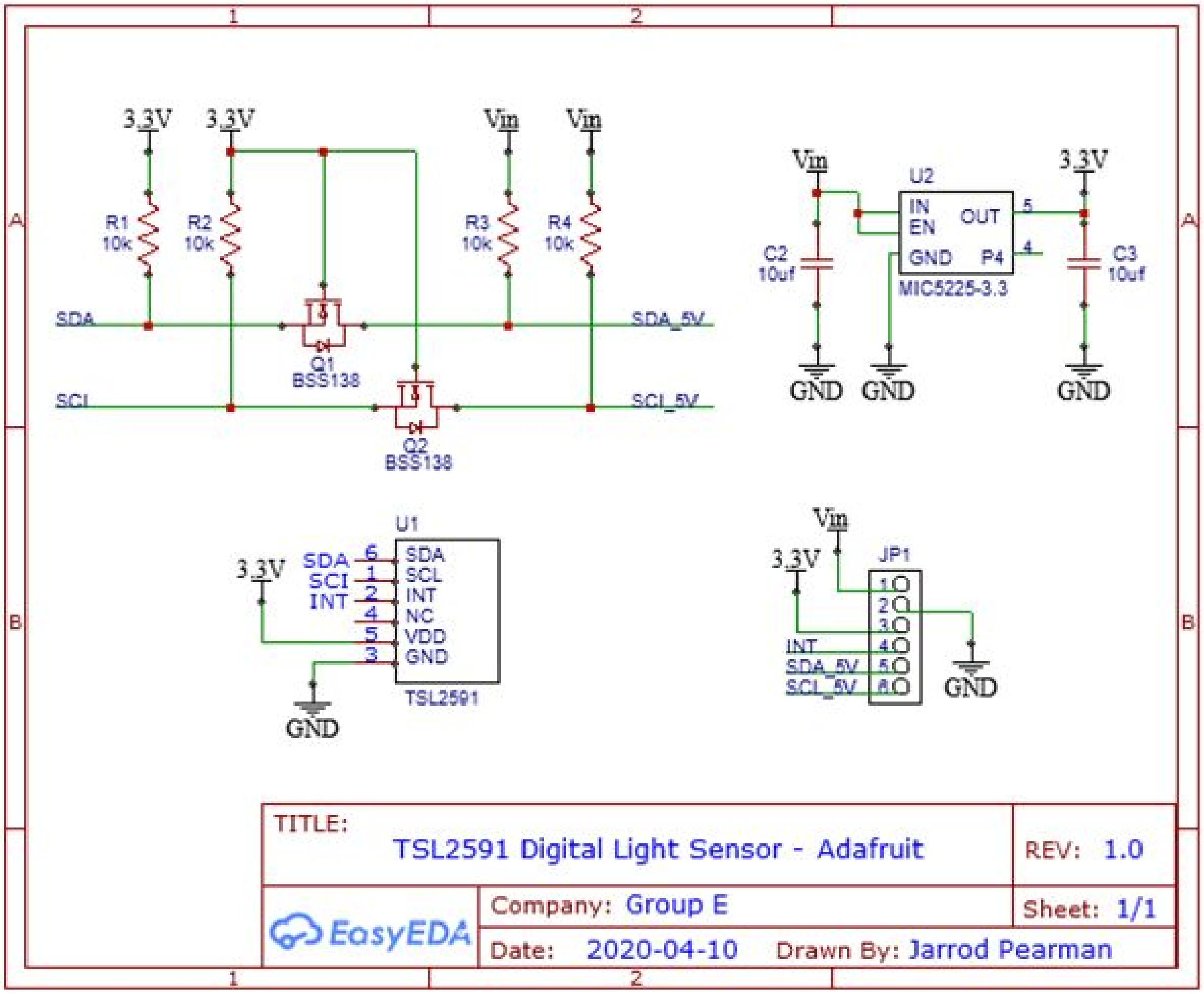


Figure 6.5 - Light Sensor Schematic

6.3.6 – Water Temperature Sensor

As discussed in section 3.3.4, not only air temperature can be important. It is important to monitor the temperature of the water within the system's various reservoirs and volumes. We discussed why back in section 3, but to reiterate. It can have many varying effects on a system such as the amount of dissolved oxygen, plants ability to absorb nutrients, potential for bacteria and fungi, and more. With this in mind, we discussed that we wanted to implement a sensor. In our research we came down to only one option, in which we discussed. Since then we looked further and were able to find other options. Despite this we went back to our original option discussed in section 3. This was a result of overall cost, precision, measurement range, and other features. This specific sensor being the DS18B20.

The DS18B20 is an extremely low-cost option that is also simple and easy to use. Its design and features have the Arduino in mind. This means implementing into our system which we have chosen to run with an Atmega2560, which can be found in many Arduino systems, should be a simpler process. The sensor consists of a small PCB module and a probe. This probe is waterproof and goes into the water in which it gets its readings. The PCP board connects to our microcontroller, in which it interprets the probes readings and passes the information. The probe can read a large range of temperature and is precise within half a degree. More specifics on the sensor is below in Table 6-6.

As far as how the sensor will be used with the overall system, it will be used to monitor and potentially handle situations based on how our final implementation goes. We will of course display the readings to users so they can keep track of what is happening in the system. We also may want to automatically handle the situation. If the water is too cold, we can heat the water up with a water heater, these are often used in aquariums for specific types of fish. On the other hand, if the water is too hot, this would be more difficult to deal with. This isn’t directly an issue but water coolers tend to be more on the expensive side. As a result, this may be a feature best left for future iterations of the design, in order to reduce the overall system cost and stay within our budget. It is needless to say however that if cost wasn’t an issue it could be a nice feature to implement. Heaters, while less expensive, can also be costly compared to other components such as most sensors, and with this being planned as an indoor system we will assume, for now, that the water temperature sensor will focus on monitoring and displaying information for the user to handle themselves.

Water Temperature Sensor	
DS18B20 Temperature Sensor Module Kit Waterproof	
Temperature Range	-55 ~ +125 °
Supply voltage	3.0V ~ 5.5V
Sensor resolution	9 to 12 adjustable
Adapter Cables	DATA, VCC, BLK

Table 6.6 - Water Temperature Sensor Specifications

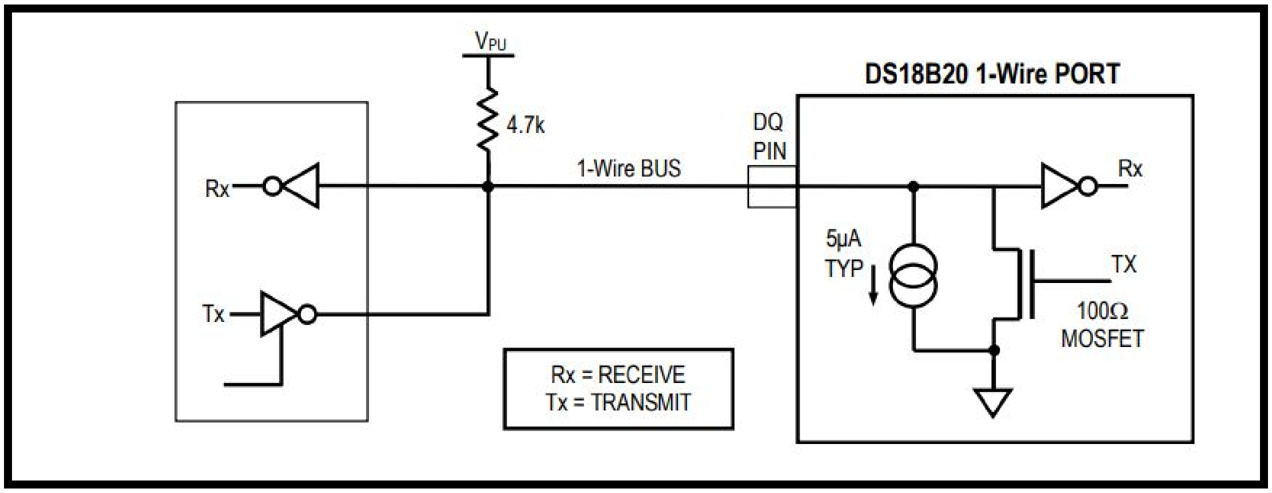


Figure 6.6 - Water Temperature Sensor Schematic

6.3.7 – Water Level Sensor

Outside of sensors for measuring data and monitoring the health of our system, we also need to keep track of certain measurements. One of these being the amount of water within the system. Water is the whole basis of which hydroponics works. Plants roots get their nutrients from the water flowing through the system. Over time, the volume of water in the system, including the plumbing and various reservoirs, will decrease as a result of evaporation. As a result, we need a way for the user to recognize a need to add more water. The use of the sensor will be to measure the water level/depth in the reservoir. We will use this data in reference to our calculated minimum height in order for the system to operate. This minimum height will be the water level in which all the plants get the required water depth they need for their roots. The goal is to keep the roots of the plants submerged such that the system can work without issues. Back in section 3.3.7 we discussed two water level sensor options. One from Adafruit which they called eTape and another by CQRobot which follows a sensor board plus probe design. The Adafruit design, as discussed in section 3, is a compelling option since its design is simple and it comes in various different depth sizes. As a summary, this sensor looks like a ruler and is meant to be submerged in the water. As the water rises, raises or lowers, it puts differing pressure in this eTape ruler strip. This change in pressure causes a change in resistance in the sensor. The resistance created is what is used to measure and determine the water level. We take this resistance in software and calculate accordingly to determine the water level. On the other hand, we look at an option from CQRobot which follows a design relative to other sensors we have gone with. There is a sensor PCB board module that connects to our microcontroller, in which it interfaces with a photoelectric probe to measure the water level height. This probe then measures water depth and passes this information, through the sensor's PCB module to our main microcontroller's PCB. Both options have good specifications for this project but vary largely in cost. The eTape is largely more expensive at over 50 dollars, up to 100 for the largest model, while the CQRobot option is between 10-20 dollars. Even though the eTape design is simpler and leaves less room for error, we decided the extra cost was not worth the benefit and went with the CQRobot water level sensor, its sensor module should help interface it to our microcontroller. WE discussed more about how this sensor works back in section 3.3.7 and for specifications on the sensor you can reference Table 6-7 below.

As discussed, we will use the reservoir water depth to determine if the water level is low or not. If the water level does not supersede the minimum determined level, we will inform the user via the android app and potentially locally. It will be left to the user to refill any lost water due to evaporation, but it is important for us to inform the user to reduce system failure. We can also in the software disable any components that functionality could be hindered by lower water levels. Such as disabling the pumps so they don't run dry and burn out. The reason for lack of automation is that it would require a hard-connected waterline for a source of water, which in general is more costly and complicated for the scope of this design. For a larger, more permanent implementation it could be worthwhile.

Water Level Sensor	
CQRobot Contact Water/Liquid Level Sensor	
Supply Voltage	DC 5V
Output Current	12 (mA)
Low Level Output	Less Than 0.1V
High Level Output	3.3V or 5V
Detection Accuracy	+/- 0.5mm
Measurement Range	NO Limit
Operating Temperature	-25 to 105 °C
Probe Length	50cm
Dimensions	30mm * 30mm

Table 6.7 - Water Level Sensor

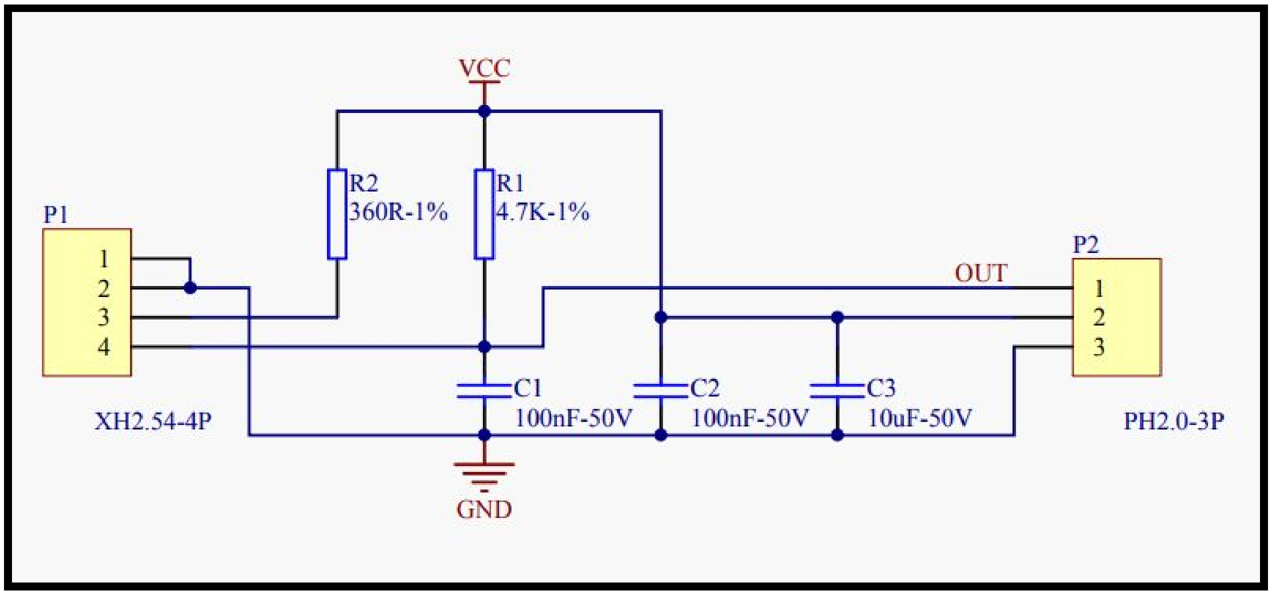


Figure 6.7 - Water Level Sensor Schematic

6.3.8 – Distance Sensor

At this point in the project there are certain circumstances in which we may need to measure distances. The main purpose we discussed is to have a lighting system that raises and lowers with the height of the plants. This would be to ensure that the Lighting grows with that plants such that they receive constant lighting throughout their growth cycle. For this project we ended up sticking with the sensor we discussed back in section 3.3.10. This sensor being the HC-SR04 which is a flexible and inexpensive option. Specifics on the sensor can be found back in section 3 and in Table 6-8 below. The lighting could be raised up and down manually by the user or mechanically with electronics controlling it. That decision will come down after everything else is implemented. It is less of a priority and if budget becomes an issue we will go with manual or even light intensity modification in or to still implement this feature. The implementation of the sensor will be simple, the readings will pass to the microcontroller in which it is further represented to the user through the various interfaces. If manual, the data will inform the user that there is a need for the lights to be raised, if automated the system will raise the lighting through the hydraulic/pulley system. More information can be found back in section 3 for the overall purpose of this sensor.

