

Automated Watering

Team Name: Easy Manage Garden

Team Members: Joe Biscan, Kyle Hansen, Anthony Ibarra, and Silas Chakravarty

Faculty Advisor: Zhao Zhang

Spring 2022

Idea Name: Easy Manage Garden

Report Allocation

Work shared between group members have equal contribution between those members include a statement of work identifying what parts of the report each team member worked on.

Joe Biscan - Engineering Requirements, Engineering Design Alternatives, Design Alternative Evaluation Criteria, Selection of Design Alternatives and Justification, Preliminary Design, Testing, Bill of Materials, Timeline, Conclusions, Parts of Appendices

Kyle Hansen - Final design, Design Alternatives, Engineering Requirements, Marketing Requirements

Anthony Ibarra - Marketing Requirements, Engineering Requirements, Engineering Design Alternatives, software design, software design for app and controller, Final User Manual

Silas Chakravarty -

Table of Contents

Automated Watering	1
Report Allocation	1
Table of Contents	2
Abstract	4
Project Goals	5
Needs Statement	5
Objective Statement	6
Background	7
Research Survey	8
User Survey	9
Marketing Requirements	10
Additional Information Collected	10
Engineering Requirements (ER)	11
Engineering Requirement Updates and Changes	12
Engineering Design Alternatives	13
Design Alternative I	14
Design Alternative II	16
Design Alternative III	17
Engineering Design Alternative Updates	18
Design Alternative Evaluation Criteria	19
Design Alternative Evaluation Criteria Updates	20
Selection of Design Alternative & Justification	20
Design Matrices	20
Final Decision	22
Design Alternatives Justification Update	22
Preliminary Design	22
Level 1 Schematic of System	22
Circuit Design	23
Software Design	24
Software Design for App	25
Software Design for Microcontroller	26
Preliminary Design Review	27
Testing	27
Hardware Testing	28
Software Testing	29

Comparing Engineering Requirements and Testing Results	30
Final Design	32
Circuit Design	32
Software Design	33
Bill of Materials	35
Task Allocation and Timeline	35
Week-by-week timeline	35
Contribution of Work	37
Lessons Learned	38
Conclusions	39
References	40
Appendices	40
Appendix A: Final User Manual	40
APP Manual	40
System Manual	42
Appendix B: Team Task Management Document	43
Appendix C: Concept Maps for Design Alternatives	46
Table C-1: General Concept Map	46
Table C-2: Design Alternative I Map	47
Table C-3: Design Alternative II Map	48
Table C-4: Design Alternative III Map	48
Appendix D: Individual Pairwise Comparisons and Decision Matrix	49
Table D-1: Individual Pairwise Comparisons	49
Table D-2: Average Weights of Pairwise Comparisons	54
Table D-3: Rating of Each Design Alternative Relative to the Criteria	55
Table D-4: Decision Matrix	56
Appendix E: Data Sheets	58
SparkFun Soil Moisture Sensor Datasheet	58
Arduino BT Datasheet	59
Solenoid Datasheets	60
16 Channel 12VRelay Module	61
AC/DC Converter 12V 65W	63
Appendix F: System Diagrams	66
Electrical Schematic of System	66
Level 1 Schematic of System Diagram	67
Software Design for App Diagram	68
Software Design for Microcontroller Diagram	68
Arduino Nano 33 IoT Pinout	69

Abstract

The applications of automation are in increased demand in agriculture and horticulture to reduce water waste and increase plant yields. For gardens hosting a wide variety of plant species that have different moisture requirements, properly meeting these requirements can be tedious. Our solution is to cater to individual plants by monitoring soil moisture levels, future weather forecasts, and watering when deemed necessary by the microcontroller. The system can be remotely controlled and provide feedback to users with current moisture level data for that specific plant. This system will make it much easier to maintain a large and diverse garden. This will both save time in garden maintenance as well offer an alternative to individually monitor moisture levels over different areas in a covered location. This is in contrast to previous solutions such as sprinkler systems with little in the way of customizable control other than timing intervals. Our solution can minimize water waste as well as allow a garden's plants to avoid being overwatered past their individual needs. Preliminary results show that our system can be effective on a small scale shown by our implementation inside the greenhouse of the UIC Plant Research Laboratory.

Project Goals

This section will detail our main goal of the project that we want to have completed by expo. In this section, a general idea of what our project is will be provided.

Our goal is to create a self automated gardening system for the Plant Research Lab on campus. This will be achieved by using two main sources of data, current soil moisture data and future weather conditions. Using the gathered data, the microcontroller will decide whether or not the plants need to be watered. If the microcontroller decides water is needed, it will send a signal to the solenoid to open the valve. The plants will be watered for a finite duration as determined by the microcontroller. The hose will be connected to the PVC pipes at all times so that the garden can be watered as needed. Each section of the garden will have its own system. The system will be placed on ground level to avoid burning plant's leaves and being an eyesore.

Data such as when the plant was last watered, current soil moisture levels of that particular area, and how much water was given will all be able to be accessed through a user interface. An app will be created to host all the information for the user to analyze and ensure the system is functioning properly and the garden is thriving.

Needs Statement

The needs statement describes why our project is important to the Plant Research Lab. In this section we explain why they need our watering system and how it will benefit them from using it.

Currently, the section of the Plant Research Lab being used for the project is watered by hand with hoses or watering cans. On our tour of the Plant Research Lab, it looked as if they had about 80-100 plants in our controlled area. There were many more plants in the ground rather than in pots and troughs. Watering this many plants can take as much as 30 to 45 minutes a day. Some of these plants even have to be watered twice a day in summer.

Our automated watering system will save the Plant Research Lab approximately 1 hour of time each day to work on other tasks such as planting new plants or tilling the soil. An automated watering system can be advantageous for a few reasons such as that the person watering the plants does not need to be there everyday to water the plants. The workers wouldn't

have to come out to campus on days they don't need to, such as weekends or holidays just to water the plants. Our system will deliver a precise amount of water that is perfect for the plant's growth. Inexperienced gardeners may not know how much water each plant needs or how wet the soil is beneath the top 1 to 2 inches. Our organization will address this need as we can deliver a completely automated watering system that takes into account future weather conditions and soil moisture.

Objective Statement

The objective statement describes how we plan for our system will operate once it is fully operational. This is a general outline of how it will work.

The objective of this project is to be able to build an automated watering system for the Plant Research Lab. An Arduino BT microcontroller will be used as it can provide bluetooth connection from a computer in the Plant Research Lab to the garden. The Arduino will receive data from the soil moisture sensor which will be placed approximately one inch below the dirt surface, and weather data from weather.com. If the microcontroller detects that moisture is getting too low in a particular sensor's area, the microcontroller will make the decision to open up a solenoid valve in that section. The plant will receive water anywhere from three to ten seconds depending on the moisture level reading at the time that data was processed in the microcontroller. Once the correct amount of water is given, the system will close the valve and will continue to process information. The pipes will be connected to the hose at all times, so no human interaction is needed. The collected data such as the time the plant was last watered and how much water, will then be displayed to the user on an interface such as an app.

Background

The system will require a 12VDC sealed linear solenoid. A sealed linear solenoid was chosen because it can protect from dust and other small debris such as dirt and sand. The solenoid will be going on top of the outside of the PVC. It will be opening and closing the spout for water to come out of the pipes as determined by the microcontroller. The solenoid has a

stroke length of a quarter inch and is overall less than two inches in size. This will allow the design to look more sleek and have the outlying technology less noticeable.

