SMART Garden

Kevin Lorch, Ryan Hassan, Jonathan Wallhauser, Lauren Melancon

### **Dept. of Electrical and Computer Engineering**

### **University of Central Florida, Orlando, Florida, 32816-2450**

***Abstract*-- As inflation in the United States continues to rise, and global supply chain issues persist, working class people are finding it harder and harder to be able to find or afford healthy food options at their local grocery stores.  In fact, according to Morning Consult's latest U.S. Supply Chains & Inflation report, over half (51%) of consumers surveyed in March 2022 reported experiencing product shortages of specific types of groceries and food, up from 43% in September 2021. With that being said, the looming fear of large-scale food scarcity coupled with the steadily declining economy have forced many Americans to turn to unhealthy food alternatives. The smart garden presents a small but impactful way to help counteract this issue and benefit the lives of those in need.**

**Our goal in creating this product doesn’t stop at financially benefiting the lives of the people who use it. The SMART Garden serves as a way to better the health of the individual, both physically and mentally. The automation that this system offers was made and implemented to give time back to the user in a world where almost every social media platform or product is trying to do the opposite. Utilizing this tool will help the user develop a potentially important skill such as gardening while simultaneously helping them avoid the unhealthy grocery store alternatives to food that can be easily grown at home. Not only that, but gardening has been proven to have a positive effect on mental health, mood, and even brain chemistry. In fact, in a 2015 study published in the journal Ecopsychology researchers showed that “gardening provided a space of one’s own... increased feelings of connectedness, and improved physical and mental health”. The SMART Garden is tool that will have a positive and real impact on those who use it consistently.**

# Introduction

Food price inflation has already become an issue in 2022 and it is only projected to get worse over the next year. In fact, the food-at-home CPI, which measures the consumer price index for grocery store or supermarket food purchases has increased 0.6 percent from August to September of this year and is 13 percent higher than it was September of last year. That isn't even the worst of it, in 2022 all food prices are predicted to see an increase of upwards of 10.5 percent, with food-at-home prices increasing 11 to 12 percent and food-away-from-home prices increasing 7 to 8 percent. This issue isn't just local to the United States, supply chain issues caused by climatic events and political conflict have caused economic issues all over the world. Food insecurity has already begun to become a serious issue and we are only at the start. Problems like these are some of the large-scale issues we hope to alleviate with our cost-friendly SMART Garden.

Our vision is to create an automated plant watering system that uses soil moisture, temperature, and humidity sensors to monitor herbs and vegetables in an at-home garden. The goal of this data collection would be to notify the user when corrective measures in regard to sunlight exposure or watering frequency for any given plant are needed via a web application. An online user interface will also allow for the implementation of automated water scheduling capabilities and open up the possibility for the user to be able to access large plant databases or view live status updates on their personal garden. The previously mentioned peripherals matched with the UI for the end user will provide a great user experience which optimizes the way the general public grows and harvests plants. This system will allow the user more time on a daily basis to be spent doing higher priority needs, while still allowing them to enjoy the many benefits of growing their own plants. This device will give users a cost friendly gardening option in hopes of increasing access to healthy food alternatives.

This document serves as a tool to report the SMART Garden design process. To start off, we will discuss the motivations behind why we chose this project and provide a comparison of competing products. Once this is complete, we will go over the objectives, project goals, and requirement specifications. The next section of the paper will be an overview and explanation of our House of Quality matrix followed shortly after by a review and analysis of all relevant technologies as they relate to the project's hardware set-up. After part selection and comparison, we will discuss decisions related to the software and communication systems. Directly after we will go over our groups initial PCB design and schematic and lightly touch on future manufacturing plans. Wrapping up the paper we will have an in-depth discussion on project operations as well as finalize all administrative content.