Distance Sensor	
HC-SR04	
Operating Voltage	5V DC
Quiescent current	< 2mA
Measure Angle	<15 Degrees
Distance Range	2cm to 500cm
Resolution	0.3 cm
Pins	4

Table 6.8 - Distance Sensor Specifications

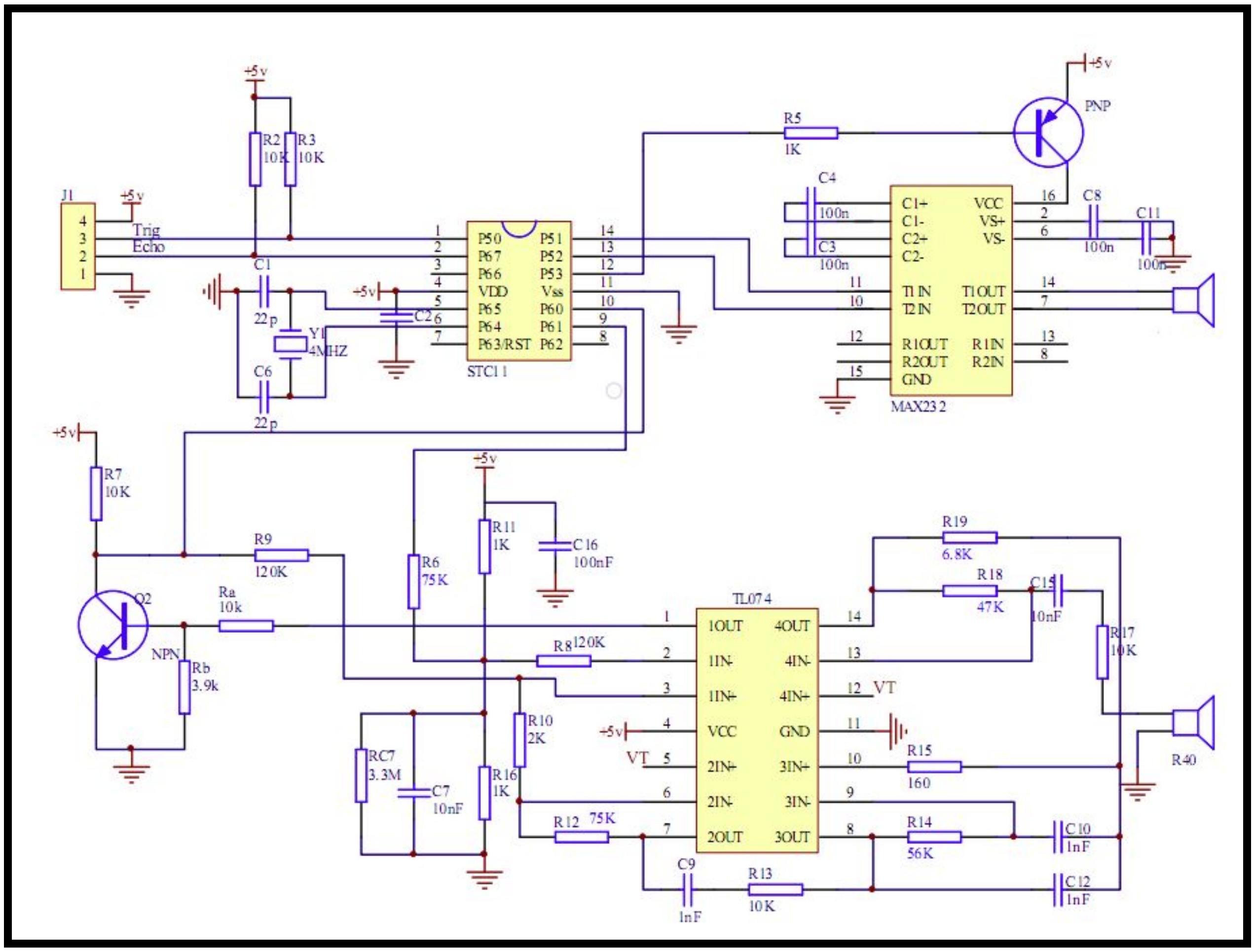


Figure 6.8 - Distance Sensor Schematic

6.3.9 – Arduino Mega 2560 Dev. Board

Something that we didn't specifically discuss in section 3 was the need for a dev board. After having decided on the Atmega 2560 microcontroller we need a system in which to test our code and sensors while our custom PCB goes under the design, model, and manufacturing phase. One of the popular systems on the market that uses the Atmega 2560 is the Arduino Mega 2560. This board uses the Arduino IDE and supports many features which are plenty for this design. In fact, our PCB design will be somewhat of a toned-down version of the Arduino Mega.

The reason we need this board over waiting for our custom PCB to finish is simply time. These Arduinos already come with everything you need to get started with the microcontroller and are designed and supported by many sensor and component manufacturers. We don't have the time to wait a month between PCB designs to be manufactured and shipped to us from our PCB vendor, so while we wait we need to be able to work on the hydroponic system. We can take what we learn with this breadboard and implement it into our PCB design. It will help show us aspects of design to follow in our own custom PCB while allowing us to test the connections between the various components. It will also allow us to get a head start on the software. Having this board can also be useful as in many cases it can be used to program our custom PCBs microcontroller. We can program everything on the Arduino and upload our code to the custom PCB. This can save time and help troubleshoot. If the code works on the Arduino but not the custom PCB then maybe there is an issue in the PCB design. An additional use for the Arduino is that of running our LCD. If we find that our choice of LCD is putting a large enough load on our custom microcontroller or that the custom PCB design is getting more complicated than it needs to be as a result, we can adopt a master slave microcontroller system in which our custom PCB is the master while the Arduino is the slave that solely communicates with the LCD. The design decision here is left more to whether or not our custom PCB is able to support the LCD effectively. More will be discussed in the LCD section of this report. Overall, the Arduino will provide us with a platform in which to test our code, component compatibility, and help aid on our own PCBs design and implementation. The specifics on the Arduino board and the connection limits and more can be found in Table 6-9 below.

Arduino Mega 2560 PCB Board	
Operating Voltage	5V DC
Input Voltage (recommended)	< 2mA
Digital I/O Pins	<15 Degrees
DC Current I/O Pins	2cm to 500cm
Microcontroller	0.3 cm
Total Pins	4
Dimensions	101.52mm L, 53.3 mm Width

Table 6.9 - Arduino Mega Specifications

6.3.10 – LCD Screen

The hydroponics system we have decided to build automates a lot of the components such as the lighting, EC, and pH sensor. It uploads the data to the phone application where the user can check the data and make appropriate changes to the system. That is convenient for the user when he/she is not home and away from the system. If the user is near the system and wants to check the readings, it would be a hassle to pull out the phone every time and open the application. Thus, we are implementing a simple LCD screen which will fit on the system itself and will display the sensor readings to the user.

There are two routes that can be considered when determining how the LCD screen will operate. One is the capacitive touch method which is found nowadays in most smartphones and tablets while the other is the conventional use of navigation buttons. For the capacitive touch, the screen not only acts as the display but also the control. It detects the user's touch based on the electrical properties of the body. While the touch screen is very popular these days and more and more things are moving towards that, the design aspect of that can be quite challenging compared to the on-board button control. Further, the screen must have good response time and have the sensitivity tuned properly for it to be reliable. On the other hand, the capacitive touch has much more flexibility in the layout and they are much more appealing to the eye.

On the other side of touch screens are the on-board button control screens. This type of screen is more durable than a touch screen. Although it may not be as attractive to use this type of screen, the design avoids touch errors that can cause issues for the consumer. This is also good for optimizing performance and making sure a mis click by the user won't result in a huge cost. Since we are working with a system that can get messy and includes water, protection from the elements is very important. The benefit of having a button controller is that enclosures can be made without the loss of functionality which will protect the device.

LCD displays with buttons allow for a simple navigation menu and interaction. While this is not the most visually appealing, the greater reliability and ease of use would be beneficial to the user. We researched different types of LCD and discussed whether we wanted a high-end display with low external buttons or a simple display with integrated buttons. The high-end LCD would look much better and give a better user experience, but it will add a lot of cost to the project. Using a display with integrated buttons would cut down on errors and simplify the design process.

Keeping that all in mind, we have decided to go with 7" Arduino Touch screen Shield. It is compatible with the popular Arduino boards. Even though the touch screen displays might not be as reliable, the size of this LCD display is what we wanted. a 7-inch display can clearly display every sensor information by just a press on the screen. Its appearance and ease of use for the user is why we decided to implement a touch screen display.

This display has a microSD card connected. It also has a 14 white-LED backlight and has 800x480 colorful pixels. The shield comes fully assembled. There is no need for wiring or soldering. The power supply is of 5V which can be connected to the PCB. There is also an open source graphics library which we can use when coding the LCD. The LCD display requires high current to operate and an external 5V power supply will be needed which will satisfy the requirements.

Specification	**Value**
Max. operating voltage	5V
Max. operating current	480mA
Diagonal size	7”
Display format	800x480 pixels
Max weight	0.796lbs

Table 6.10 – Specification for LCD display

6.3.11 – Peristaltic Pump

One of the most important features of our hydroponics system is that it automates the process where a user would add nutrient or pH buffer solution to the water after they measure these parameters using the sensors. The best way to add liquid solution to the water reservoir with the control of a microprocessor is using electrical pumps. Each liquid which will be needed to the water reservoir will need its own motor so each can be added independently.

The liquid pump design that has been decided upon is called the peristaltic pump. The way it works is very similar to our digestive system. When we swallow food, our muscles contract in sequence forcing food and beverages down to our stomach. The tube gets smaller behind whatever has been swallowed. With a peristaltic pump, the tube is fitted within a circular pump casing. A rotor pushes several rollers against the tube and pinches the tube off. This pushes the fluid within the direction of the moving rollers. Peristaltic pumps can work with DC motors, thus making them easy to interface with microcontrollers and not having a huge impact financially.

The main objectives that a peristaltic pump must accomplish are: be able to deliver the liquid solutions from the outside of the reservoir to the inside, be able to send precise amounts of liquid solution that can be controlled by the microcontroller, and be able to use a simple DC motor and common tubing for the transfer process and pump assembly. Table xx shows the specifications which the peristaltic pump must maintain.

Specification	Value
Motor	DC
Max. operating voltage	5-6V
Max. operating current	500mA
Flow rate	100mL/min
Life span	40 hours intermittent use
Max weight	104 grams

Table 6.11 – Specification for Peristaltic Pump

The peristaltic pump we have chosen has been the optimal type of pump for our hydroponics system which requires the transfer of small but precise amounts of liquid from the outside of the device enclosure to the interior of the water reservoir. We need to use two peristaltic pumps: one for pH buffer and other for adding nutrients to the solution. Thus, varying amounts of nutrient solution and pH buffer solution can be added to the reservoir independently of each other.

This pump will be installed in the device enclosure, next to the other pumps as well as the microcontroller and sensor printed circuit board. Silicone tubing will run from below the device enclosure where the containers of nutrient solution and pH buffer are located, up through the enclosure and out into the water reservoir by the various sensors. The tubing will need to go deep enough to make sure it will not affect the sensors in the water reservoir.

6.3.12 – Air Pump

One of the most important tasks for our hydroponics system is to oxygenate the water in the reservoir that the plants are absorbing the nutrients from. During the research portion of this project, we found out that oxygen must be present in the water for the plants to grow properly and prevent them from rotting. This holds especially true for the type of hydroponic that we have implemented which is the Nutrient Film Technique. In this system, the roots sit inside a closed space which means it may lack oxygen and thus it is important to add oxygen to the water solution which they will be absorbing.

The air pump design that we will be using for our system will be the diaphragm pump which pumps gas in a manner like the way that lungs operate. These types of pump work with DC motors which make them easy to interface with microcontrollers and which will be light on our budget.

The main objective of this sub system is to: be able to pump air continuously into the water reservoir, use a low power DC motor to power the pump with the capability of running at all times, and use a filter to oxygenate the water as air is pumped through it. Table xx shows the specification of the diaphragm pump subsystem.

Specification	**Value**
Medium	Air
Motor	AC
Max. operating voltage	120V
Max. operating current	80mA
Max. Flow rate	15 L/min
Life span	3-6 months continuous use
Max weight	3.1lbs

Table 6.12 – Specification for Air Pump

Since the air pump will be running continuously, this sub system will have the most power consumption. With the air pump that does the work of pumping the air into the higher pressure of the reservoir, the air filter/air stone will help diffuse the air into the water while filtering out solids contained in the water reservoir. A Nylon tubing will be required for this subsystem which will start from the enclosure and continue down to the bottom of the water reservoir, where the air stone and air filter are mounted.