To track moisture levels in the soil, the system will be using a soil moisture sensor operating on 5VDC. The soil moisture sensor has two prongs, each approximately an inch long that will go into the soil. This is ideal because to measure the moisture content in soil, usually the first half inch to two inches of the soil is the most reliable for indicating if the plant needs water. This moisture sensor is for use with Arduino based devices. It has three screw terminals on top for VCC, GND, and SIG which we will be using to power the moisture sensor and to transmit data back and forth between the moisture sensor and the microcontroller.

This section explains current competitors and any relevant designs that we were able to find on the market.

Competitor Analysis

We don't necessarily have any direct competitors in this project because we have created this for a specific garden. Although, we were able to find relevant systems to our project. The similarities and differences of the systems will be explained.

We were able to find one indirect competitor online whose user interface is similar to what we have. This online article was about an app that can help plant owners know when to water their plant and if their plant is getting enough light. "The app, which costs \$1.99, uses artificial intelligence to analyze users' photos and tells them whether their plant babies aren't getting enough light, too much light, or not enough water [1]". The way this works is that the user sends a picture of their plant, and from the photo the "plant expert" can tell the user several important figures. The plant expert tells the user if the plant looks like it's getting enough light and how wet the soil looks. The difference between our projects and the app is that our system is doing all the work for the user, we are just reporting back the collected data from the microcontroller to the user. That data includes what type of plant was watered, how much water it received, and at what time it was watered. The apps tell the user what to do but the app cannot be exactly sure of the details such as how much water the user gave it, how much sunlight the plant receives or temperature and humidity of the room. This app is more geared towards first time gardeners.

Research Survey

In our search for relevant literature, we searched for information on automated systems. When there are several AGVs, some serious problems may arise in managing them, such as blockings, conflicts, deadlocks and vehicle-collisions. [2, pp.40]. AGV is an abbreviation for Automated Guided Vehicle. This article was about guiding automating vehicle systems. In this excerpt from the text, the author discusses a problem and how they were able to overcome it in the process. The situation at hand was that the author was searching for a way to prevent collisions in autonomous vehicles. He created several diagrams of ladder logic with theorems and proofs to ensure that all areas of the autonomous vehicle were prepared for a collision from any angle.

For our project, we were also talking about preparing a custom survey for each specific garden. In [3], the author talks about how they had to develop a special system for an automated welding system. “For form-fit-and-function testing, a special RP system, including software and hardware, is developed. This system is capable of handling tolerance specifications and material properties [3, pp. 2]”. RP is the abbreviation for Rapid Prototyping. Their system had to be specially designed just for their project. This is similar to the case at hand in the sense that we must create a custom system for the Plant Research Lab. The theory of the automated watering system will be the same no matter where it is located. The layout of where items are placed in the system will alter depending on the location of the garden we are designing the system for.

From the research done, there have been many possible limitations as compared to other similar solutions similar to our automated gardening system. We are limited by multiple factors such as time, funds, and the Chicago winter. As it was stated in the Proceedings of International Conference on Computation, Automation, and Knowledge Management “In order to test the proposed system operability, its prototype was worked out, which uses the following devices at the level of sensors/actuators:[4, pp. 416-420]”. An air temperature sensor, an air humidity sensor, a soil moisture and temperature sensor are just some of the sensors that have been used for their prototype. Thus showing the possibility of limitations due to a lack of amount of information compared to the prototype as discussed in the conference.

Patent US9271454B1 [5] is about a gardening system that can measure moisture in the soil and send information about the plant and the surrounding environment to a cloud service. There is a sensor in the pot directly measuring the moisture content. “The gardener can track this data on a user interface along with other data such as type of plant, soil type, plant size, etc. The data is all connected to a central server. [5]”. This patent is somewhat similar to our idea in the sense that it can track, record, and send relevant data to a user. Our project has an advantage over this idea because our system will be able to water itself. It does not require any additional work from the user.

User Survey

We reached out to Matthew Frazel who is the Plant Research Lab Greenhouse Manager. He gave us a tour of the garden and we had a chance to ask him some questions about the garden. There are about 80 to 100 plants in the garden at the current time. In the section for the system, most plants are in the ground with a few plants in pots and troughs. A variety of plants are grown here such as fruits, vegetables, flowers, trees, and shrubs. We plan on working with several of each type. The trees and shrubs were planted in the ground and covered with a layer of mulch to help seal in water. The fruits, vegetables, and flowers were mainly planted in pots and troughs. The Plant Research Lab has an outdoor garden and a greenhouse but we will just be focusing on the outdoor garden. Matthew explained that the majority of the garden gets full sun year round except for the part of the garden west of the greenhouse since there are so many trees. The plants in that area are mostly shaded. We have some plants in this area west of the greenhouse. He said that the Plant Research Lab would continue to use our system after completion as long as it works as intended for the plants.

Marketing Requirements

Marketing requirements are general requirements for the project in general that we need to create a successful project. These requirements will ensure that the system is efficient and durable, allows us to gather enough data to compare our system to plants not grown in our system, and that the plants will be able to flourish using the automated watering system.

1. System must be able to withstand weather conditions such as heavy winds, strong rain, and extreme temperatures.
2. There must be a reliable water system such as a hose in the garden for the system to function.
3. All electrical components must be protected from liquid, dust, and pests.
4. The system must be durable to withstand damage done by animals or any accidents caused by humans.
5. System must be controlled wirelessly.
6. Garden must have at least 50 outdoor plants.
7. No overhead watering.

Additional Information Collected

I am not sure what to put here since we didn't collect any more data that would apply to the above subsections.

Engineering Requirements (ER)

The engineering requirements are the building requirements for the project. We added specific, trackable numbers to our marketing requirements to ensure that we can keep track of our progress. We are not able to provide justifications from IEEE sources because our garden is made specifically for the Plant Research Lab. We are providing our own standards for the system based on the Chicago climate, pipes we will be working with, and water pressure coming out of the hoses for our system. The left hand column shows what marketing requirement the engineering requirement corresponds to.

Marketing Requirements	Engineering Requirements	Justification
------------------------	--------------------------	---------------

1	Copper wire must be run through PVC to prevent it from cracking due to water freezing from temperatures below 32 degrees Fahrenheit.	Copper is a good conductor and heat traveling along the wire will prevent water from freezing.
2	Water supply must be located within 50 feet of the area where we will be constructing our system.	Don't want to have to purchase more hoses. Keep costs low.
2	Water pressure from the supply must maintain a constant 45 PSI in a hose with diameter of $\frac{5}{8}$ ".	Ensure that we are consistent in our data and consistent with how much water the plants get. Can be tracked using a pressure sensor.
3	Components should be contained in a sealed area or container, which prevents the possible contact from water, water from a hose, possible rain storm, or possible submerging(at least withstand $\frac{1}{2}$ submergence if contained in a case or container) in case of water level rising, from the area it is installed or contained.	With high durability the system becomes reliable for the long term use without any core component replacement.
4	PVC of the system must be able to withstand at least 5 pounds of force in every direction at every part of the PVC in the system.	If a small animal such as a squirrel or bird is to be on it, it must be heavy enough to not collapse under the weight.
5	User interface specifying which plant is being grown, how much water is being	User must be able to access data away from the system

	given to the plant, and at what time the plant was watered.	to ensure that it is functioning properly.
6	Our system needs to be able to water an area of 100 square feet with soil moisture within 5% of intended moisture level.	Our system needs to be reliable in watering a large area for our customer.
7	System must be placed at dirt level to avoid burning of leaves in temperatures above 85 degrees fahrenheit.	Chicago summers can be hot and steamy and can burn plant leaves if watered from above thus killing them.