# Objectives

The primary objective of this design project is to design and assemble an automated gardening assistant that can be utilized in a remote fashion. This system should be able to apply for gardens of varying levels of sophistication. It should be scalable to operate from a very small indoor garden setup to a larger more robust setup. This would mean that users can use a single SMART garden for a very simplistic garden setup, or they can use multiple SMART gardens in conjunction for a more robust setup. Our project is separated into multiple different subgroups that work in unison. As such each group is designated its own specific objectives whose collective mechanisms produce the desired results. Listing these objectives individually is necessary to clarify the requirements of this project’s design.

## Hardware

The hardware for this project will consist of a development board, which will host a Wi-Fi module, power system, moisture sensors, humidity sensor, containers for the plant food and water, and a laptop. This setup will allow the user to see the sensor reading and adjust the system settings for the SMART gardens system easily remotely through the web application using a computer or mobile device. Finally, it includes the container of the plant itself.

## Software

The software for this design will include many different subcomponents used to operate the controls, sensors and data management. The software will collect data and transfer it from the development board to the server, so it can be accessed by the computer or mobile device. This will allow the user to set up the automated water/feeding remotely when desired. This includes the IDE used to program the Arduino Uno we will be using to control the SMART garden. This also includes the Backend and Frontend coding which is comprised heavily of Java Script code and HTML code.

## Control

The Development Board will act as the intermediary between the smart garden system and the user’s device. The microcontroller will relay information to the computer which will do all the calculations and processing and return instructions for the microcontroller to enact on the system itself.

## Power Supply

The main goal of the power supply of this project is intended to be as efficient as possible. The power system will consist of a wall outlet power-source.

# Requirements and specifications

|  |  |
| --- | --- |
| **Size Specifications** |  |
| Base | 12 Inches |
| Length | 16 Inches |
| Height | 8 Inches |
| **Accuracy** |  |
| Accurate readings | 90% of time |
| Optimal Conditions | 95% of time |
| **Battery Life** |  |
| Lithium Ion | 18 Hours |
| **Reservoir Pump** |  |
| Flow Rate | Greater than 70 Liters per Hour |
| Operating Range | 3.0-5.5 Volts |
| **Database Plants** |  |
| Herb | Cilantro |
| Herb | Basil |
| Herb | Parsley |
| Herb | Oregano |
| Herb | Sage |
| Vegetable | Carrot |
| Vegetable | Tomato |
| Vegetable | Potato |
| Vegetable | Green Onion |
| Vegetable | Kale |
| **Protype Plants to be Tested** |  |
| Herb | Cilantro |
| Herb | Basil |
| Herb | Oregano |

*Table 1: Engineering Specifications*

# System Components

## Microcontroller

The overall design and functionality of our system is dependent on a reliable microcontroller that can seamlessly integrate all necessary hardware components together. The SMART Garden project utilizes the ATMega328p-pu with an external 16MHz crystal.

This microcontroller chip is low cost, easy to use, and meets all performance metrics our system requires. In addition, due to the fact the ATMega328p is used in the Arduino Uno development board, we were able to utilize the Arduino Uno IDE, as well as a wealth of information that is available through the development boards vast user base. Figure 3 shows the connections made to the microcontroller to appropriately connect it all hardware components.

## Soil Moisture Sensor

The soil moisture sensor is a crucial aspect of our overall design. The sensor chosen for our project is the HiLetgo LM393. This sensor has four pins. It requires 3.3V-5V input voltage, an external ground, and has two I/O pins for both digital and analog output. This device also has two pins which are connected to a fork-shaped conductive probe. This probe is made up of two exposed conductive plates that act as a variable resistor.

The values obtained from these conductive plates range from 0, when the sensor is completely submerged in water, to 1050 when the sensor is completely dry. After research on the ideal moisture level for all plants being tested, we associated a set of values in this 0-1050 unit range that defined all readings from the soil moisture sensor as being either too dry or ideal.