The diaphragm pump will be mounted next to the other pumps like the peristaltic pump or water pump. A plastic nylon/silicone tube will run alongside the peristaltic pump tubes and will be routed into the water reservoir where the air filter and airstone are located. The air filter and airstone take this air incoming and squeeze water into the porous substance, which causes dissolved air to permeate the water flowing through the filter. The air stone and air filter will be mounted to the center of the reservoir as possible.

It will be better to mount these devices to protect from any water damaging the system. It will be mounted using screws and holes and will be placed in an environmentally protective case.

6.3.13 – Water Pump

For a hydroponic system, it's important for the water solution to circulate through the system for the plants to grow properly. It is important for a Nutrient Film technique hydroponic system to have a water pump with adequate water output and head pressure. The water pump will be connected to the power outlet, ideally to a relay-controlled power strip which can be controlled by the microcontroller. There are multiple options for pumps that can be utilized in a hydroponic system.

Most hydroponics systems incorporate a submersible pump of some size and strength depending on the water flow needs. The pump is utilized to deliver the water-nutrient solution from the main reservoir and up to the growing chamber and root zones. Submersible pumps are available on the market in a wide range of strengths at low cost. The submersible pumps are simply constructed from mainly an impeller and an electromagnet to spin it and create suction.

The strength of the pump for a hydroponic system is determined based on each individual structure's needs. Some things that are considered when determining the strength of a pump are flow rate, system volume, head height etc. Strength of a pump is determined in gallons per hour. A pump's GPH and cost are correlated. The minimum strength needed is determined by dividing the entire reservoir water volume by the time needed to circulate the entire system's volume of water. A good rule is to choose a pump that is twice the minimum GPH that a system needs.

Head height is another factor that is important when determining the pump strength. Head height is the vertical distance between the water line in the reservoir and the system's delivery outlet. Most manufacturers include a head height that describes what the GPH rates are for the pump at several different head heights. It is important to achieve the correct flow of water depending on the system.

NFT systems take into factors such as water volume, head height and the angle of the growth channels. Depending on the angle of the growth channel, the pump needs to be able to keep up with how fast the water is flowing down. Typically, NFT pumps are weaker and cheaper because the GPH and the head height are kept lower than other systems. Table 6.13 shows the specification of the water pump system.

Specification	Value
Medium	Water
Motor	DC
Max. operating voltage	6V
Max. operating current	250-300mA
Max. Flow rate	150mL/min
Life span	3-6 months continuous use
Max weight	0.33lbs

Table 6.13 – Specification for Water Pump

6.3.14 – Lighting

For our project, we needed a light source for the plants to grow in. Since our design is on the elongated side, an elongated grow light to match would be a perfect fit. That said, we have chosen the Monios T8 Plant Grow Light Strips. These strips are perfect in size to accommodate our design. They are 2 feet in length and they can come in packs of 6. This will give us more coverage and more flexibility when building our hydroponic system. These can also be daisy chained with either cords or directly so that one plug is used for multiple grow lights. Full specifications are given below in Table 6.14.

Size	24 x 1.29 x 1.38 inch
Power	24 W
Voltage	AC85 ~ 265V
Frequency	50/60 HZ M
Material	PC + aluminum

Table 6.14 – Product Specifications

This grow light has full spectrum sunlight replacement ranging from 420-470 nm to 610-680 nm. This means that our plants will be receiving the full color range of visible light. The light is also around a 2900K color temperature so for a more comfortable looking environment. This grow light also has a V-shaped design that allows it to have 2 rows of LEDs as opposed to one. In turn, there is more coverage and the light is more uniform across the surface for the plants. A sample image is shown below in Figure 6.9 of what the product looks like.

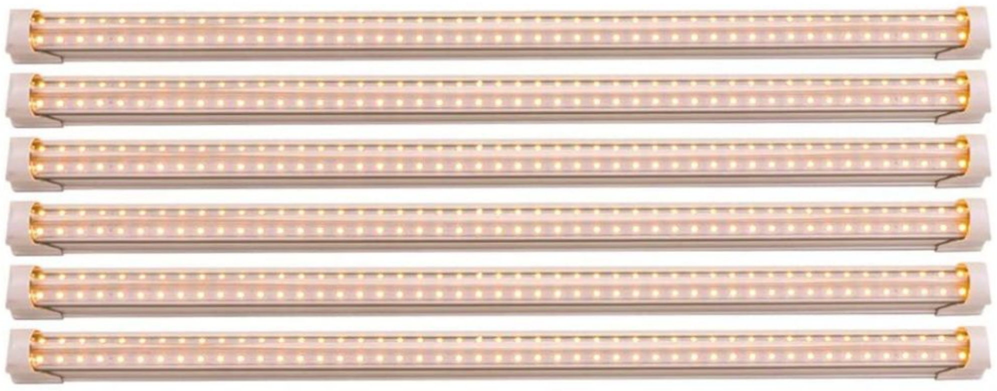


Figure 6.9 – Product Image

6.3.15 – Wi-Fi Module

In order to have our application connect to our hydroponic system, a Wi-Fi module is going to be needed. This is meant to connect with our microcontroller and can transmit data across Wi-Fi. For our project, we decided to go with the Adafruit ESP8266 microcontroller. There are plenty of features in this microcontroller that can be utilized in our project to give our users the best experience.

First off, it is an 80 MHz microcontroller with its processor coming from Espressif. It has full front-end capabilities as a client and access point, which will help connect our hydroponic system. It also has support for TCP/IP with DNS. That said, this microcontroller is fully capable of 802.11b/g/n connectivity and Wi-Fi 2.4 GHz connection with WPA/WPA2 support. Communication with the main microcontroller can be done via SPI, UART, or I2C communication. There are many pins on this microcontroller and the most important ones are the transmit (Tx), receive (Rx), ground (GRD) and power (V+/VBat) pins. These pins along with other pins can be seen in Figure 6.10. The transmit and receive pins are what is going to be connected to the main microcontroller to update our interaction with our database. This generation of microcontroller is also breadboard friendly as it now features a 500mA 3.3V regulator. It’s also CE and FCC emitter certified.

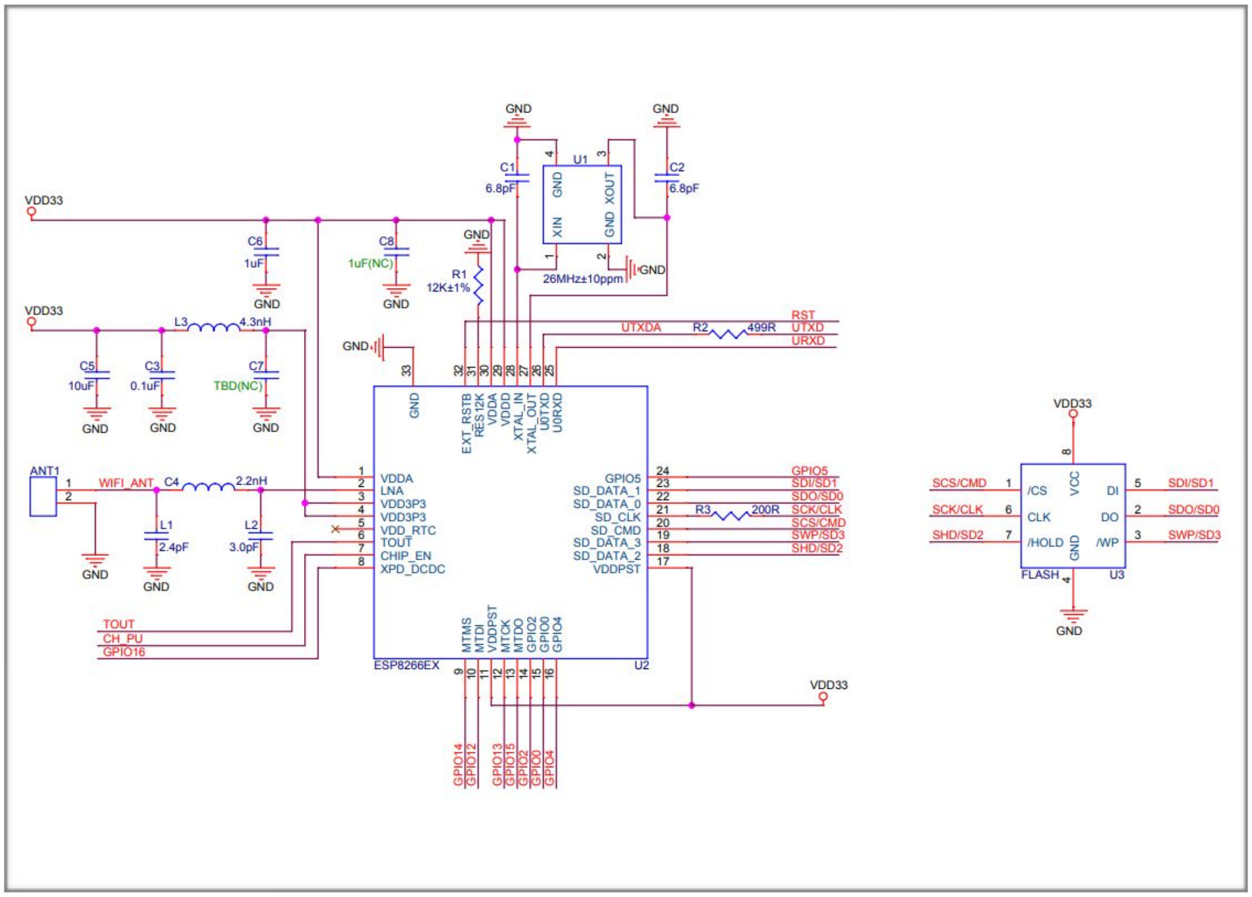


Figure 6.10 - Wi-Fi Module Schematic

6.3.16 – Device enclosure

The enclosure will be used as a protective structure to mount the printed circuit board, motor components and other electrical components inside. It will serve to isolate the sensitive electrical components from the environment, and it will bring structure and group the different design components to make the system look more professional.

The main objective of this enclosure will be: to house and protect all the electrical components by keeping all the sensitive circuits out of reach of water splashes that might damage them, be portable enough to be able to move, be made of transparent material so that the interior components can be seen during prototype demonstration, and made up of inexpensive though durable material.

The main function will be to house all the electrical components and to keep them safe and secure from outside interference like shock or weathering, and provide an appealing looking mounting system for the device. The enclosure will have to be transparent for demonstration purposes.

6.3.17 – Major Component Parts List

Component Type	Chosen Component	Brand/ Manufacturer	Notes	Cost
Microcontroller	Atmega 2560	Atmel	8-Bit	$5-$10
TDS Sensor	Gravity Analog TDS Sensor	DFRobot	Module & Probe	$15
pH Sensor	Liquid PH Value Detection Regulator and BNC PH Electrode Probe	Voktta	Module & Probe	$10 - $15
Temperature and Humidity Sensor	SMAKN DHT22	SMAKN + Adafruit	Measure both temp. and humidity. Based on the Adafruit DHT22 sensor, implements PCB module for simple interfacing for microcontrollers.	$10
Light Sensor	TSL2591	Adafruit	All in one Sensor Module	$10
Water Temperature Sensor	DS18B20	TZT teng	Module & waterproof probe	$2-$5
Water Level Sensor	Contact Water/Liquid Level Sensor	CQRobot	Photoelectric, Module & Probe Design. Contact with the probe and water is safe.	$15-$20
Distance Sensor	HC-SR04	HWA YEH	Ultrasonic Sensor	$1-$5
Dev. Board	Mega2560 R3	Elegoo	Same as a standard Arduino Mega, but a cheaper non ‘name brand’ version.	$15-$20
LCD	7” TFT LCD Display	EastRising	LCD display & module	$40
Wi-Fi Module	ESP8266	Espressif	Wi-Fi Module	$10
Air Pump	Hydrofarm Active Aqua Air pump	Hydrofarm	Air pump	$30
Liquid Pumps (For Dosage purposes)	Peristaltic Pump	N/A	Need 3, One for nutrient and two for pH	$10-$25
Water Pump	Standard Aquarium/Pond Water Pumps	N/A	Minimum 1, maybe more if more power needed	$15-$50
Lighting	T8 LED	Monios-L	Multiple LED beams and daisy chain support	$65

Table 6.15 - Major Component Parts List

6.4 – Power Subsystem

This section will talk generally about the power design for our hydroponic system. It will require many components in order to ensure the proper power distribution throughout the system. Each component that is electrically connected will need a 120V, 60Hz receptacle to be plugged into. This will be accomplished by using a power strip that will extend any nearby receptacle that the user will need. For the systems that can be plugged directly into the power strip, such as lights or water pump, internal AC to DC converters won't be required since there are no special plugs needed.

In facilities and homes around the United States, a receptacle featuring one hot, one neutral, and one ground terminal is considered typical. Most of these receptacles are rated 120V/15A or 120V/20A and operate at around 60Hz. These ratings will be referred to as "receptacle" doing forward. One consideration that will be encouraged is that they use ground fault circuit interrupter receptacles in order to prevent any encounters with water. The way it works is that it detects when it detects an imbalance in current flow between the receptacle and whatever it is connected to, the breaker will trip in order to prevent whatever is taking the current away from the circuit.