Engineering Requirement Updates and Changes

We had to update a few requirements because we changed the scope of our project by setting up our system indoors. We will discuss the outcome of engineering requirements that were either changed or we were not able to complete.

The first requirement stating that copper wire must be run through PVC to prevent it from cracking due to water freezing from temperatures below 32 degrees Fahrenheit. This requirement is no longer applicable since we moved our system inside the greenhouse. We do not have a substitute for it.

Another engineering requirement that changed was the one stating that the water pressure from the supply must maintain a constant 45 PSI in a hose with diameter of $\frac{5}{8}$ ". This requirement could not be met because in order to measure water pressure we need a pressure meter. A pressure meter put us over budget.

Another engineering requirement states that the PVC of the system must be able to withstand at least 5 pounds of force in every direction at every part of the PVC in the system. We tested the strength of the PVC by first applying PVC cement to bond the pieces together. After it bonded, we tried to take apart the pieces using our strength and the PVC did not budge.

This next engineering requirement was partially met. It states that the user interface should specify which plant is being grown, how much water is being given to the plant, and at what time the plant was watered. In our completed app we show what plant is being grown. Instead of showing how much water is being given to the plant and at what time it was watered, we now show which solenoid waters each plant and current soil moisture sensor readings of each plant.

The last engineering requirement that we changed stated that our system needs to be able to water an area of 100 square feet with soil moisture within 5% of intended moisture level. This requirement could be met if we were planning to water a garden of that size, but our garden was not 100 square feet. 5% of intended moisture level was achieved when calibrating the soil moisture sensors.

Engineering Design Alternatives

In this section we introduce our Level 0 Schematics for each design alternative we have created. We list the basic inputs and outputs of the system for each design alternative. We also discuss the functionality of each system with those particular components. This will serve as a guide for future reference when we are determining what design alternative works best for our system.

There are a few parts of our project that will stay the same in all the design alternatives. In the future weather forecasts, we will be inputting the chance percentage of rain for that day. On the app, there will be an interactive user interface. The user will be able to input what plant data they want to interact with. That data will include last time the plant was watered, how much water was given, and type of plant it is. These factors will remain constant in any design alternative chosen.

We have several factors that we will focus on such as time needed to complete, cost, and ease of product use among others that can be found in the [Design Alternative Evaluation Criteria](#). The three factors mentioned before are some of the most important to us. For our power system, we firmly believe that power from the greenhouse outlet is the best way to go compared to using a solar panel or battery. This way we know that we are getting constant power from the outlet. It

will also be the cheapest option. For our piping system, we know that rubber hoses are the cheapest, but they are not as durable as PVC piping. Durability is key since the system will be outside. We are not as concerned with which microcontroller we use since they are all relatively the same price with the same functionality. We are also not too concerned about which type of sensor we use since they all have the same functionality. For the user interacting with data, we believe that the app will be easiest since most people have a smartphone and they can check it on the go. Displaying output directly on the enclosure isn't ideal because the user would have to be at the garden to check data. We believe that an app is more straightforward to use than a website.

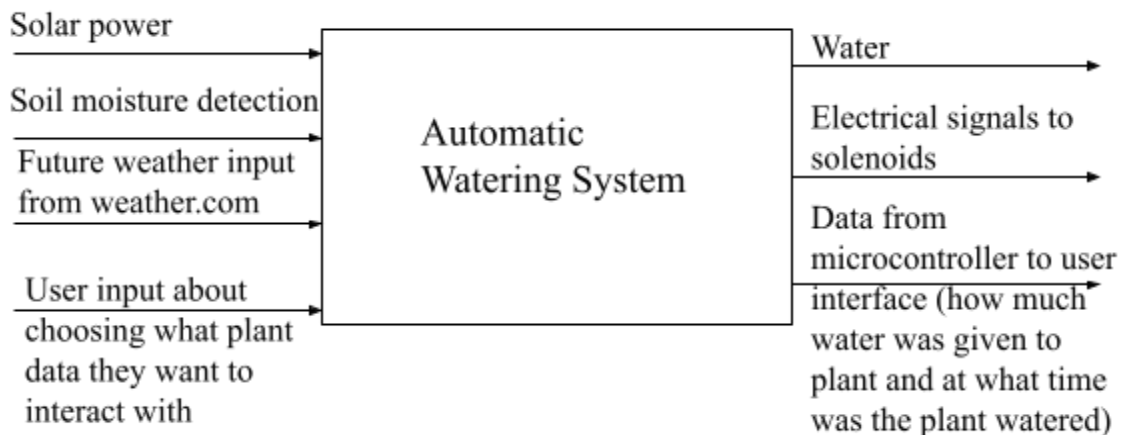
Design Alternative I

Table 1: Input/Output Design for Design 1

Module	Automatic Watering System
Inputs	Solar power, soil moisture detection, future weather forecasts (probability chance of rain for that day provided by weather.com), and water source (water from outdoor hose faucet). Inputs from user interface when user chooses to interact with app data such as when the plant was last watered, how much water was given, and soil moisture levels of that plant. These inputs from the user interface have no effect on the system's functionality.

Outputs	Water to the plants, data from microcontroller to user interface (how much water was given to plant, soil moisture levels of that plant and at what time was the plant watered), Arduino sends signals to solenoids to open or close valves depending on current action.
Functionality	Power generated from solar panel will be stored in a battery and available to use for the system. The system will then use that stored power to power the microcontroller which will send signals to water the plants based on current soil moisture levels and future weather conditions. The generated solar power will also power the solenoid which is responsible for releasing water to the plants.

Figure 1: Level 0 Schematic for Design 1

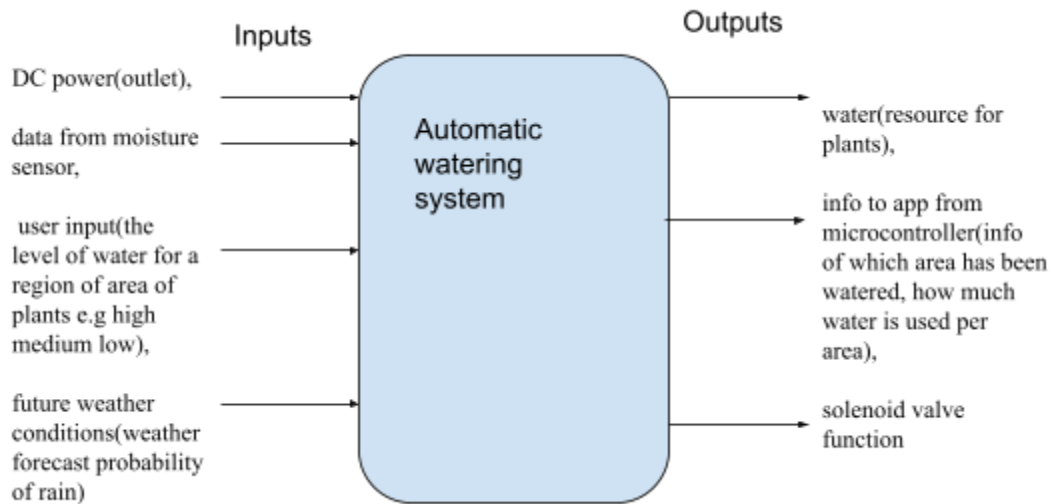


Design Alternative II

Table II: Input/Output Design for Design II

Module	Automatic Watering System
Inputs	DC power(outlet) , data from moisture sensor, user input(the level of water for a region of area of plants e.g high medium low), future weather conditions(weather forecast probability of rain), water source (hoses from outdoor hose faucet)
Outputs	water(resource for plants), info to app from microcontroller(info of which area has been watered, how much water is used per area), solenoid valve function
Functionality	The system efficiently waters the regions of plants based on probability of rain in the future, and on the amount of water currently within the region of plants(e.g water from outside source like gardner)