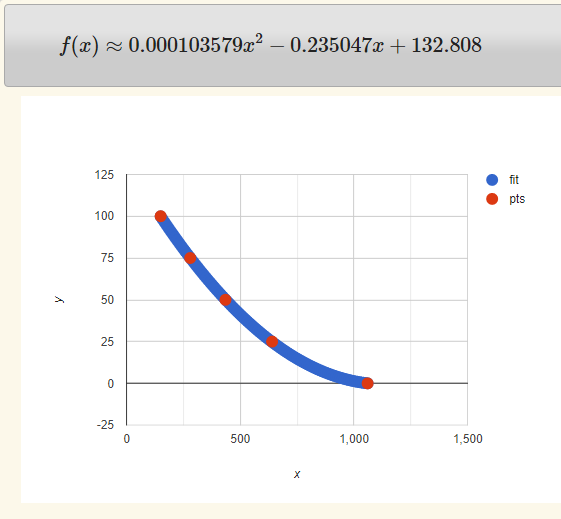
|  |  |  |
| --- | --- | --- |
| Soil condition with relation to an average plant’s needs | Soil Moisture Content:  Analog Representation | Soil Moisture Content:  Percentage |
| Soil too Dry | 150 – 619.9 | < 27% |
| Optimal Soil Moisture | ~620 | 27% |

*Table 2: Soil Moisture Conditions for Basil Plant*

We supplemented this by establishing a number of baseline readings. First established a baseline of 80mL of water per ¾ cup of soil to be considered sufficiently wet enough to be considered 100% moist. This gave us a reading of 173. Then we measured 60mL of water per ¾ cup of soil as 75% wet. This gave us a reading of 280. Thirdly we took a reading for 40ml of water per ¾ cup of soil as 50% wet. This gave us a reading of 455. Next we took a reading of 20mL of water per ¾ cup of soil as 25% moist. This gave us a reading of 640. Finally, we took a reading for soil that we considered to be completely dry. This reading read as 1010. Given the data collected for our readings we were able to create an equation to establish what percentage of wet the soil was. This equation allowed us to establish reasonable ranges for the state of the plant moisture, from “too dry” to optimally moist. Below is the data chart and associated graph.

|  |  |
| --- | --- |
| Percentage of wetness | Sensor reading |
| 100 | 173 |
| 75 | 280 |
| 50 | 455 |
| 25 | 640 |
| 0 | 1010 |

*Table 3: Soil moisture benchmark readings*



*Figure 1: Equation calculated from soil benchmark readings*

## Relay Module

In order to supply water to the plants and soil mixture located in the main compartment of the garden a relay module is needed. A relay module is necessary to control when the water pumps turn on and off to push water to the soil.

The relay module chosen for this system is the Sunfounder 5V 2-channel relay. This relay module comes with 2 relays, each of which has a maximum current rating of 10A at 250VAC, or 30VDC. In addition, the Sunfounder relay module comes with built-in octocouplers on the logic inputs. These offer electrical isolation between the relay power and the logic control input which acts as an additional layer of protection for circumstances such as a failure on the relay’s AC load.

This module operates on a 5V input and draws around 140mA of current when both relays are activated. Flyback diodes are also included in the relays design, which serve to safely shunt current when the relay coil de-energizes. The relay module are located close to the water pumps for ease of access.

## Temperature and Humidity Sensor

The second sensor that will be utilized in the SMART Garden is a temperature and humidity sensor. The purpose of this sensor is to ensure that the external environment around the plant is within the ideal condition for efficient plant growth.

The temperature and humidity sensor chosen for this system was the DHT-22. This is a low-cost sensor, priced at just under eleven dollars, and produces values with an accuracy of ±.5 degrees Celsius and ±2%RH. This sensor will be mounted on the top of the gardens structure to provide the most accurate results possible for the end-user.

## Water Pump

The water pump is essential for the distribution of both water and nutrient solution to the soil mixture. The engineers on the SMART Garden project chose the Siphytoph DC 3-5V Micro Submersible Mini Water Pump which is priced at just over ten dollars.