The PCB board will require a special type of jack to be plugged in to receive power from the receptacle. It is desired that it is outfitted with a 2.1-millimeter female barrel plug that has its connection by a 9V to 12V DC power adapter. Many sensors will be attached to this PCB. Most sensors require small voltages in a typical range of 3.3-5V which is simply received straight from the PCB. There will be few components that will be needed to make up the power subsystem. This section talks about power supply, ac-to-dc, regulators, and batteries.

6.4.1 – Power Supply

Power supply delivers electric power to an electrical load. The primary function of a power supply is to convert electric current from a source to the correct voltage, current, and frequency to power the load. The circuit components were all evaluated for power consumption utilized. Once the power needs were calculated the power supply was ready to be considered. There are several components that make up power supply. The goal of a power supply is to take the AC from the wall outlet and convert it to DC and reduce the voltage using an input power transformer. The components used are: Transformer, Diodes, Voltage regulators, Capacitors.

The first consideration was the transformer. The transformer will be required to deliver around 1 amp of current for the component needs. Depending on the primary winding and secondary winding, a step-down transformer can be created to get the required Voltage for all the components. After some calculations, the max voltage the circuit needs to operate is about 12-Volt DC.

The input signal is still a full sign wave. Once the signal comes through the next step is to rectify the signal. A full bridge rectifier circuit can be used. This removes the negative half of the sine wave. 4 1N4007 diodes can be used for this purpose. Once the voltage has been rectified, the fluctuations in the waveform need to be removed with the use of capacitors. The capacitor stores energy on the rising edge and expends the energy during the voltage drop. This significantly reduces the amount of voltage drop and smooths out the voltage. Increasing the storage capacity of the capacitor generally produces a higher quality power supply. Once the voltage is rectified, there is still some variation in output called ripple. The voltage is then passed through a regulator to create a fixed DC output with less ripple.

6.4.2 – Voltage Regulation

This project requires a couple of voltage drops in order to operate properly. First, a 12V to 5V drop is required and a 5V to 3.3V. There are generally two types of voltage regulators. This section will talk about how they operate and how they achieve voltage regulation. Linear regulators tend to be a little cheaper to implement but aren’t as efficient.

Linear Regulators:

One way to think of linear regulators is as an active series resistor. It works by varying its effective resistance so that the output voltage can remain the same. The pros for this design are that it is cheap, provides a clean output, and it's simple to implement. The downside is that it dissipates a relatively large amount of power. The way to think about the power dissipation is to think of a linear regulator as a series resistor. If 9V goes in and 5V comes out, there is a 4V drop. Using the power equation P=VI, the power dissipated can be calculated. Most linear regulator chips require an input capacitor and an output capacitor.

Switching Regulators:

Understanding of switching regulators work on how transistors work and how to store energy in inductors and capacitors. In theory, when a transistor is functioning as a switch, it drops no voltage while it's on and blocks all current when it is off. If there is no voltage drop then there is no power wasted as heat. In practice, there is some power being wasted since there is a small voltage drop. Inductors store energy in a magnetic field when current can flow through them. Capacitors use their energy to keep voltage constant. When the transistor turns on, it charges up the coil. When the coil is fully charged, the transistor turns off. The output capacitor works with the inductor to keep the voltage constant.

6.5 – Software Design.

Technology has improved a lot over the years and has given us more possibilities when expanding a system with features and to incorporate the user. With advances in computer programming and software, we are able to create products that can lighten the load on a user. This is something we have been looking into as a group and want to incorporate an Android application to our hydroponics system.

Nowadays, everyone has a mobile device and there are plenty of software on the market that have hardware specific coding requirements to program in different mobile platforms. This can make it difficult for us developers as we would like to have our application to work on all platforms to reach the most amount of people on the market. However, most of the members on the team have an android cellphone and cross-platform functionality isn't as important for our project prototype. Therefore, it would be in our best interest to develop for Android.

With that being said, we will be developing an Android application for our hydroponic system. Since we also want to keep the cost as low as possible for this prototype development, we decided to go with Android Studio as our Android development environment. There are many great features about this IDE and it works with the team's hardware specification in Table 5-1. There is a large user base with loads of documentation for help. Countess forums are also a plus as many considerations can be presented to come to a solution for any problem you might have. Other features have been explained in detail pertaining to specific software features. Since we are developing an Android Studio, there are only 2 programming languages we can use to develop the application. These languages are Kotlin and Java. We will explore the advantages and disadvantages of each for faster development.

Operating System	Windows (7/8/10)	Mac OS (OS X 10.10 +)	Linux OS (GNOME or KDE)
Minimum RAM	3GB	3GB	3GB
Minimum Disk Space	2GB	2GB	2GB
Minimum JDK version	JDK 8	JDK 8	JDK 8
Minimum Screen Resolution	1280x800	1280x800	1280x800

Table 6.16 - System Requirements

Kotlin is a newer language that has been brought up from company to rival the Java programming language for Android development. It is a statically typed language that is based on the Java Virtual Machine (JVM). Therefore, it is completely compatible with Java technologies and features. Kotlin variables are non-nullable, meaning the variables and object can’t be assigned null values. There is a way to bypass this if there is really a need to assign or return a null value, but the added null pointer exception is an added bonus. Kotlin also provides extension functions. You can extend existing classes with new functions to handle more tasks. This is more efficient than Java as they would need to make a new class and inherit a different class to add features to it. There is also coroutines support in Kotlin, which is a feature that manages threads. There can be multiple threads running, even though Android is single threaded by default. Kotlin can suspend processes at different points without blocking threads when a huge program is being run with many operations. There are no checked exceptions in Kotlin. This means that there is no need to declare a try/catch/finally statement to catch exception in the system. While this can be a good thing to remove annoying programming errors and less verbose error handling, there is some benefit to having more options for error handling. Data classes are also handled better as certain work is done for the programmer. Getter, setter, and constructors are automatically handled at compilation so the programmers don’t have to write as much code.

Java is an object-oriented programming language. Java is not safe with null pointers, which can become an annoyance when developing. Since Java allows users to assign null values, errors are going to need to be handled in a program. Extending functions to classes is also a small inconvenience as it requires the process of inheritance to need more code to accomplish the same thing. The thread support on Java is not as easy to use as Kotlin. When the system is under high intensity of operations, certain threads will get blocked. While there is plenty of support from libraries to handle threads, it is harder to manage multiple threads and can become a complex task. Checked exceptions are also a feature that allows for more error handling by manually handling errors with try/catch/finally statements. However, more code needs to be written and there are more opportunities for error prone code. Adding to the more verbose coding style of Java, data classes are also a pain to write as it requires more code. Setters, getters and constructors all have to be manually inputted. This eats up development time, which isn’t a good thing.

With all that said, it’s safe to say that Java will be used to produce our android application. Most of the team members already know Java from our lectures and only one has some experience with Kotlin. In order to engage the team in all aspects of the project, we want the computer language to be as universal as possible.

To add with Android Studio and Java, we are selecting MongoDB as our database. This is going to hold all the user’s information on their accounts and information of various plants. There is the MongoDB Atlas Cloud Database that allows us to make an account in the cloud. There is also the MongoDB Stitch Serverless Platform to process data and access data with low amounts of code. On top of this, the MongoDB Cloud Manager is also available to manage our databases and backup any data that was lost. Real time charts also exist to give more perspectives on the data, which can help out in developing the application. These are some reasons for choosing MongoDB and why we will be using it to develop our Android application.

6.5.1 – User Interface and Features

In order to have a positive experience with products, it needs to be easy to use. This is exactly what we want our application to be and are gearing the user interface with a minimalist design. That said, there are 6 main panels or tabs to navigate the application in order to use the features of the app. We wanted to keep the number of panels, or main features, under 10 because it can feel too bloated with content at that point. While it may give more options, there is a diminishing return on usefulness and many consumers would like to have that tradeoff. Therefore, the 6 main panels and smaller panels are going to be explained below for their function and use to the user.

Login Screen

The very first panel to see when entering this application is the login screen. Of course, the user won't have an account at first and will need to create one as a first-time user. First draft UI implementations can be found below in Figure 5-1. Although the login panel is a standard panel for any application that uses any sort of introductory authentication, there are some desired features in this step of the application revolving security. These features are ones that could make our users data more secure and prevent damage to our Android application. Since this is going to be a native application with Android Studio, there are some added benefits to the security of the system. This is beneficial to our project as we don't want to have our user's information to be changed or stolen. If we were to expand or replace some features that have cross user interaction, security would be an important measure. This can also be a concern on large scale commercial systems with classified data that needs to be protected. With features like Android Application Sandbox that isolate data and code between different apps. There are also user-granted permissions to limit options to certain users by limiting their access to certain data and system features. There is also an encrypted file system to protect data on lost phones to protect their data. There also exists memory management features like ProPolice, NX and ASLR that minimize memory errors in the code. This helps with security by leaving no exposed data on the memory for others to retrieve, which further increases the remote security measures available.

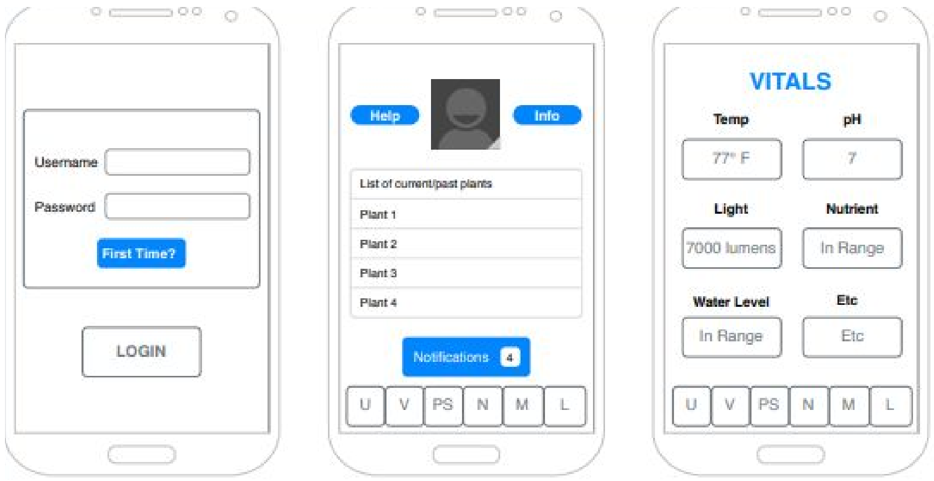


Figure 6.11 - (Left to Right) Login Screen UI, User Account UI, Plant Vitals

User Account

The next panel to appear after login has been successful is going to the user account panel. This is the tab “U” in Figure 6.11, which is shown above. This panel holds a few important buttons and handles personal information. There is going to an info button that holds common information like name and number. This button will also allow you to upload a picture and edit any of the inputted information on the user. There will also be a help button to explain proper use of the application with accompanying pictures to further guide the users. At the center of the panel, there will be general information on previous plants and currents plants that are growing. This is a quick to view information on your plant right from the account page. At the bottom of the panel, there will be a notification button for alerts on the system. These can be viewed through to be deleted and see to see if any errors are still occurring or if its been fixed. It’s important to note that these are mock drafts on user interface implementations that can change in the development stage of the project. If additional features are wanted when creating the application, then they’ll be implemented where we see fit.

Plant Vitals

Arguably the most important tab on the application in the plant vitals. This is the tab “V” on Figure 6.11 above and it holds the most recent information on the parameters that keep the plants thriving. The design of this display is pretty simple, as seen in the mock design above, and it is meant to be that way. This is to ease the most strain possible away from the user and have information readily displayed with little room for distractions. The parameters that our microcontroller will be monitoring will be recorded and displayed on this tab with the most recent information available. It’s important to note that the hydroponic system will be taking hourly measurements on all the sensors hooked up to the system. These recorded measurements will be able information used to update the vitals tab.

Plant Search

The next important tab in the "PS" tab in Figure 6.12 which is the plant search tab. This tab allows the user to scroll through the system's database of plants. The plants hold data on ideal parameters recommended for that plant. In the figure, these plants are represented in miniature cards that will hold a picture of the plant, followed by the name of the plant underneath. The reason for preset plants is for the user to select recommended conditions for a plant for optimal growth. These parameters are then sent to the microcontroller to update the system controls. This gives the user the power to grow a wide variety of plants as there will be quite a selection of common plants in the database. There will also be a new plant button to allow the user to create custom plants cards and add them to the app and database. This can benefit our users as they can add exotic plants that aren't in our database and can even experiment with different parameter values for plants we do have in our database. To make this process more intuitive, you can copy parameters of a plant from the selected card and edit any values to create a new plant card. There is also a search feature to quickly jump to the desired plant the user wants to view.