Figure 2: Level 0 Schematic for Design II



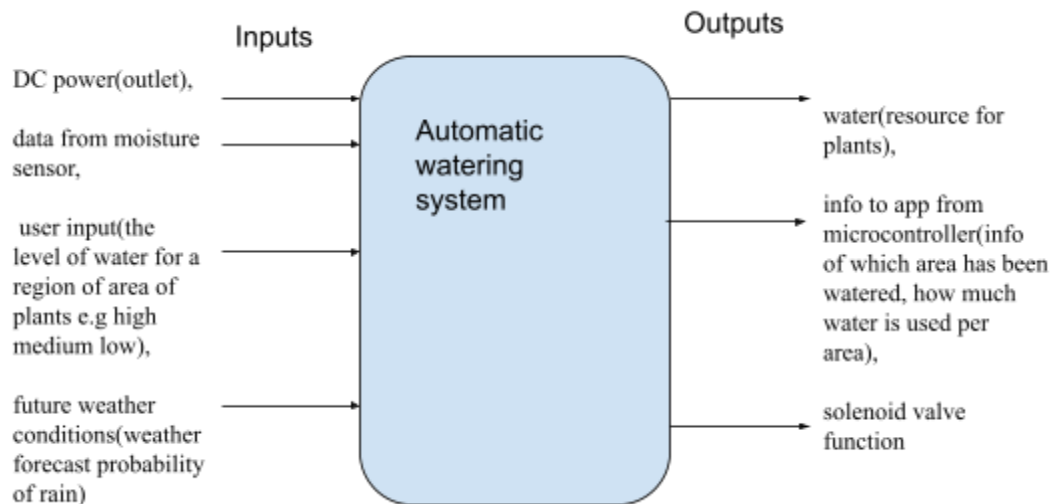
Design Alternative III

Table III: Input/Output Design for Design III

Module	Automatic Watering System
Inputs	DC power(outlet) from greenhouse, data from moisture sensor, user input from the app(the level of water for a region of area of plants e.g high medium low), future weather conditions(weather forecast probability of rain), water source(water from outdoor hose faucet)
Outputs	water(resource for plants), info to app from microcontroller(info of which area has been watered, how much water is used per area), linear solenoid valve function
Functionality	The system efficiently waters the regions of

	plants based on probability of rain in the future, and on the amount of water currently within the region of plants(e.g water from outside source as in gardener)
--	---

Figure 3: Level 0 Schematic for Design III



Engineering Design Alternative Updates

One thing we did not address during the fall semester was the creation of the server to transfer data between the UI and the arduino code. This server allowed a text string to be sent from the UI to the app and app to the UI. This string consisted of data such as future weather conditions, current soil moisture of plant, and solenoid status. The data was used in different ways as it depended on what decision the UI or the arduino was doing at that moment.

We believe all the rest was covered in the fall semester. We kept the same inputs and outputs to the system. Our engineering design alternatives basically all followed the same I/O scheme so we didn't have to make any major changes or updates to it during the spring semester besides the server.

Design Alternative Evaluation Criteria

In this section, we detail why we chose a certain evaluation criteria and why we believe it's important to our project. We believe these factors will steer our project towards success. These are the criteria that we will be using in our individual pairwise comparisons and the rating of each design alternative relative to the criteria.

1. Ease of Product Use

It is important for us that the user is able to interact with the provided data fairly easily. The user will be able to interact with recorded measurements such as plant watered, time the plant was watered, and how much water was given.

2. Development Cost

We are constrained in what we can accomplish by the value of our budget. It is extremely important that our design meets all of our engineering requirements while not going over budget. If we were to sell this to consumers, an elevated development cost would mean the final product would cost more to the consumer.

3. Suitability for Demo 397

We want to be able to present our product at demo. We plan on bringing a small, functioning prototype that would show how our product actually works in the garden. It is important that we are able to complete our project on time to show that we are able to meet deadlines and deliver on a final product.

4. Time Needed to Complete Design and Demo

Time management is key in a project where we have multiple deadlines to meet and other classwork to complete. We must manage our time to ensure that we are able to have a finished project for the expo.

5. How Attractive This Design Method Will Look on the Resume

It is important that this project shows what we learned and how we were able to apply those skills to a project. In our case, we are able to use our engineering skills to help an area that doesn't usually involve any technology, gardening. We want to show that we are more than prepared for any type of potential future projects at a workplace.

6. Ease of Satisfying the Previously Defined Technical Specifications

Our system is meant to do certain things, and we want to make sure that our system is able to complete those tasks in a timely, efficient, and safe manner. We want to be able to complete everything that we originally set out to do.

Design Alternative Evaluation Criteria Updates

After completing the project we still believe that these were the best Design Alternative Evaluation Criteria to base our project on. It is important that the project had a low cost and was easy to use for consumers because otherwise no one would want to purchase something that's expensive and is hard to use. It was also important that the project was suitable for 397 Demo because we wanted it to seem like a challenging project that is something we could talk about to our future employers.

Selection of Design Alternative & Justification

Design Matrices

We first decided what criteria was most important to us and our project as a whole. First, we individually rated each aspect of the project against another aspect. This allowed us to ultimately determine what were the most important factors for us in our project. We then combined our data to create the [Average Ratings Matrix](#). As a team, we rated the design alternatives and scored them according to our Average Ratings Matrix. This data can be found in the [Decision Matrix](#). This is how we were able to determine what was the best alternative for us to go with in this project. The figure seen below is the decision matrix.

Scores for alternatives	Alternative I	Alternative II	Alternative III

Ease of product use	0.33	5	3	3
Development cost	0.16	5	3	3
Suitability for demo for 397	0.21	3	3	3
Time needed to complete	0.15	5	5	3
Resume attractiveness	0.08	5	5	3
Ease of completing the technical specifications	0.07	5	5	5
Score		4.58	3.60	3.14

Final Decision

We decided that design alternative 1 was the best design alternative for us to go forward with. We believe that this would be the easiest to use since we will be programming on an Arduino that will send data to the app via wireless connection. It's also easiest to have our system's power supply connected to the greenhouse outlet. We would be creating another project within itself by introducing a solar panel in our project, which ultimately would increase our overall cost and would force us to allocate less time to other aspects of the project. We believe that design alternative 1 is also the cheapest option available. This is because we determined that the PVC piping would be cheaper than metal piping, an app is cheaper to create than a website, and the Arduino seems the least costly of the three microcontrollers we had to choose from.