This water pump is submersible and easy to install. Each water pump contains an inlet and an outlet allowing for the user to place the pump inside or outside the water reservoir, depending on their own personal preference. The pump is rated for voltages of 3 volts and 4.5 volts and can discharge water at a rate of 100L/H. The water pump is convenient due to its small size and minimal noise production. A total of two water pumps will be used for the smart garden project. For the purposes of demonstration, they will be submerged within the water reservoirs located on the side compartment of the garden’s physical design.

## Wi-Fi Module

The Wi-Fi module chosen for this system is the ESP 8266. This component plays a key role in ensuring the automation of our SMART Garden and is crucial in maintaining a working user interface.

The Wi-Fi module is in charge of pushing all updated sensor readings to the database to be displayed on the SMART Garden website. The ESP 8266 requires a 3.3 V input, has 1MB of flash memory, and requires minimal external circuitry for set-up.

## AC/DC Power Conversion

As for every system, a plan needs to be made for how the system will receive power. The initial plan for the SMART Garden system was to tailor the product to be used as both an indoor and outdoor garden. When this goal was in place the design for the power supply was to use 9V Lithium-Ion Rechargeable batteries to power the PCB and all other components.

The reason batteries were chosen was to ensure that the device was self-sufficient and could run without needing to be close to an outlet on the user’s home. Although after discussion, it was decided to revise the original goal of the system and focus on making an efficient design for an indoor only gardening system. After the system was redefined as an indoor gardening system, the need for portability that required the use battery packs was no longer a system restraint.

Another notable issue with the battery pack as a power source was its limited current capacity. This would require the end-user to have to spend unnecessary time replacing batteries. This inconvenience would defeat the purpose of the watering system and make the product less desirable if it entered production.

Given this information, the power supply was changed to a 9V 1A AC/DC converter with a wall adapter. This change slightly limits the mobility of the garden when indoors, but overall offers more benefit to the end-user.

# Electrical design

## Hardware Block Diagram

A microcontroller serves as the main computing power for our project. It is a device whose purpose is to gather a desired input or inputs, processes that information, and perform a specific task as a result of the information gathered.

In Figure 2 we see that the MCU is the core of our project's hardware set-up. It is connected not only to all four of our sensors (which are the main desirable features for our project) but also to the relay module, water pump, and power supply.

Chart, diagram

Description automatically generated

*Figure 2: Hardware Block Diagram*

## Schematic

In this section we will be reviewing the SMART Garden Schematic Design. A list of all included components on our schematic and PCB board design can be found in Table 4. All digital and analog I/O pins are routed to one of the two 18 position pin headers located on the left and right side of the schematic. The system requires a 5V regulator to supply power to three soil moisture sensors, a temperature and humidity sensor, a 2-channel 5V relay, as well as the ATMega328p microcontroller. As depicted in Figure 3, with the exception of the ATMega328p microcontroller, which is wired directly to the 5V output of the LM2576 voltage regulator, the LM2576 is supplying a 5V output to a total of 5 pins on our 18 position pin headers. These pins are used to power all other 5V hardware components via their on-board VCC pins.

Diagram, schematic

Description automatically generated

Figure 3: PCB Schematic

In addition, due to the fact all six hardware components need to be grounded using their on-board ground pins, there are a total of 10 pins on the 18-position pin header which are connected directly to ground. This leaves room for every component to be grounded via the pin headers and allows for flexibility of component placement and connection to the PCB board.