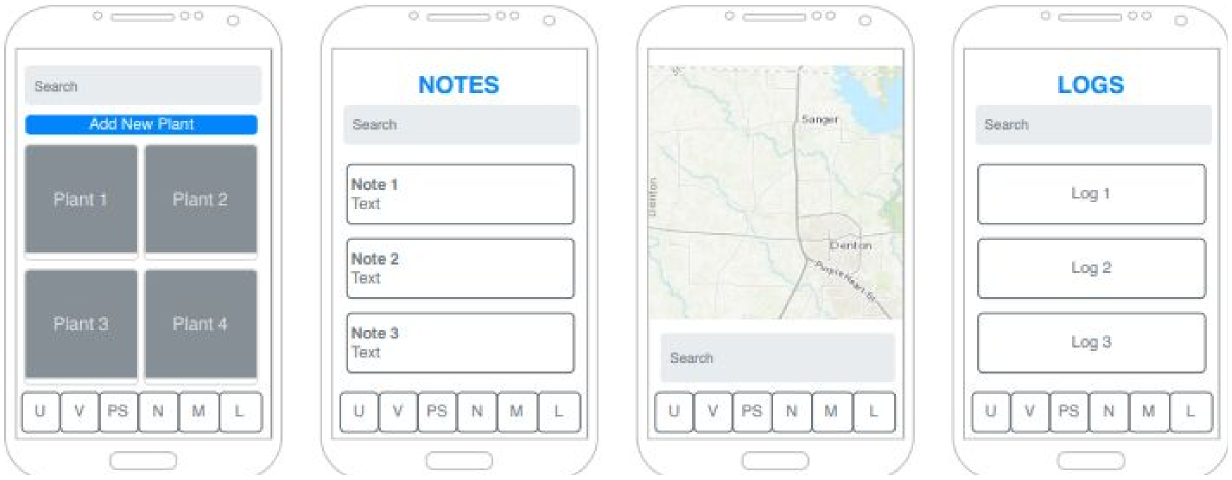


Figure 6.12 - (Left to Right) Plant Selection, Notes, Maps, Logs

Notes

The next tab is the notes tab and it is shown in Figure 6.12. This is an everyday notes tab that a user may have on the growing process of the plants and jot down some items on materials to pick up. We find that having a notes tab is beneficial in holding common information on plant progress and is a fast delivery method to use. In order to maintain some order in the notes, there will be a search feature that will match the searched string with the closest matched title of the notes. This ensures fast delivery and a small wait time in receiving information. There are also plans of incorporating a tag feature on each of the saved notes. The tags will be able to be sorted through a filter button that will be on this panel. This ensures more organization and will even be color coded into categories for easy application.

Maps

The second to last tab is the maps tab and is denoted as “M” in Figure 6.12. The map feature on this tab will most likely be Google Maps. It has its own Google API that is loaded with features and is compatible with many languages and devices. This Google API is also very popular in the development community and is constantly being updated for new features and security measures. Since it is used by many developers and has a large following base, there are many tools developed with the google maps and accessing features within the Google Maps layout. There is also a search function with this tab to locate certain items on the map. It will be able to locate stores that sell hydroponic parts in the area for replacements parts if needed. It can also locate hydroponic services to repair any functions or installations in the hydroponic system if any damage to the system were to occur.

Logs

The last tab on the application is the log tab and it is demonstrated in Figure 6.12. We want to give our guests the ability to see everything going on with the hydroponic system. Since the sensors hooked up to our hydroponic system are going to be active all day, it is going to be recording information on the system. This system will be sampled every hour and sent to the database to then reach the app. The app is planned to display the logged information into small cards. These small cards will contain the vitals that have been recorded every hour. The user can then search for these logged cards with a search bar at the top. The logs will be stored based on time, so there will be an implementation to look up and range the dates. A useful feature to note is that the user can also save any of the old logs to make a new card in the plant selection tab. This eliminates the need to memorize information and makes navigating the app faster.

6.5.2 – User Interaction with Database

In this section we are going to highlight user interactions with the database. When the user logs in, the inputted username and password will be sent to the database to check for authorized users in the system. If an account hasn’t been made yet, then and will have to be made to put you in the database. There can be multiple accounts with one hydroponic system, so it is important to keep those separate. You can either gain entrance to the user’s account page or you are denied access to the app. When entrance is denied the user can try to login again and the same process to gain entry will occur. This can be seen in Figure 6.13. When login is successful, the user gains access to all app features.

When the hydroponic system is running it is constantly reading in values for every sensor monitoring the system. In order to keep the user updated on plant progress, the values are going to be sampled at once an hour. That way the information is still relevant, useful and won't feel outdated. When sampled, the sensor data will be transferred to our database via Wi-Fi module. The database will be updated with new sensor data once an hour to match our system sampling. With our app, the user is able to modify plant data that is in our database. This is for custom plant configurations to yield different results in plant growth. When a user changes any information on the plant score cards, the update will be reflected within the database. The same goes for when the user adds a new plant scorecard.

The user is also capable of changing plant parameters in the hydroponic system. This is for when a different plant is going to be planted or if the user wants to change any parameters on the current plant they are growing. When the user changes plant vitals on the app, it needs to be sent to the database. It will then get to the microcontroller via Wi-Fi module and update system parameters. When new changes are made, values in the log will change to the set parameters.

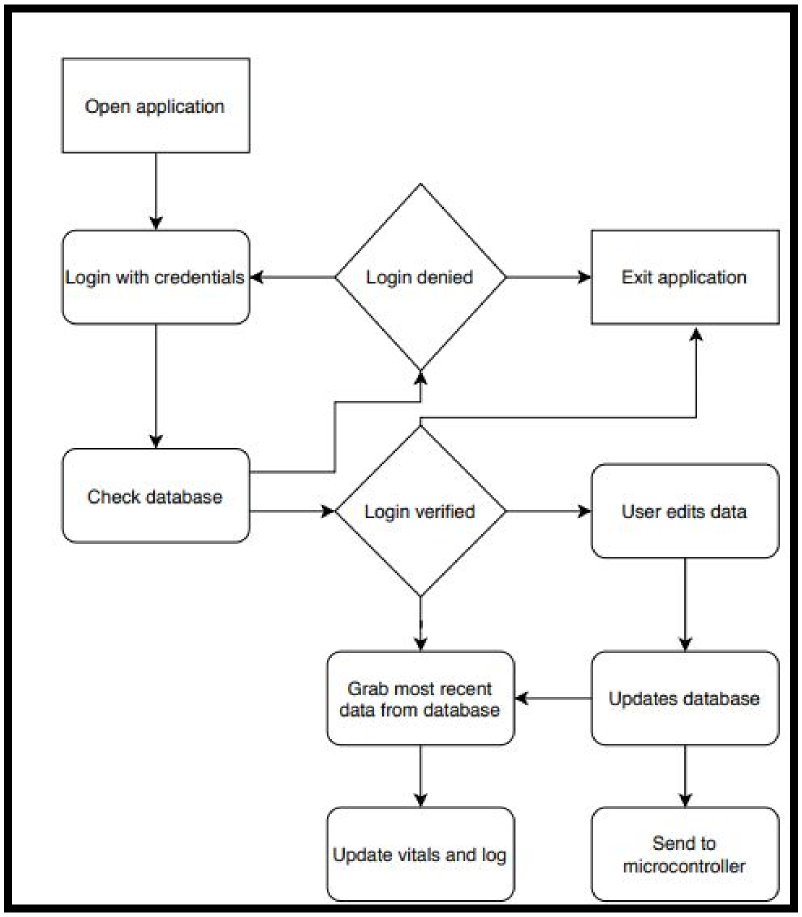


Figure 6.13 - Application Design Flowchart

6.5.3 – Embedded Software Design

In order for our hydroponic system to be automated, a microcontroller is going to be used to monitor and update system parameters. There are going to be many sensors attached to the microcontroller where each sensor is responsible for recording a plant vital. The main way the system is going to work is through polling, as it will check all system parameters in a loop. This makes sure that every sensor is checked and that the most recent information is reported.

At first, the user will boot up the microcontroller and start up the system. Shortly after doing so, the microcontroller will check each sensor's status to make sure they are operating perfectly. If they are not, their status will be sent to the LCD to alert the user of any issues. If all sensors and modules pass, initial recorded values will update the system. This means that present recorded values are going to show up on the LCD and be sent to the user via our app. The user has the option to just let the hydroponic system be on and not update to a given value. If this option is chosen, then the system will just report sensor data and not change system parameters. When growing plants, its vitals are going to be checked to see if it is within desired ranges. If they are not, our microcontroller will either adjust or alert the user. Full details on how the embedded logic will work are below in Figure 6.14.

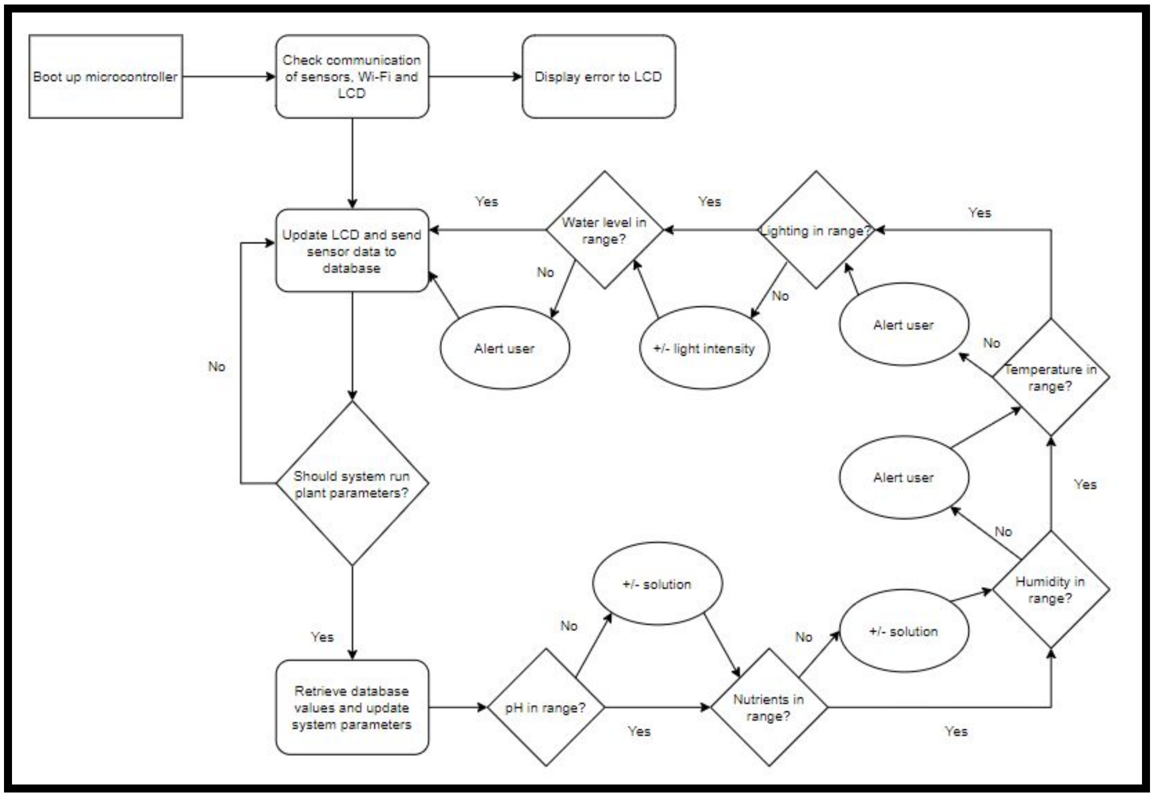


Figure 6.14 - Embedded Logic Design

6.6 – Physical Design

In order to achieve one of our goals, which is to have a medium sized system, we decided to make our hydroponic system about 4 x 3 x 5 feet. This so that the object is still able to fit through the standard American front door and be placed inside a home. The size also accommodates a good amount of plants that could be placed. We believe this maximizes the plants to size ratio. Our hydroponic system is also going to be built with wood and PVC pipes as its backbone. This provides good support and fits within our spending budget.

As mentioned before, our hydroponic system is going to model after the nutrient film technique. This means that the growing channels are going to be tilted by a small angle to allow the water to form one side to another. While this is hard to represent in Figure 6.15, a tilt will be in place with the end that is connected to the pump being higher. The water will be at the bottom of the system in a reservoir or container. In Figure 6.15 it is illustrated as a box under the plants. There is a cut out to show the inside, but there will be a covering in the prototype. There is going to be a pump and a tube that will push the water upwards into the growing channels. This is illustrated as dashed lines in Figure 6.15 along with the drainage tube to flow water back down. This completes the basics of the nutrient film technique.

Other elements include the lighting panel that is represented as a board held up by a pole. The lighting will be attached under the board and will provide light intensity to the plants. The pole will be movable so that different light intensities can be achieved while the plant is growing. There is also going to be an LCD attached to the project that is represented as a rectangle on the side of the hydroponic system in Figure 6.15. This is to display plant data and alerts to the user. There are other modules and sensors that aren’t shown in Figure 6.15, but they will be placed around the system accordingly to make best use of the space.

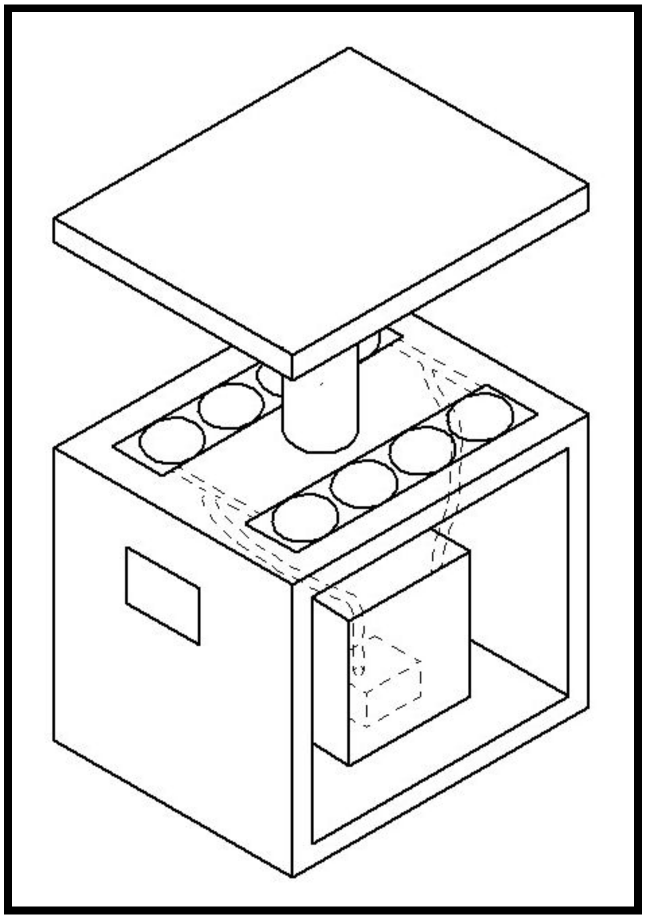


Figure 6.15 - Sample Hydroponic System

7.0 – Project Prototype Testing and Integration

In order to create a project that is both reliable and durable, various kinds of test methods will be mentioned in the subsections below in order to prototype and test hardware components, circuit designs, schematic designs, software components, and user interfaces. Depending on the portion being tested, the subsection may include testing methods, simulations, and/or expected readings to compare to in order to ensure equipment is working correctly.