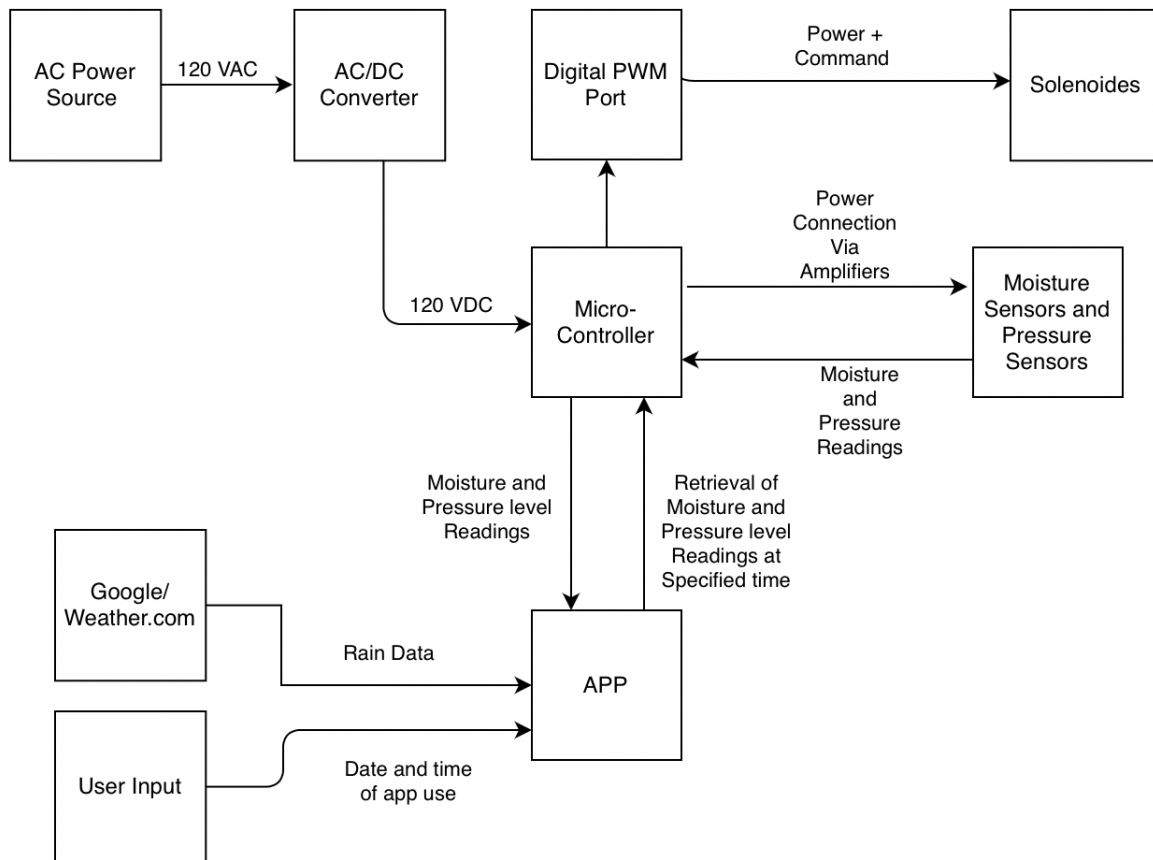
Design Alternatives Justification Update

We did not change any justifications or add any new justifications. We still think what we stated in the fall semester is what was best for our project. It made the most sense because it seemed like it would take the least amount of time, be the cheapest build, and provide the most reliable and efficient system structure for our project.

Preliminary Design

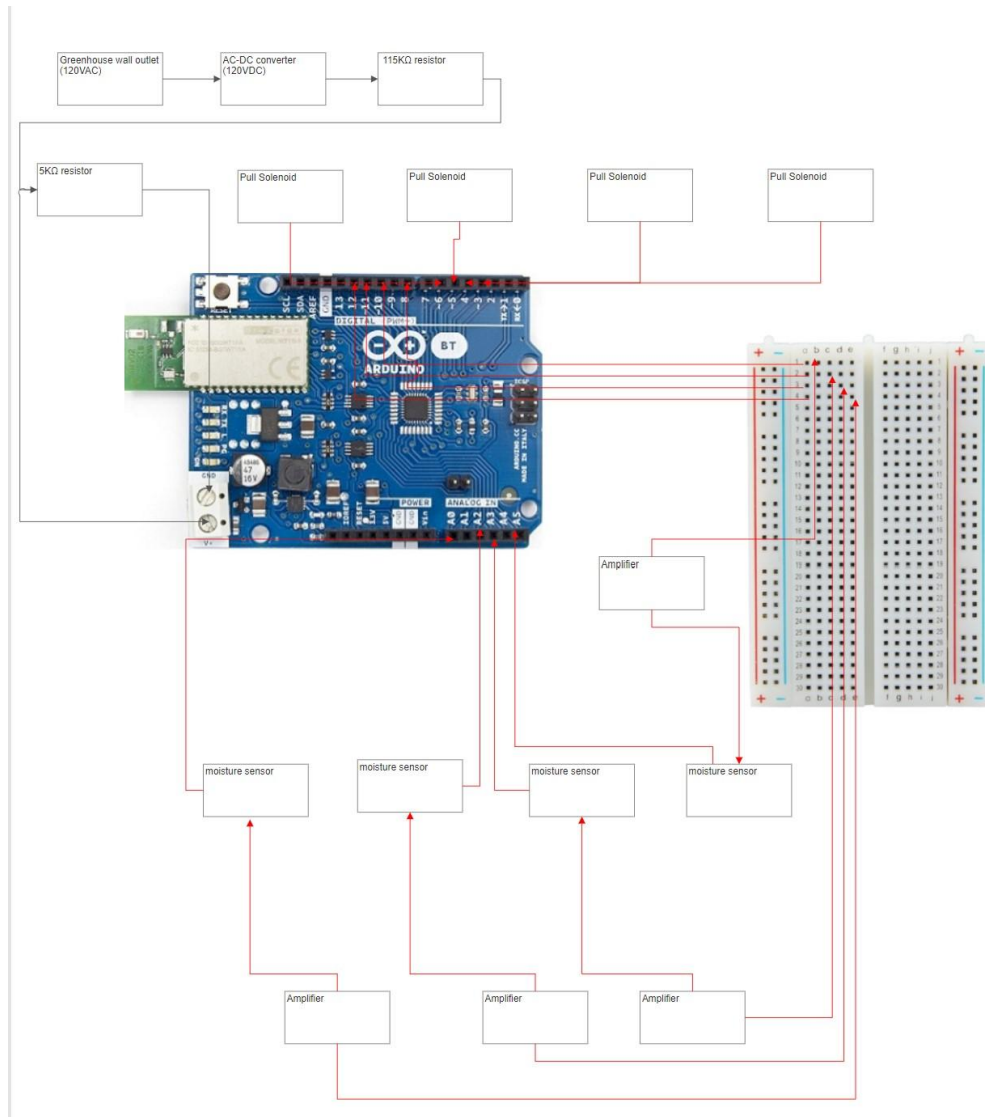
Level 1 Schematic of System

The level 1 schematic outlines how we plan for the entire system to work by integrating parts of our circuit design and software design. Each section for [circuit design](#) and [software design](#) is explained in detail in their respective sections.