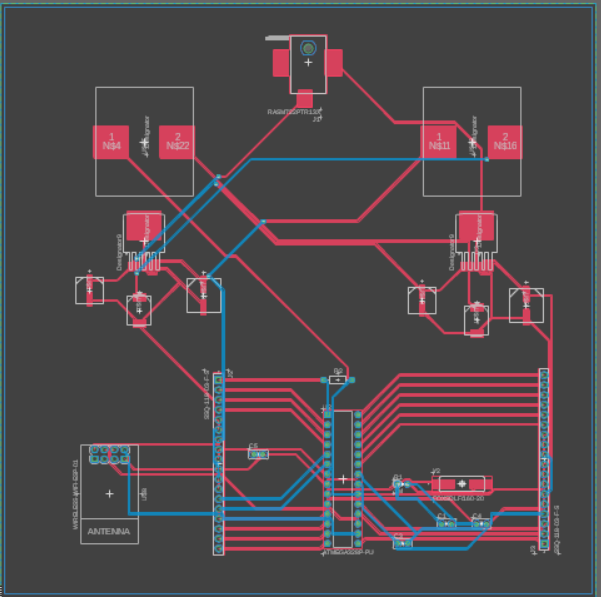
The Wi-Fi module is mounted to the PCB board and is routed to the output of the 3.3V regulator directly. To accommodate for the addition of another 3.3V component, if the user so chooses, we have also added a 3.3V pin on the 18-position pin header that is connected to the output of the LM2576 3.3V regulator.

|  |  |
| --- | --- |
| List of On-Board Components | |
| Microcontroller Chip | ATmega328pu |
| 5V Regulator | LM2576SX-5.0/NOPB |
| 3.3V Regulator | LM2576SX-3.3/NOPB |
| 16 MHz Crystal Oscillator | FOXSDLF/160-20 |
| DC Power Jack | RASM722X |
| Inductor (x2) | 100uH |
| Resistor (x1) | 1 MΩ |
| Resistor (x1) | 10 kΩ |
| Capacitor (x2) | 1000uF |
| Capacitor (x2) | 100nF |
| Capacitor (x2) | 22pF |
| Capacitor (x1) | 0.47uF |

*Table 4: List of On-Board Components*

## Printed Circuit Board

Shown below is our finalized 2-layer PCB design. The footprint of the board is 6000 × 6000mm (6 inch x 6 inch). Eagle and Fusion 360 software were used in the creation of the schematic, and Fusion 360 was used to create the final PCB board layout. The initial prototype of the PCB board was ordered through JLCPCB. The unordered updated design depicted in Figure 4 contains two 18 position pin headers which will be used to easily and efficiently connect all external sensors as well as our 2-channel relay module to our PCB board. Mounted on-board are the 3.3V and 5V regulator circuits, our ATMega328pu micro-controller and external 16Mhz crystal, as well as the ESP8266 Wi-Fi module. The board is being powered by a 9V 1A AC/DC converter through an on-board DC power jack.



*Figure 4: Printed Circuit Board Layout*

# Software Design

## Software Block Diagram

Diagram

Description automatically generated

*Figure 5: Software Block Diagram*

## Arduino IDE and ATMEGA Programming

Due to selecting the ATMEGA328P, the simplest way to program it was through the Arduino IDE. This application was designed for loading individual programs, known as ‘sketches’ onto the ATMEGA that would run continuously while powered, requiring no connection to a computer. This IDE also allows for the integration of a variety of libraries, both official and community-made, which simplifies many (perhaps needlessly complex otherwise) aspects of the code. With selectively including libraries, we can maximize the space on the ATMEGA itself.

The ATMEGA is able to read data coming into its pins, as well as sending data out through them. This is done by setting a pin as high or low voltage if the pin is set to output, or by using either the analogRead(pinName) method or a library’s own method to retrieve data from the pin. This was the case with our digital humidity and temperature sensor, which allowed for the use of the method ‘dht.read()’ to retrieve the sensor’s readings as percentage of humidity and temperature in Celsius. Other sensors provide less helpful readings when seen as raw data, and must be interpreted further. For example, the soil moisture sensors provided a number between 0 and 1023 that represents the resistance between the prongs. This is clearly not an obvious way to understand the moisture of soil, so we had to develop a way to interpret these values. An equation was determined based on values found through testing to find the equivalent moisture percentage based on these abstract values.