The hardware and software environments are mentioned in their respective sections in order to have a test environment configuration that mimics the products environment as much as possible to identify any environment or configuration related issues before the final design.

7.1 – Hardware Test Environment

All hardware tests should be done in an indoor location that is temperature controlled in order to match the environment that our hydroponics system will be in most of the time. Should in person classes be reinstated by the Fall 2020 semester, most hardware testing should be performed on campus at the University of Central Florida. The two rooms best suited for testing and integrating hardware components are the Texas Instruments Innovation lab located in the Engineering atrium and the Senior Design Lab located on the fourth floor of the Engineering One building.

The Texas Instruments Innovation Lab has four electronic lab benches that feature oscilloscopes, function generators, multimeter, and three outlet power supplies. These instruments will be useful in verifying that our sensors are reading the correct data, in power components until our final product power supply is ready to use, and much more. The Senior Design lab features a similar electronic lab bench set up. Along with the electronic lab benches, the Innovation lab features a 3D printer and laser cutter that may be useful for prototyping hardware designs that will be used in conjunction with our electronic components.

The final important feature is available in both the Innovation Lab and the Senior Design Lab, which is soldering stations. The solder stations also have available microscopes for soldering surface-mount devices. The Innovation Lab is available for use during most of the day on weekdays with staffed student assistants and the Lab Director Don Harper ready to help on a rotating basis. The Senior Design Lab is available 24/7 but has less resources available unfortunately. Both areas feature security that requires a student ID card to gain access to the lab, which should provide security in case a component is forgotten or sabotaged in any way. The Senior Design Lab also features lockers that will allow us to store work overnight should we have too much equipment to easily move around.

Our hardware design is based around the ATmega 2560 Microcontroller and for the purpose of testing we will be using a development board that provides easy access to the pins of the microcontroller. Through this testing we can identify what pins we will be using on the microcontroller and design our printed circuit board around that. Many of the chosen sensors were made to have simple compatibility with Arduino software, which we may be using to run tests with the development board.

7.2 – Hardware Specific Testing

The following section will describe in detail the testing and debug tools used on each specific component. Furthermore, the individual test scenarios used on each component will be detailed and made to reflect what the component will be responsible for in the final hydroponic system.

7.2.1 – Sensor Testing

The following subsections will speak to the testing of specific sensors found in the hydroponics system. Much of the sensor's limits and calibration comes from the documentation specific to each sensor and is reiterated to specify the important information that is needed in order to properly test our system by system before full assembly and integration.

7.2.1.1 – Water Temperature Sensor

The DS18B20 digital thermometer includes it's own probe that is able to measure temperatures from -55 degrees Celsius to 125 degrees Celsius with a +/- 0.5 Celsius accuracy when between the temperatures of -10 and 85 Celsius (our expected environment for the sensor's use is within this range). The resolution of the DS18B20 digital thermometer is set to 0.625 Celsius by default, but can be modified by the user. To test the temperature sensor, the DS18B20 digital thermometer will be connected to the development boards pins and will be interfaced through an Arduino IDE to see if the readings match our expected results. Through the IDE we should be able to read the results of the thermometer through the 1-Wire data line that provides "Direct-to-Digital" results. First, the digital thermometer will be exposed to ice water in order to test the lower limit of our expected environment, the outputted results will be compared to those of an analog thermometer. Next, in a similar fashion, water will be heated on a stove to slightly below boiling and the results of the digital thermometer will be compared to those of an analog thermometer. Should results of both of these fall within a reasonable error tolerance (for example two percent), then the digital thermometer shall be deemed usable for final design purposes.

Test Input	Expected Value	Acceptable Value
Boiled Water	≈ 100° Celsius	±1° Celsius of analog reading
Iced Water	≈ 0° Celsius	±1° Celsius of analog reading
Room Temperature Water	≈ 20° Celsius	±1° Celsius of analog reading

Table 7.1 - Test Ranges for Water Temperature

7.2.1.2 – Ambient Temperature and Humidity Sensor

The SMAKN DHT22 (also called AM2302) digital sensor collects temperature and humidity levels. The operating range of the DHT22 falls between 0 and 100 percent RH for humidity and -40 to 80 Celsius for humidity. These limits should work fine for our design as they cover the total of possibilities for our project's typical environment. The humidity portion of the sensor shall be tested by testing in an isolated room in the presence of a humidifier. By changing the local humidity with the humidifier, the sensor should pick up the changes and report the data through the data line. Similarly, the temperature portion shall be tested by exposing the sensor to an icebox to check the lower limit and a heat source to check the upper limit. Because the sensor has been designed for use with Arduinos, connecting the sensor to our development board and obtaining data should be simple. The DHT22 features a single-bus digital output and two pins to provide 2.2-6 V DC.

Test Input	Expected Value	Acceptable Value
40 RH	≈ 40 RH	±1 RH of analog Psychrometer
50 RH	≈ 50 RH	±1 RH of analog Psychrometer
60 RH	≈ 60 RH	±1 RH of analog Psychrometer

Table 7.2 - Test Ranges for Humidity

7.2.1.3 – Light Sensor

The TSL2591 High Dynamic Range Digital Light Sensor is a precise luminosity sensor that allows for exact lux calculations in the light range of 188 uLux up to 88,000 Lux. The indoor conditions experienced by the hydroponics system should fall easily within this range. By combining a broadband photodiode (which captures visible and infrared light) with an infrared-responding photodiode it is able to have the most accurate response possible. In order to test the response of the luminosity sensor, the TSL2591 will be exposed to a dark room and return a minimal amount of Lux through its data bus, then exposed to differing amounts of light and perform accordingly. The TSL2591 communicates with the microcontroller through a standard two-wire I2C serial bus and shall have effectively no adverse effects from noise due to being digital.

An extra feature to be tested is how the TSL2591 luminosity sensor responds to sudden changes in light intensity. According to the datasheet for the sensor, the TSL2591 supports an interrupt feature that detects meaningful change in light intensity. This meaningful change in light intensity can be detected by the user, for instance the difference between a room with all the lights off and the main light on. This feature is important to test in case we implement some sort of sleep mode in which the hydroponics systems LED display may turn off, since this is an indoor design and during the night should minimize it's appearance if possible. The system would then wake up it's display when the user awakes and turns on the light to the room.

Test Input	Expected Value	Acceptable Value
Direct Sunlight	32,000 to 100,000 lux	±1° Celsius of analog reading
Normal Indoors	401 to 1,000 lux	±1° Celsius of analog reading
Dark Indoors	51 to 200 lux	±1° Celsius of analog reading

Table 7.3 - Test Ranges for Light Sensor

7.2.1.4 – Water Level Sensor

The CQRobot Contact Water/Liquid Level sensor is a photoelectric water liquid level sensor that operates by using optical principles. There are no mechanical parts involved in the design so luckily little calibration is required with its design. The live parts of the sensor are located far from the controlled liquids it measures so it should not pose a serious safety threat during testing and in the final design. The water/liquid level sensor works between -25 and 105 degrees Celsius, which is well within the environment the hydroponic system will be in. According to the specifications, there is no limit on its measuring range, but for the sake of our testing we will use a full container as the upper limit to test and an empty container as the lower limit to test. Testing will be done by using the sensor on our water and nutrient containers at differing levels of liquid. The results of the water/liquid level sensor will be compared to manual measurements done with a ruler. Should the readings from the sensor closely match our manual readings, we will deem the CQRobot Contact Water/Liquid Level sensor as ready to be used in the final design.

Test Input	Expected Value	Acceptable Value
Empty Container	0 inches of water	±.2 inches of water
Half-filled Container	6 inches of water	±.2 inches of water
Full Container	12 inches of water	±.2 inches of water

Table 7.4 - Test Ranges for Water Level Tests

7.2.1.5 – Total Dissolved Solids Sensor

The DFRobot Gravity Analog TDS Total Dissolved Solids Sensor detects the cleanliness of water in Parts Per Million (PPM). The measurement range of the sensor is between 0 and 1000 ppm, which should cover all of our expected usage cases. The live parts of the sensor are located far from the controlled liquids it measures so it should not pose a serious safety threat during testing and in the final design. Unlike most of the other chosen sensors, for cost reasons we chose to use an analog TDS sensor and thus will have a slightly harder time testing compared to other sensors in this section.

In order to test the DFRobot TDS sensor for accuracy, we shall use it on a variety of liquid substances with different typical TDS values. To check the lower limit, the sensor shall be used to analyze drinking water, which typically falls in a Parts per Million of 0 to 50 PPM. Next, we shall test the TDS sensor on hard water, which has a typical PPM value of approximately 170. The DFRobot TDS sensor shall then be used on tap water, which has an average PPM of 170 to 320. Finally, in order to check the upper limit we shall continually add extra contaminants to tap water and this change should be reflected in the TDS sensors readings. Should all of the TDS sensor readings fall within the expected PPM readings as listed above, then the TDS sensor shall be deemed viable for use in our project.

Test Input	Expected Value	Acceptable Value
Drinking Water	0 to 50 Parts per Million	0 to 50 Parts per Million
Hard Water	170 to 320 Parts per Million	170 to 320 Parts per Million
Tap Water with Increasing contaminants	300+ Parts per Million	300+ Parts per Million

Table 7.5 - Test Ranges for Total Dissolved Sensor Tests

7.2.1.6 – pH Sensor

The liquid pH Value Detection Regulator Sensor Module is used to measure the acidic and basic potentials of a solution. The pH scale ranges from 0 to 14, with 0 being the most acidic solution and 14 being the most alkaline. A pH reading of 7 is considered neutral. To ensure we have the most accurate reading of pH levels possible, we must calibrate the sensor before using it. Calibration is accomplished with the use of liquid substances called “buffers”. These buffers have set pH values and by measuring them with the pH sensor we should be able to identify if the sensor works properly.

There is no set optimal range for the pH levels of crops, and it varies widely from crop to crop. In the research into the topic, usual pH levels range from as low as 4.0 to as the lowest and 8.0 as the highest. Because of this we want to make sure that the pH sensor is as precise as possible to avoid plant decay due to inappropriate pH levels. Besides the buffers mentioned above, there are hydroponic “pH up” and “pH down” solutions that help modify a liquid's pH levels. For testing purposes these will be added to a neutral base and as more liquid is added, it shall be reflected in the readings of the pH sensor. When it comes to how the system will actually work, these pH up and down solutions will be used to correct the hydroponic systems plant environment when the pH level is detected to be out of range. Because of this, it is important to use these solutions in our sensor tests. Besides pH is a product of inorganic and organic matter, we also must drain and refill the water periodically as debris that falls in the water can have a major impact on the solution's pH levels. The readings on all the tests mentioned above shall be compared to a manual reading of the pH levels with pH test strips and should the sensor readings all seem correct, the liquid pH Value Detection Regulator Sensor Module shall be deemed ready for the final project design.

Test Input	Expected Value	Acceptable Value
Vinegar	3.0 pH	3.0 pH
Wine	2.8-3.8 pH	2.8-3.8 pH
Beer	4-5 pH	4-5 pH
Milk	6.3-6.6 pH	6.3-6.6 pH

Table 7.6 - Test Ranges for pH Level Tests

7.2.1.7 – Distance Sensor

The Ultrasonic ranging module HC - SR04 provides 2 centimeters to 4 meters non-contact measurement function with an accuracy of 3mm. The testing of the Ultrasonic Distance sensor shall be fairly simple: the sensor shall be placed on a flat surface with a large object in front of it. The tester shall vary the distance between the ultrasonic distance sensor and the large object periodically and compare the recorded result with a manual tape measure reading. The test shall be repeated with a smaller object next to ensure the sensor is able to measure smaller objects distances as well. If the results of the distance sensor seem to match up with the manual tape measure readings, then ultrasonic distance sensor shall be deemed viable for use in our project.

Test Input	Expected Value	Acceptable Value
10 cm	10 cm	±2 cm
20 cm	20 cm	±2 cm
40 cm	40 cm	±2 cm
60 cm	60 cm	±2 cm

Table 7.7 - Test Ranges for Distance Level Tests

7.2.2 – Other Hardware Testing

Besides just the sensors of the hydroponics system, other subsystems such as the water pump and LED display need to be tested before putting together the final design. The following subsections outline how each of these subsystems will be tested and ensured to work.

7.2.2.1 – Water Pump Testing

The water pump in our system will need to be tested to ensure that it works properly and reliably. The pumps in our hydroponics system will be peristaltic pumps (commonly known as roller pumps), which are used to displace liquids. Unlike the aforementioned sensors, this subsystem will not be connected to the microcontroller, but instead to some kind of power source that will detect whether the microcontroller says to supply power or not, with this the water pump can be “controlled”. The water pump will be proven effective and capable of use in the final design if it can reliably move the specified amount of liquid per pump from the pump location to the final location, then the water pump that was tested shall be considered reliable enough for use in the final design of the hydroponic system.