Circuit Design

This is the circuit design that we are planning for our system. We are receiving 120VAC power from the inside of the greenhouse. The power will then travel to an AC-DC converter to convert the voltage to 120VDC. The current then goes through two resistors. The 115k Ω resistor leads to the V+ pin on the Arduino BT. The 5k Ω resistor leads to ground. There are four linear solenoids connected to the Digital PWM ports of the Arduino BT, each solenoid on a separate pin. There are four soil moisture sensors connected to the Analog inputs of the Arduino BT. The soil moisture sensors connect to the pins of the Arduino through amplifiers.



Software Design

The software design is split up into two separate sections, app design and microcontroller design.

The app is completely reliant on the user. The app will be receiving text data from the microcontroller such as the time the plant was watered and what type of plant was watered. It will then display this data to the user with the information being updated whenever the user inputs that command. We will also be sending weather data to the app to use in calculations for the microcontroller to decide whether or not to water the plants. The app will send this to the microcontroller over a bluetooth connection. We will be using weather data from weather.com.

The microcontroller we are using will be an Arduino BT microcontroller. That will be our main data manager which is in charge of accessing, collecting, and analyzing the data to determine what is the next step for the system to take. The microcontroller will be continuously receiving data from the moisture sensors and the weather information stating what the chance of rain is for the Plant Research Lab. The microcontroller will be waiting for the user to refresh the data to send data to the app. After making the decision, the Arduino can either send info to the app or send a signal to the solenoids to release water to the desired location. Each moisture sensor will be in charge of a section of garden that has similar factors such as ratio of sun to shade and what type of plants are there.

Software Design for App

Applications of some functions:

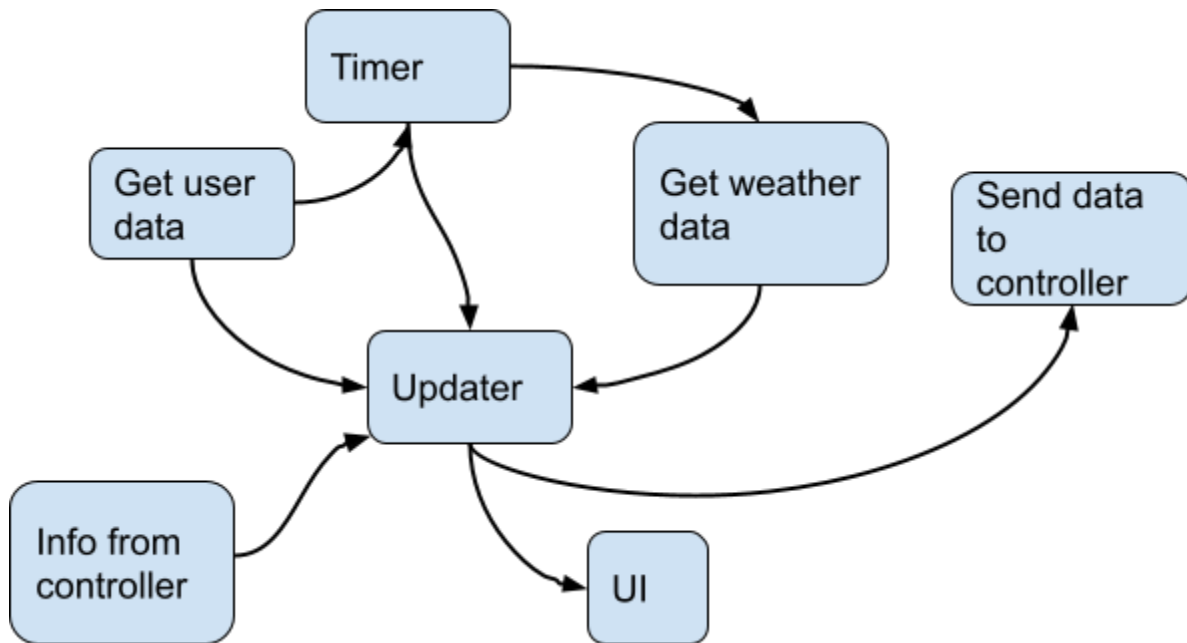
Send data to controller(): this will handle the communication and data transfer between the app and the microcontroller, for example data from the weather receive from the updater, or any information that has been changed or modified by the user like plant information like water consumption, another example is if the user want to control any of the solenoids from the app.

Get Weather data(): this will be the function that will obtain any necessary information from the weather like temperature or humidity or the probability of rain as an example. Once all information is obtained and formed into a txt file and from the updater it will both update the UI and sent via bluetooth to the controller

Timer(): the timer will keep track of time for various cases;

case 1: default for get weather data will be for 7 days or weekly meaning every week the app will get data regarding the weather (user can change the time period to their own preference like from 7 days to 3)

Case 2: would refer more towards the updater focusing on the time period for the updater will update any specific object



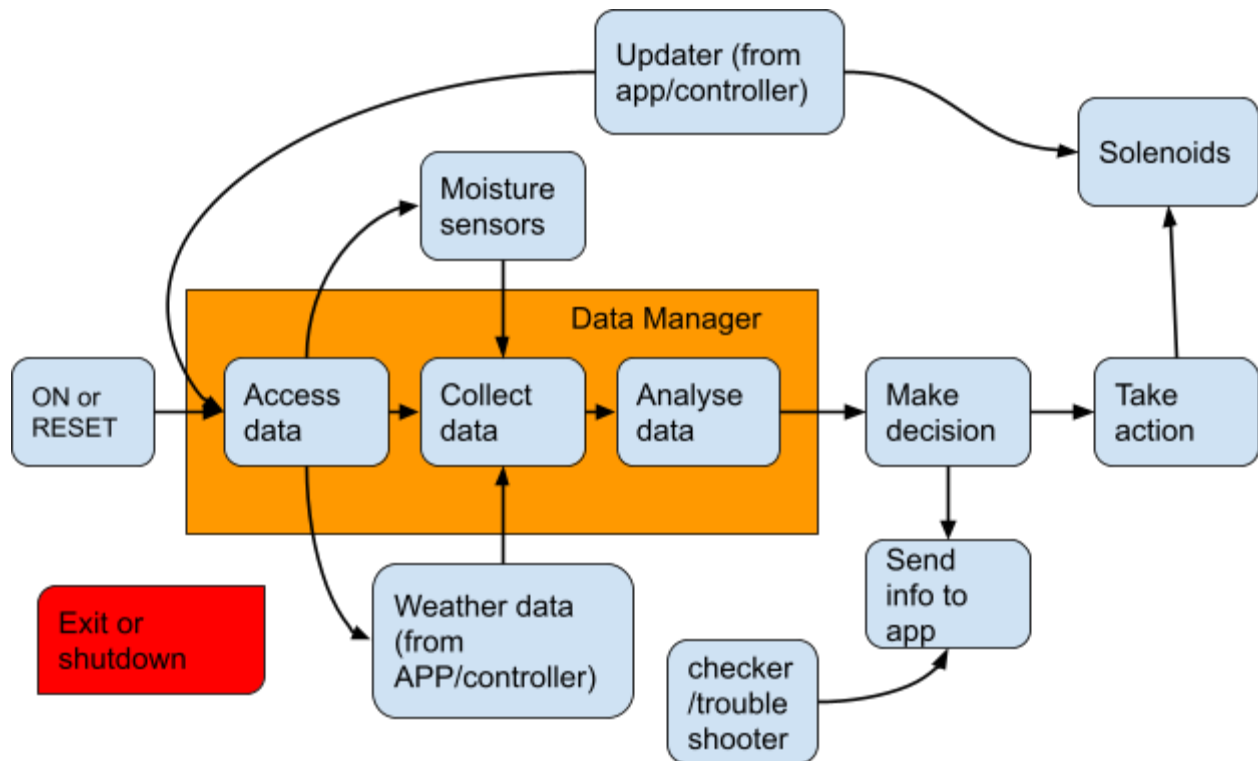
Software Design for Microcontroller

Application of some functions:

Make decision(): this function will be the one to decide whether the controller will need to water any specific area given the data from the data manager; for example if there is a 50% chance that a light rain shower will happen for a couple of minutes the function will most likely water the area; on the other hand if it's 50% of a rainstorm for a couple of minutes then most likely it will not water the area.