While many libraries are helpful, it is important to understand what they are designed for. If a library is designed for a chip with much more onboard dynamic memory, it can easily take up space that would be used for vital variables elsewhere. For example, rather than using a library to quickly parse a string in JSON format, it was necessary to read through the string character by character, since the variable used to parse such a string into easily accessible components required too much RAM. Attempting to use it interfered with other variables because the sheer size of the object overwrote much-needed data. A similar issue occurred when attempting to incorporate an LCD screen into the garden. The screen needed a buffer for operation that would require nearly one third of the dynamic memory available. Attempting to use these would cause issues with either stored data that would be necessary to send to the database or that had been retrieved from the database.

## Wi-Fi Integration

The Wi-Fi module used in this project was the ESP8266 ESP-01 serial Wi-Fi transceiver. Though more complex options exist, many were either more complex than required and overlapping with the ATMEGA328P in capabilities or were incompatible with our initial design for other reasons, such as voltage requirements. The ATMEGA was able to issue the Wi-Fi module instructions by sending ‘AT commands’ , or Attention commands, to its transmit (Tx) and receive (Rx) pins. The ESP-01 has a limited default library of AT commands compared to more complex versions, so the http request type and destination was determined by the data sent to it after prompting it with an AT command, rather than using a variety of more specific AT commands. Most essentially, the command ‘AT+CIPSEND=# ’ was used to prompt the ESP-01 to receive data to transmit to the previously provided web domain. The ‘#’ would indicate the exact length of the content it would be provided, which includes the file path on the website to more specific web addresses that would receive the data and make changes to the database based on the content provided.

The content required to update the database (with an http PUT request) was created based on sensor readings, but the content required to retrieve data (with an http GET request) about the current plant was constant, as the php file that returned this information didn’t require any information as an input.

## Database

The general purpose of a database is to store a large collection of data in an efficient manner so that multiple sources can access and update it. A critical part of developing this project was establishing a way to receive information from the sensors of the smart garden and use them to update the website element of this project. With this level of information integration, the SMART garden can effectively read the status of the plant and care for it accordingly.

The remote database we selected as part of the LAMP stack is MySQL. MySQL is a table-based system. The table-based architecture utilizes a data query structure. The simplistic and efficient database organization structure made it easy to organize our data and access it where desired. Our database stores all of the preset ideal plant settings. It also stores and custom ideal plant settings as well as the current readings for the plant that is actively in the SMART garden so that the SMART garden can feed and water it as necessary to keep it healthy. The user would be able to manipulate the databases indirectly, via the php files on our web server that receive data and use it to execute commands directly on the MySQL database, whether it is reading data or updating the selected plant.

## User Interface

The user of our smart garden would be able to change the active type of plant by visiting our website (smartgarden34.com) selecting a plant from the dropdown menu, optionally setting the name and ideal values of a custom plant, and clicking the button to submit. Beyond ensuring the garden has enough water and plant food in the reservoirs to maintain a plant adequately, this is the only thing required of a user- the garden takes care of the rest. It automatically retrieves the ideal values from the database and updates the website with new readings from the sensor. These readings can help the user know exactly what conditions they can provide their plant to help it grow. If the temperature is too high, they can try to move the garden somewhere in the house that is cooler, and if the humidity is wrong, they could try to find somewhere with higher humidity, or they could even go so far as to buy a (de)humidifier. It could also be the case that the conditions cannot be altered, in which case the user could look at these sensor readings and then try growing a plant that would thrive in their home’s environment. The smart garden allows the user to put in as much effort to grow a plant as the user desires, including almost none.

The majority of the ‘processing’ of the website is handled by JavaScript functions behind the scenes, which can send requests to php files to change or receive data in the MySQL databases. They can use the information from the php files to update HTML elements, such as by retrieving each option in the dropdown menu from the database on load, or take a user’s selections and use them as the basis of how to update the database. An example of the latter is that each of the dropdown’s options are based on the name from a list in the database, and selecting a plant with this name sends that name to a php file that sets the current plant name and ideal values to this name and the values associated with it.