7.2.2.2 – LCD Display Testing

The LCD display of our hydroponics system will be tested using either an Arduino programming environment or using a Raspberry Pi, depending on how we choose to control the display in the final design. To ensure that the LCD display is able to obtain information from the microcontroller, we will code the display to print “The quick brown fox jumps over the lazy dog”, which includes every letter of the English alphabet, along with the number one through nine to ensure we are able to transfer numerical data as well.

7.2.2.3 – Logic Level Shifter Testing

Logic level converters are used to step-up and step-down voltage in order for devices to be compatible with different voltages. For instance if a 3.3 Volt device is connected to a 5 Volt system, then it will not work as expected. In our hydroponics system so many subsystems will be connected to our microcontroller so it is important that they are able to interface correctly regardless of what voltage they use.

In order to test our logic level converter we will need at least voltage sources to power our logic level converter board. This can be accomplished by using the three-channel power supply available in the Senior Design Lab and Texas Instruments Innovation Lab. One of these voltages will be the high voltage and the other will be the low voltage. Once the voltage sources are connected to the logic level shifter, we will use the function generator to generate a 5-volt peak-to-peak square wave with a DC offset of 2.5 volts peak-to-peak. Utilizing the 5-volt square wave as the input signal we should expect to see the square wave transform to match the step-down or step-up voltage we set it to. If the results of the logic level converter seem to match up with our groups expected voltage results, then the ultrasonic distance sensor shall be deemed viable for use in our project.

7.3 – Software Test Environment

The software for the hydroponics system will mainly be composed of designing the code for the microcontroller, Wi-Fi module, Android App and the database communication. The Atmega 2560 microcontroller and the Wi-Fi module will primarily be programmed with Arduino software. This will provide us with a familiar testing environment to those used in previous classes and projects. Through serial communication we will be able to exchange data from the various sensors located in our project in a quick and efficient manner and be able to identify any errors that may occur and troubleshoot from these readings.

Because the Arduino platform is so prevalent among microcontroller development and design, there is a very strong user driven community online that is able to assist with troubleshooting and development issues. A large library of code is built-in to Arduino for simplicity, along with many user created libraries available online. Also, Arduino allows for easy installation of common microcontrollers like ours.

The Android Application will be developed and troubleshot using the Android Studio IDE and the Java Programming Language. A majority of the testing of the Android application will be done using an Android virtual machine to emulate the environment of an Android smartphone. Once sufficient testing has been done on the virtual machine through Android Studio testing will be done on the group members personal devices to ensure the application runs as designed on real world devices and in a real world situation. When choosing between developing with the Java programming language or the Kotlin programming language, it was clear to us that because all of our team members had some experience with the Java programming language, it would allow all members to have some input and comprehend all the code being developed for the hydroponics system.

Along with the aforementioned information about the testing of our Android Application, we also need to discuss how we intend to test and validate that our application is successfully able to communicate with our database provider. For our hydroponics system we chose MongoDB as our cloud database. Further details of how we will validate that data is successfully created, stored, and exchanged between the Android Application and database will be further detailed in the Software Specific Testing section below.

7.4 – Software Specific Testing

The following section will describe in detail the testing and debug tools used on specific components of the software design. Furthermore, the individual tests scenarios used on each section will be detailed and made to reflect what the section will be responsible for in the final hydroponic system. As mentioned in the software design section of this document, there are 6 main panels and some smaller panels that need to be tested and verified as ready for final use.

7.4.1 – Login Screen Testing

The login screen is the first panel that the user interfaces with upon opening the hydroponic systems Android application. From here we will have to test to make sure that users are reliably able to create an account upon first using the application. Should the user already have an account, we need to ensure that they are able to successfully log into their account with the credentials they used upon creation. Beyond account creation and account entrance, we need to ensure that the panel is displayed correctly across a variety of commonly used screen sizes and resolution with correct resolution and placement.

The method used to test account creation and account entrance will be to first create several different accounts to be stored on our database, but not yet through the application. After we have successfully created some accounts to be used for tests, we shall attempt to log into the app using one of these tests accounts. Should the Android Application be able to successfully validate that the account is correct through the database and pull in the user information, then we can confirm that user login is working as intended. After we have validated that users are successfully able to log into the tests accounts, we will focus on creating accounts through the application itself. Information entered by the current user upon account creation in the application should be sent back to the database and all stored in one user's account.

7.4.2 – User Account Screen Testing

The user account panel appears after the login panel and handles users personal information. The first feature of the user account screen to test is the information button that will display common information about the user like name and number. Using multiple of the test accounts that we create, we should successfully be able to bring up information about the logged in user that is stored in our database and also allow the logged in user to upload a personal picture and edit any of their information. The next part of the user account screen to test is the help button portion that will instruct the user how to use the application through images and examples. In order to test this portion of the application we shall create several example photos and videos along with accompanying text that explains each photo/video that should be displayed upon the user interacting with this button. Should the help button successfully display our display media and texts, then we shall consider the help button portion of the user account section as working.

The next relevant section of the user account screen to test is the quick view of the users current and former plants in the hydroponics system. Test plants will be assigned to the test user along with made up statistics that are to be displayed in the plants vitals screen section. If the users current and former plants are correctly listed and link to the appropriate plant vitals screen for each plant, then the quick view part of the user account shall be deemed as working as intended. Finally, the last part of the user account screen to test is the notifications portion. Any alerts from the hydroponics system should be clearly visible in the lower portion of the screen with the ability to expand to read more about each notification. Should the notification portion of the user account screen be functioning, we should then move on to testing the push notifications portion in order for users to still receive alerts when outside of the application. Beyond the functional aspects of the user account screen, we need to ensure that the panel is displayed correctly across a variety of commonly used screen sizes and resolution with correct resolution and placement.

7.4.3 – Plant Vitals Screen Testing

The plant vitals screen will provide up to date information on the chosen plants condition and environment. To begin testing we will use test plants that have been assigned to our test account with constant plant vitals. These vitals include the plants temperature, pH level, light level, nutrients level, water level, and more. Should we get the screen running correctly using the test plant with constant values, then we will try changing the constant values periodically to ensure the screen is able to update with plant changes over time. With all the testing using a test plant validated, we shall move on to using a real user assigned plant and pulling in all our plant vital ratings from the sensors connected to our microcontroller. Should the screen accurately respond to our real life sensor readings, we will deem the plant vitals screen as ready for use in our hydroponics system. Finally, after all functional portions of the user account screen have been completed, we will ensure that the screen displays correctly across a variety of commonly used screen sizes and resolutions.

7.4.4 – Plant Search Screen Testing

The plant search section will allow the user to scroll through the hydroponics systems database of available plants to be grown. Each plant will have preset conditions that are ideal for that plant's growth, and if that plant is selected, then the hydroponics system will automatically create an environment advantageous to the plant's growth. The first feature of the plant search screen to test is that all available plants to grow are displayed to the user with a picture and description. Once this is done the search feature will be implemented and tested to ensure that users are able to quickly go to the plant they want without needing to scroll through all the available options. Should the search feature work with the tests plants available, we shall move on to controlling the hydroponic system.

The selection in the plant search screen has to be able to modify the parameters that the microcontroller system will work to contain. We will test this by choosing a plant in the plant search system for the hydroponics system to adapt to. Within 5 hours the microcontroller system shall correct the environment to match the desired parameters and report it’s status every hour to the Android application. Once we have user plant selection working and successfully relaying that information to the microcontroller, we will next test allowing users to create their own plant conditions. If the plant is successfully added to the application and database and working with the plant management system then the functional aspects of the plant search screen will be validated. After the plant search screen is able to implement all of our planned features, we will polish the user interface to work and verify that it works on a variety of common screen sizes and resolutions.

7.4.5 – Notes Screen Testing

The notes screen will allow users to take simple and quick everyday notes about their plants in an easy to use and basic screen. The first thing to develop and test in this section is the ability to make a basic text note that will be saved to the user account and be available across any devices the user uses with the application. Should the text portion work fine, we will test the ability to add multimedia to the notes such as plant progress pictures. With the ability to create and store user notes being tested, we will begin to test the search features of the notes screen. We will input test searches that should parse through the available notes for some that contain contents similar to those of the input string. Tags will be used for searching as well as basic string searches. By choosing a tag such as “Progress”, all notes that have been tagged as “Progress” notes for the current user should appear in the search results. Once user notes creation, string searches, and tag searches have been developed and verified, the functional parts of the notes screen shall be complete. With the notes screen being functional and working, we will finally focus on the presentation aspect to ensure that button placement and presentation are consistent across a variety of common smartphone resolutions and displays.

7.4.6 – Maps Screen Testing

The maps screen will act as a way to access hydroponic specific functions of the Google Maps API. By including this section in the application we save the user save by keeping them in our application instead of using an external application. The aspects of the maps screen that will need to be tested are searching for hydroponics suppliers in the users nearby area for supplies and finding repair locations. If the Android application is able to successfully find nearby retailers and suppliers compared to those of a typical

Google search, then we will consider the maps screen to be working and capable of being used in our final design.

7.4.7 – Logs Screen Testing

The logs screen will give the user a deeper understanding of what is happening to the hydroponics system from the outside. The sensor readings of the hydroponics will be relayed every hour and the information sent back to the database of the application. This is the first thing we will be testing, and if the readings of the sensors are reliably sent back to the database every hour we will move on to making sure that the user is able to read these readings through the logs screen portion. The log screen portion will be tested to have a new log card every hour or so with the vitals from that respective time. Once log cards have been successfully tested and implemented the final thing to test is the search feature. Several time frames will be tested to ensure the application returns logs that fall within the searched time frame. After the search is implemented the last thing to test is the overall aesthetic design and implementation of the logs screen across multiple devices resolutions and sizes

8.0 – Administrative Content

This section discusses the administrative content of this project. The previous sections in this document discussed the research, design, and a plan for building our hydroponics system. It also went into ways of testing our prototype design. Now that all of that is taken care of, this final section of the document examines the objectives completed or will be completed in the future. This section talks about the challenges faced achieving a milestone, schedule for completion of this project using major milestones, budget discussion for the project, and a section on management and division of labor among the four group members.

8.1 – Milestones Discussion

This section talks about the major milestones deadlines that are used to keep the team on track regarding the research, design, and building phase of the project. Furthermore, it also talks about some of the challenges that might be experienced reaching the milestone or might experience in future. It also talks about how the workload was distributed based on everyone's interests and skills.

8.1.1 – Senior Design 1

Senior Design 1 is about selecting and researching a project which aligns with everyone’s interests in the groups. The project was selected based on a few factors in mind. The first thing considered was selecting a project which can be completed in two semesters time. This was very important in order to ensure that the project can be completed in time, but not too small that the requirements for the project aren’t met. For the initial few weeks of the semester, our group researched a lot of ideas for what project we wanted. Hydroponics was one of them. Other ideas that we talked about were a smart skateboard, or ways to make an old car smarter using cameras and sensors. There was a bit of confusion in the beginning on what project to select since we weren’t sure about the complexity and the time it would take for these projects. Hydroponics was a very popular project which had been done by a lot of groups in the past. Also, the topic was something new that all of us were interested in. The project itself is very flexible and we can choose to add/remove parts from the project to make it more/less complex. Also, since we have 2 computer and 2 electrical engineering students, this project is perfect such that not only it requires working with circuits and electric components but also requires software design.

Most of the major milestones for Senior Design 1 is related to accomplishing major design decisions for our hydroponics system. We spent numerous discussions on what hydroponics system we want to implement. During our research, we came across 6 different types of hydroponics. We wanted to make sure it was the best type for the plant we wanted to grow. One of the other milestones was to come up with the physical structure for our design based on the major design constraints. The physical dimensions, material used, the functionality of the design and much more was discussed. Another example of a milestone in design was selecting how we wanted to power the major components of our design. Since using solar cells would add to the complexity of our design, we decided to use the standard wall outlets. We also decided to implement relays in order to control different components. During our research, we encountered many significant milestones like the projects done in the past. As a result of accomplishing them, we achieved a clear and practical design.

One of the recent milestones we accomplished was selecting the components such as sensors, lighting system, pumps and others. This is a very important step in order to get prepared for the next semester where we need to build our system. The two main things that were considered when selecting our parts were: the cost and the technical description to ensure the component will work with our system. We are still in the process of acquiring all the parts and need to make sure we can order them as soon as possible, especially with the delays that the coronavirus situation will cause.

Another major milestone that we still need to work on is finalizing the design of our PCB. This is the most challenging milestone for us as a group since we aren’t much experienced with PCB design. We have the requirements defined and we have an idea of how many pins we will need for all our sensors, relays, and peristaltic pump. We have decided to use the base PCB design from open-source hardware provided by Arduino. This will be a good starting point for our PCB design. Next, we need to work on removing all the unnecessary components from the schematic, insert the components based on our requirements and submit the Gerber file to a PCB manufacturer. We are still a few weeks away from completing our PCB design.

During the semester, we were given deadlines to work towards our design report. The final page submission is for 120 pages minimum. There were smaller deadlines at regular intervals which helped us reach the final page requirement. Table 8.1 covers the milestones that were accomplished during the semester and the time required to complete them.