Take action(): this function will focus on the actual mechanical side of things it will handle the implementation of the plan(decision for make decision()) with the solenoids

Checker /Troubleshooting(): this function will make sure that everything is working properly as intended for example if a solenoid is supposed to be turned on or open, but it is not then it will send a warning message to the app to notify the user.



Preliminary Design Review

We did not make any major changes to the preliminary design that we created in 396. The only thing we changed out of the software and circuit design were using a different combination of values for the resistors in the circuit design. Everything else has followed from our design before. We had a good idea of what we were expecting everything to look like so that we were prepared for the start of 397.

Testing

We separated testing into software and hardware components. For the software component, we tested how the UI and Arduino interacted with each other. For the hardware component, we tested the solenoids and soil moisture sensor. Each section of the testing will be explained below.

Hardware Testing

We first tested and calibrated the soil moisture sensor. We used sample code provided by the manufacturer to perform calibration. We first tested the soil moisture sensor in air to see the value received. After confirming that value, we tested it in waters of different temperatures to make sure we would get the same value. Then we tested the soil moisture sensor in the dirt without water and then with water to make sure it would work. We calibrated the sensor to know that soil that was just watered to the sufficient level is the base value.

Testing Steps:

1. Connect soil moisture sensor to the manufacturer's code in the Arduino IDE
2. Gather value of soil moisture sensor in air
3. Gather value of soil moisture sensor in room temperature water
4. Gather value of soil moisture sensor in hot water
5. Gather value of soil moisture sensor in cold water
6. Gather value of soil moisture sensor in dry dirt
7. Gather value of soil moisture sensor in dirt that has been watered to the sufficient level
8. Complete calibration of testing by setting value acquired in Step 7 as the base value

Variables Tested:

1. Soil moisture sensor readings in different mediums
2. Soil moisture sensor connection

Results:

We performed this testing in the Plant Research Lab greenhouse as this was where the system would be going. By testing in this environment, we knew our calibration would be as accurate as possible.

We then tested the solenoids using our Arduino code that we wrote. We checked to make sure that each solenoid opened and closed without any running water. We then connected the solenoids in our system with the PVC piping to test with water running through the solenoids. The solenoids held up to the test with water. There was not much testing to do for the solenoids because they are either on or off.

Testing Steps:

1. Connect solenoids to the previously written code in the Arduino IDE

2. Open and close solenoid with air as the medium
3. Connect solenoids to system and turn on water
4. Open and close solenoids every three seconds using the code in the Arduino IDE

Variables Tested:

1. Solenoid connection
2. Solenoid function when opening and closing

Results:

The unit testing for the solenoids went smoothly so we were able to connect it to our system right away. We were comfortable with the results.

Software Testing

Our software testing consisted of testing the Arduino code, UI code and server code. We performed periodic testing on the Arduino code and the UI code as we were building it to make sure that our functions were working and had no errors. The UI was easier to test because we could see what the product looked like as soon as we compiled the code. We created the server code in the Arduino IDE. The Arduino and UI had to be connected over the same WiFi network to work. We tested sending text packets from the Arduino to the UI to ensure data was getting sent. After that worked, we put in the values that we wanted to send from the Arduino code that controlled the solenoids. We also tested sending information from the UI to the Arduino such as weather data code so the Arduino could decide whether or not to water the plants.

Testing Steps:

1. Write Arduino code and test after writing every 10 lines
2. Write UI code and compile after every 10 lines to see what the UI is looking like
3. When Arduino and UI are complete, write server code to connect the two
4. Connect Arduino to the same WiFi network that the computer running the UI is on
5. Send a text packet with 10 characters of data from the Arduino to the UI and make sure packet is received
6. Open and close solenoids in the Solenoid Tab on the UI to ensure that that data is being sent to the Arduino through the server. Server should say “packet received”
7. Add final variables to be sent through the server in the Arduino and UI code

Variables Tested:

1. Server connection functioning correctly
2. UI code
3. Arduino code

Results:

The server, Arduino, and UI code worked as we intended it to work together. Data was correctly sent and received as we instructed the code to do in certain situations.

Comparing Engineering Requirements and Testing Results

Engineering Requirement	Was it met?	Result
Copper wire must be run through PVC to prevent it from cracking due to water freezing from temperatures below 32 degrees Fahrenheit.	No	We determined this wasn't useful because our system got moved inside the greenhouse where temperatures are always above freezing.
Water supply must be located within 50 feet of the area where we will be constructing our system.	Yes	This was met but it was not as a result of testing. System placement was not included in testing.
Water pressure from the supply must maintain a constant 45 PSI in a hose with diameter of $\frac{5}{8}$ ".	No	We could not test this requirement since a pressure sensor was out of our budget.
Components should be contained in a sealed area or container, which prevents the possible contact from water, water from a hose, possible	Yes	We only tested the soil moisture sensors in different environments because only the soil moisture sensors would be affected by the

rain storm , or possible submerging(at least withstand $\frac{1}{2}$ submergence if contained in a case or container) in case of water level rising, from the area it is installed or contained		water of the hose. The entire was system was moved inside so we were not worried about the rest of the components.
PVC of the system must be able to withstand at least 5 pounds of force in every direction at every part of the PVC in the system.	Yes	After applying PVC cement to strengthen bonds between PVC connections we tried to take apart the connections by hand using our full strength. The bonds did not break.
User interface specifying which plant is being grown, how much water is being given to the plant, and at what time the plant was watered.	Yes	This was satisfied in testing when we created the UI and Arduino code.
Our system needs to be able to water an area of 100 square feet with soil moisture within 5% of intended moisture level.	Yes	We did not have 100 square feet to water, but we could easily have completed it if we had the area. 5% intended moisture level was met when we calibrated the soil moisture sensors.
System must be placed at dirt level to avoid burning of leaves in temperatures above	No	This requirement is no longer applicable since our system was moved indoors. Our area