## Project Organization

With so many aspects of the project relying on each other, it was important that we were able to stay on the same page even if we didn’t have the chance to meet up in person. To set up meetings and keep track of important documents, we used a group discord server. The variety of text channels there helped us organize important links and topics. A channel was as broad or specific as we needed it to be, from the topic of Arduino Programming to a channel simply called ‘ideas’ that we could post thoughts about solutions to current roadblocks we had faced, or what features we may be able to add. Another facet of project organization was the github repository we used to update the web server. We could synchronize our local files to the ones on the repository in order to prevent people from working with outdated files. In an environment that needs specific data types and formats with references to other files being a primary aspect of managing variables, version control is a very important thing to manage, especially when multiple people might otherwise be working on the same file at the same time.

# System Design

## Plant and Soil Compartment

Plants need to have plenty of room to grow since they need enough room to spread out their roots. This allows them to absorb enough water and nutrients from the soil to promote healthy growth. If the plant is large, ample root space can help anchor it in place, allowing for the plant to grow large without a chance of it uprooting itself due to its own weight. If there are multiple individual plants in the garden at once, spreading out can help them avoid competing for resources. With room to grow securely while also allowing for space between them, plants would also have enough airflow to avoid crowding each other for light and fresh air. Sufficient airflow helps prevent the buildup of moisture and, consequently, mold on the plants. Allowing for plenty of space for each plant also ensures the quality of the soil remains high. If the plants aren’t given enough room to grow, the soil would run out of nutrients too quickly to help the plants grow. For these reasons, the smart garden design includes plenty of space for multiple plants depending on the individual species’ requirements. There is enough surface area for multiple small plants, and enough depth to accommodate some of the larger plant species users may want to grow.

A picture containing window, floor, indoor, blue

Description automatically generated

*Figure 6: SMART Garden System Design*

## Liquid Reservoirs

While our final build simply used glasses to hold the water and nutrient solutions, an ideal design wouldn’t be much different. The ideal reservoirs would be removable for easy cleaning and made of plastic so that there would be no chance they would shatter and cause a safety hazard. Frequent cleaning helps prevent the buildup of dirt and debris in the liquid reservoirs, which could cause issues with the pumps’ water pressure and thus the health of the plants. Further, plastic is relatively inexpensive, allowing us to keep the production cost down and maintain the garden’s affordability for the intended demographic. Everyone deserves to be able to feed themselves without breaking the bank. Additionally, using a clear plastic container would allow the user to easily see the level of each liquid remaining and ensure the plant always receives sufficient watering to remain healthy.

## Nutrient Solution

The nutrient solution that we selected to utilize in our design is Miracle Gro Indoor Plant Food Foam. This plant food selection was very convenient and effective. Miracle Gro is a top-of the line plant food product. Its implementation as a foam makes it easy for us to mix in the plant food reservoir. It was easily mixed with an appropriate amount of water and stored. This allows us to dispense it just as easily as we do with the water. This plant food mixture utilizes urea. Urea provides carbon, oxygen, hydrogen, and nitrogen to plant. Urea phosphate is also included for its properties that aid in the growth and development of plants that produce flowers or fruits. It is also critical in preventing the growth of fungus that can kill the plant. Potassium chloride provides plants with the potassium that is necessary for the leave of the plants to maintain their green color and grow tall and healthy. The boric acid also aids in cell division so that the plans retain structural stability.