Milestones	**Due Date**
Senior Design 1 Project Idea Brainstorming	01/10/20
Divide and Conquer Document	01/31/20
Research past projects	02/16/20
Individual research writing	04/21/20
60-page Senior Design 1 Draft	03/20/20
Prototype testing	04/21/20
100-page report	04/03/20
120-page report	04/21/20

Table 8.1 - Senior Design Milestones

8.1.2 – Senior Design 2

Senior Design 2 is where our hydroponics system will be built and tested. Since our project requires an extended time to test, working on building our project earlier will be very advantageous. Our Senior Design 2 starts in the fall semester, giving us the entire summer break to get a head start on our project. During the summer, we plan on getting all the parts required for the project, start working on the app, designing and ordering the PCB and much more. The more time we spend in the summer towards our project, the more beneficial it will be later. Since some of us still have one or two classes to take with this class, having to do as little as possible will be ideal.

Senior Design 2 milestones include ordering the parts, assembling the parts, building the prototype, coding the microcontroller, and testing and debugging. There is also a report that needs to be written like Senior Design 1. After the project is built and tested, it will be presented to an engineering committee at UCF that consists of professors in the EECS Engineering Department. We have decided to make an aggressive schedule and start working in the summer to compensate for any unforeseen circumstances in the design phase or any part failure. Due to the coronavirus situation going on around the world, the parts will take extended time to arrive. Thus, our goal will be to order all the parts including our PCB board before the start of Senior Design 2. Also, we plan to start on building the app required for this project during the summer. The testing of our project will require from a few weeks to a couple months. We need to be able to let a plant grow in our hydroponic system to show that the project we built was successful. The rough estimated milestones are given in Table 8.2.

Milestones	**Due Date**
Order/Test Parts	08/01/20
Build Project / Program	09/30/20
Test Prototype	11/20/20
Senior Design 2 report	11/20/20
Final presentation	12/05/20

Table 8.2 - Senior Design 2 Milestones

8.2 – Budget and Finance Discussion

Even the basic Hydroponic system can be complicated and requires a lot of components. It requires the user to monitor the pH level, TDS, humidity levels and much more. Our system also implements a lighting system, implements a battery incase of power failure and has more functionality which adds to the cost of the project. Even though we have tried to keep everything minimal, the cost of our project will roughly add up to $1000. Since we don’t have a sponsor, we have decided to split up the cost evenly. One of the most expensive items was our LED lights. This came in at around $100. Given that we plan on adjusting the light intensity with our microcontroller depending on the conditions, we decided to make the led move up and down depending on the need. We used a distance sensor in order to achieve this functionality. This was one of the more expensive parts but designing and implementing it ourselves would be too big of a task considering we have everything else we need to do.

Most of the sensors used for this project have been relatively cheap. For example, light sensors or the distance sensor were only a few bucks each. Even the microcontroller and the Wi-Fi chips we decided were super cheap. For automating the pH and TDS adjustments, we decided to go with digitally controlled pumps. We were able to find cheap peristaltic pumps for pumping the pH and EC solutions into our water as needed. Those were also super cheap, and we thought it would be a good idea to get those. During our research, we came across several all-in-one pH and EC monitor and pumping systems but those were far too expensive, and it was our job to automate our system. One of the more expensive sensors was the water level sensor. We found one at around $20 which would work with our Arduino. Also, we decided to implement a LCD screen on the body of our hydroponics system so the user can easily read the data without having to access the app. These were some of the components that we would be using for our projects. I'm sure we will have to add/remove some parts as we start implementing our project. Building the physical structure of our hydroponic system will also cost us a bit. Also, getting the PCBs and electric components for them will add to the expenses. Based on the previous hydroponics projects, the budget we chose is within reasonable range. We are most likely to go under our budget but barring some unforeseen circumstance, it should be enough to cover all the expenses. In the case we don't use all our budget, we will divide the remaining equally between all the members. Table 8.3 below shows the breakdown of our expenses so far:

Atmega 2560 Microcontroller	$15
Arduino Mega	$30 (including power supply)
Peristaltic pump	$25
pH sensor	$15
Temperature Sensor	$10
Distance sensor	$5
Water level Sensor	$20
Light sensor	$7
TDS sensor	$20
WiFi chips	$20
LCD	$20-$30
LED Lights	$100
PCB construction	$50
Assorted electronic parts	$50

Table 8.3 - Breakdown of expenses

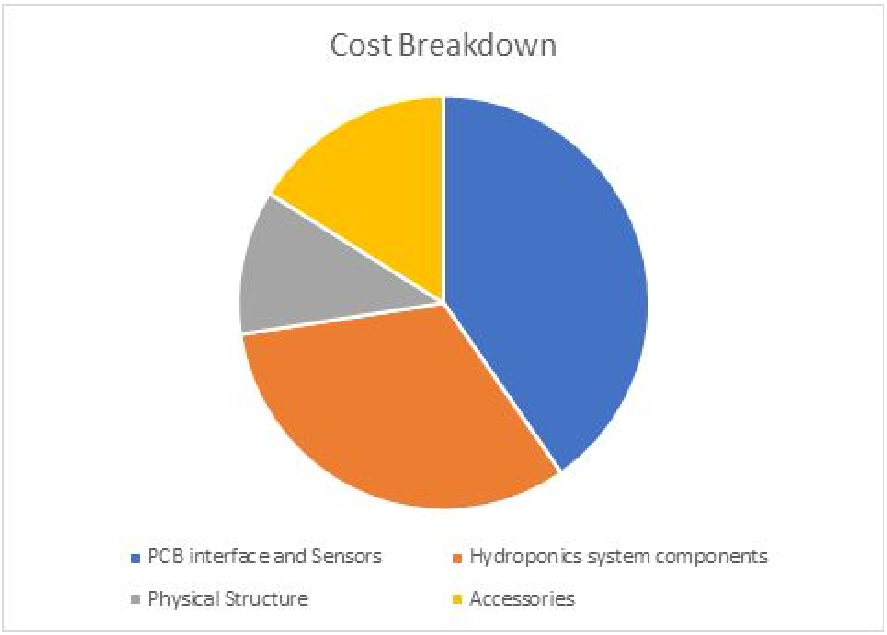


Figure 8.1 Cost Breakdown

8.3 – Division of Labor

Different topics of this project design and construction have been divided according to the respective interests of each team member. Each member was responsible for the research and design of the part assigned to him. It is important for each member to take the responsibility for the success of the project. This section talks about what sections were assigned to which group member and the reasons behind it.

Jarrod is the team leader since he is the most knowledgeable with the technology related to hydroponics. He is the one organizing meetings and leading them. He brings up the topics that need to be discussed and everyone chimes in with their insight. He was responsible for researching the electronic components used in our hydroponics system such as sensors and microcontrollers. He is also responsible for the software side of things including coding for the microcontroller and the app. Luiz also helped with the software stuff.

Luiz and Hardik were responsible for the design of the physical structure of our hydroponics system and the science behind the hydroponics system. We were able to understand the actions and precautions that need to be taken when running a hydroponic garden. We were responsible for reservoir design, the housing that supports the water solution and the root support system. We were also responsible for the system which adds oxygen to the water solution which helps plants grow better. We were also responsible for figuring out what plant we want to grow that is best suited for the type of hydroponics we wanted to implement. Also, the nutrients that will be suitable for the plant and their growth stages. Luiz was also responsible for coming up with the app design and coding with Jarrod. Luiz and Jarrod were responsible for the software side of things since they both are computer engineering majors.

Chris is responsible for helping Jarrod with researching electronic parts used for our system. His main responsibility is to figure out the PCB design for our project along with Hardik. Since they both are electrical engineering students, they were responsible for PCB design and the hardware side of the project. Chris and Hardik were also responsible for the power design of the system. Chris and Hardik were also responsible for the Administrative tasks such as organizing research efforts and design goals for the hydroponics project.

In the end, we decided to divide our projects based on everyone's interest shown in that topic and the major they are going for. Since we are 2 electrical engineering and 2 computer engineering majors, it was easy to divide the project up into the hardware and software side. Even though each member was responsible for a section, we were willing to help each other out when needed.

Appendix A - References

- D'Anna, Christina. *The Must-Know Facts About Hydroponic Lighting*. 4 July 2019, www.thespruce.com/hydroponic-lighting-basics-1939224.
- D'Anna, Christina. *Learn the Basics of Hydroponics: the Most Efficient Gardening Method*. 6 Jan. 2020, www.thespruce.com/beginners-guide-to-hydroponics-1939215.
- Godfrey, Mia. *What You Need to Know About PH in Hydroponics*. 19 Oct. 2018, university.upstartfarmers.com/blog/what-need-know-ph-in-hydroponics.
- *Gravity: Analog TDS Sensor/Meter for Arduino*. www.dfrobot.com/product-1662.html.
- *HOW TO GROW HYDROPONICS: TEMPERATURE & HUMIDITY*. www.hydroponics-simplified.com/how-to-grow-hydroponics.html.
- *Hydroponic Gardening for Beginners*. www.hydroponics.net/learn/hydroponic_gardening_for_beginners.php.
- *Hydroponic Systems 101*. www.fullbloomgreenhouse.com/hydroponic-systems-101/.
- *Hydroponics*. 14 Apr. 2020, en.wikipedia.org/wiki/Hydroponics.
- "What Is Hydroponics? - A Simple Introduction." *Explain That Stuff*, 5 Jan. 2019, www.explainthatstuff.com/hydroponics.html.
- "Hydroponics." *NAL*, www.nal.usda.gov/afsic/hydroponics.
- "What Is Hydroponics?" *Simply Hydroponics – What Is Hydroponics?*, www.simplyhydro.com/whatis/.
- "What Is Hydroponics?" *General Hydroponics*, generalhydroponics.com/about-us.
- "ATmega2560." *ATmega2560 - 8-Bit AVR Microcontrollers*, www.microchip.com/wwwproducts/en/ATmega2560.
- "Arduino Mega 2560 Rev3." *Arduino Mega 2560 Rev3 | Arduino Official Store*, store.arduino.cc/usa/mega-2560-r3.
- "Android Developers." *Android Developers*, developer.android.com/.
- "RAD Studio." *Embarcadero*, www.embarcadero.com/products/rad-studio/.
- "High-Intensity Discharge Lamp." *Wikipedia*, Wikimedia Foundation, 4 Feb. 2020, en.wikipedia.org/wiki/High-intensity_discharge_lamp.
- "Niwa: Get Growing Fruits and & Vegetables at Home!" *Getniwa*, getniwa.com/.
- Cloudponics. "Cloudponics Fully Automated Plant Grow Systems." *Cloudponics*, cloudponics.com/.
- "Lithium-Ion Battery." *Wikipedia*, Wikimedia Foundation, 19 Apr. 2020, en.wikipedia.org/wiki/Lithium-ion_battery.
- "Top 10 Free PCB Design Software for 2019 - Electronics-Lab." *Electronics*, 13 Sept. 2019, www.electronics-lab.com/top-10-free-pcb-design-software-2019/.
- "1149.1-2013 - IEEE Standard for Test Access Port and Boundary-Scan Architecture." *IEEE*, standards.ieee.org/standard/1149_1-2013.html.
- "2700-2017 - IEEE Standard for Sensor Performance Parameter Definitions." *IEEE*, standards.ieee.org/standard/2700-2017.html.
- "Quality Guidelines: Android Developers." *Android Developers*, developer.android.com/docs/quality-guidelines.

- “Understanding and Interpreting Lux Values.” Win32 Apps | Microsoft Docs, 2018, docs.microsoft.com/en-us/windows/win32/sensorsapi/understanding-and-interpreting-lux-values.
- Davis, Nick. “The History and Basics of IPC Standards: The Official Standards for PCBs.” All About Circuits, 20 Oct. 2017, www.allaboutcircuits.com/news/ipc-standards-the-official-standards-for-pcbs/.
- Hammond. “Definition of Protection Grades IEC 60529.” *Hammond Mfg.*, www.hammfg.com/electrical/technical/iec.
- McClellan, Dalin. “JTAG Explained (Finally!): Why ‘IoT’ Makers, Software Security Folks, and Device Manufacturers Should Care.” Senrio, 28 Sept. 2016, blog.senr.io/blog/jtag-explained.
- Oram, Brian. “The PH of Water.” Water Research Center, water-research.net/index.php/ph.
- Teel, John. “Tutorial (Part 1): How to Design Your Own Custom Microcontroller Board.” *PREDICTABLE DESIGNS*, 15 Mar. 2020, predictabledesigns.com/tutorial-how-to-design-your-own-custom-microcontroller-board-video-part1/.
- “The Basic Hydroponic System Types.” *Home Hydro Systems*, www.homehydrosystems.com/hydroponic-systems/systems.html.

Appendix B - Copyright

- “ESP8266 Hardware Design Guidelines.” *Espressif*, Espressif Systems, www.espressif.com/sites/default/files/documentation/esp8266_hardware_design_guidelines_en.pdf.

Appendix C - Datasheets

- Liu, Thomas. “Digital-Output Relative Humidity & Temperature Sensor/Module .” Aosong Electronics Co.,Ltd, components101, components101.com/sites/default/files/component_datasheet/DHT22 Sensor Datasheet.pdf.
- Albert, Dan. “Sonar Rangefinder (HC-SR04).” *TekBots - Sonar Rangefinder HC-SR04*, eecs.oregonstate.edu/education/hardware/hcsr04/.
- “DS18B20 Programmable Resolution 1-Wire Digital Thermometer .” *Maxim Integrated Products*, Adafruit.com, 2008, cdn-shop.adafruit.com/datasheets/DS18B20.pdf.
- “Liquid_Level_Sensor-FS-IR02_SKU__SEN0205.” *DFRobot*, wiki.dfrobot.com/Liquid_Level_Sensor-FS-IR02_SKU__SEN0205#target_5.