## Waterproofing and Enclosure

The enclosure is made of multiple pieces of wood tightly screwed together and supported by L-brackets to maintain stability. It consists of two main sections. The larger section of the enclosure is the part designed to contain the plants and soil. Its large space is sufficient to house multiple small plants with space to grow. It is waterproofed by multiple layers of waterproofing paint. The second, smaller section of the enclosure houses the plant food and water reservoirs. It is large enough that it can contain ample supplies of both resources to support the plant. This too is waterproofed through the use of specific paint. The lower portion of the larger enclosure is hollow to allow for the addition of a drip pan to allow for any excess water to be collected neatly.

# Conclusion

Our goal was to design a system that could help the average individual raise and maintain a plant. Furthermore, we intended for our design to be scalable so it could be used for a single plant or a large garden. We have succeeded in achieving our goals by creating a system that automatically monitors the conditions of a plant and maintains it accordingly. This will help individuals who may have busy schedules or who may forgetfully to successfully grow a garden for themselves.

The reason this is important is because, firstly, homegrown plants are healthier for consumption as you will not need to worry about harmful pesticides or GMO’s. Secondly studies have shown that maintaining a garden can be very helpful for improving mental wellness.

Our solution is effective because each unit can contain a few smaller plants in its housing. And each unit can monitor and maintain itself. This way a user can have one unit or several and it will still be relatively simple to maintain. The user will just need to make sure the food and water levels are sufficiently maintained, and that the batteries level is adequate. As with all engineering projects. The most critical technique we utilized for our project was effective research. By learning from he mistakes of those who created similar projects before us and utilizing the methods they used we were able to bypass a number of the hurdles that may have otherwise impeded our progress. Effective use of a multimeter was massively helpful in diagnosing any circuit set-up issues we encountered. Localized hosting through the use of Visual Studio Code was also a large time-saver, allowing us to see the changes to the HTML in real time before pushing the code to the live website server. If any issues were encountered that seemed to be caused by these updates despite these precautions, they had been first pushed to GitHub, which allows us to roll back the repository to an earlier version. This allowed us to undo mistakes and start back from a more stable build.

# Acknowledgement

As a group, we would like to acknowledge the efforts of all the peers, professors, and mentors who have helped to guide us during this long journey. We would also like to give a special thanks to Dr. Lei Wei, Dr. Samuel Richie, and our review committee for taking the time to help us through this semester.

**Biographies**

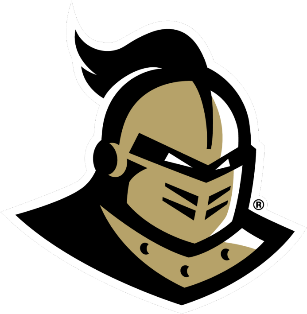
Kevin Lorch is a student at the University of Central Florida that will graduate in May of 2023 with a Bachelor of Science in Computer Engineering (BSCpE). Currently dedicating full time to studies and aspiring to work at NASA or an associated company after graduation.

A person wearing glasses

Description automatically generated with medium confidence

**Kevin Lorch (CPE)**

Ryan Hassan is a Computer Engineering (CpE) student scheduled to graduate in May of 2023. Currently working for the UCF student union as a technician. Plans to work in software development and management.



**Ryan Hassan (CPE)**

Jonathan Wallhauser is a student at the University of Central Florida that will graduate in May of 2023 with a Bachelor of Science in Computer Engineering (BSCpE). During his time at UCF, he held a position in leadership for the Surf Team at UCF for three years and president his senior year. After graduation, Jonathan will begin his career with Lockheed Martin as a software engineer a part of the F-35 Joint Strike Fighter program.



**Jonathan Wallhauser (CPE)**

Lauren Melancon is an Electrical Engineering (EE) major student, on track to graduate from the University of Central Florida with a Bachelor of Science in Electrical Engineering (BSEE) degree in May of 2023. She has received a job as a developmental engineer for the United States Air Force at Wright Patterson Air Force Base as a second lieutenant in the United States Air Force. She will commission as a brand-new second lieutenant later in May 2023 and start at her new base after receiving orders later this year.

A person smiling for the camera

Description automatically generated with low confidence

**Lauren Melancon (EE)**