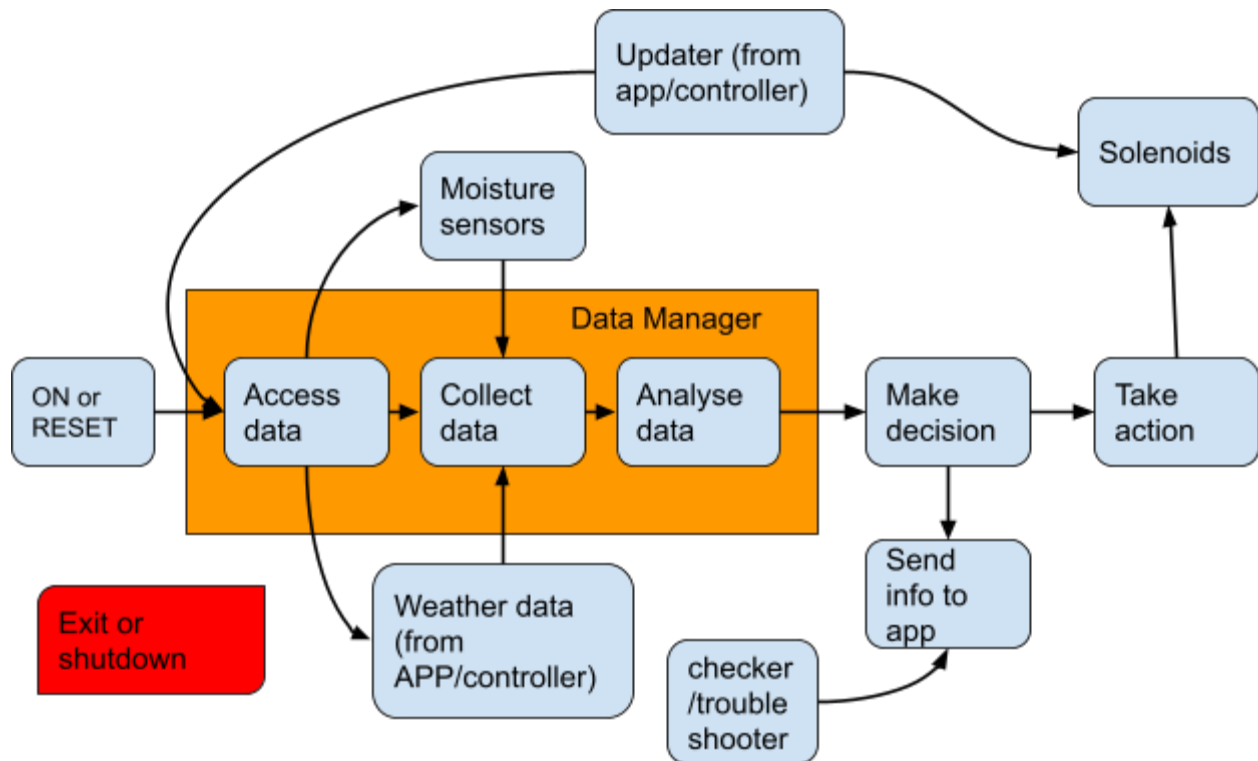
85 degrees fahrenheit.		in the greenhouse had a little shade so plants were not receiving direct light which could cause harm.
------------------------	--	--

Final Design

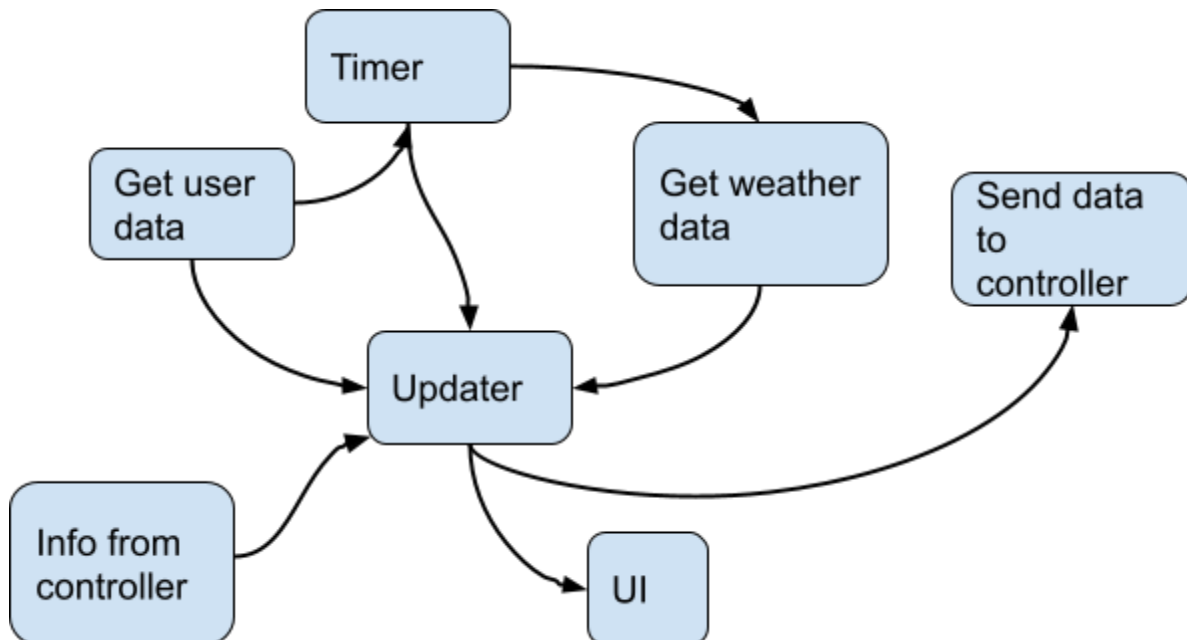
In this section we will explain about how we went about our final design and include all supporting diagrams. Our final design differed from our preliminary design as we first designed our system to be outdoors and span a range of about 100 square feet, this was then changed to an area of three plants in pots inside a greenhouse due to the ground not thawing in time for us to set up and test our system. This change in scenery caused us to have to drastically scale down our system. Our system went from fifty feet of PVC to about five feet of PVC. This change in area also caused us to use less solenoids with the inside version of our system only requiring three compared to the four solenoids we would have used outside. The code of our system was unchanged and still took into account user input, future weather conditions, and moisture levels of the soil to make a decision on watering the plants or not. We no longer required a waterproof container to house our electronics so we changed our casing to a cardboard box to protect our electronics from debris. Our inside system used a soft hose from a water spigot to transfer water into our main apparatus which contained three branches of PVC, each controlled by its own solenoid connected to the arduino, to the three plants. All in all, the only changes to our system, caused by us moving our system inside, are the housing of our electronics and the physical size of our system.

Circuit Design

For the circuit design we had to create a schematic for the system which can be found below and in the [Appendix](#). We had power coming from the wall going to an AC/DC converter. Once converted, the power went to the Arduino and the channel relays. The channel relays connected to the solenoids to open and close them. The channel relay then connected to the



Our software design for the app did not change either from 396. We kept the same format that we originally designed. This can also be found in the [Appendix](#).



Bill of Materials

This section displays what we purchased, quantity of each part, and cost per part. We had a budget of \$250.

Name of Part	Quantity	Unit Cost	Total Cost
1/2" 12VDC Electric Plastic Solenoid Valve	4	\$16.99	\$67.96
16 Channel 12V relay module	1	\$14.99	\$14.99
ARDUINO UNO WiFi REV2	1	\$20.70	\$20.70
AC/DC CONVERTER 12V 65W	1	\$29.85	\$29.85
SEN-13637	4	\$6.95	\$27.80
1/2 PVC sch40 coupling	6	\$0.63	\$3.78
10pck 1/2 PVC sch40 connectors	1	\$14.99	\$14.99
1/2 Vinyl OD Tubing 20ft	1	\$7.96	\$7.96
1/2 PVC sch40 10 ft	3	\$4.96	\$14.88
1/2 in PVC sch40 male spigot	6	\$0.87	\$5.22
Final Cost			\$208.13

Task Allocation and Timeline

Week-by-week timeline

This table shows what we had planned for the semester. There are estimated start and end dates which we roughly followed to keep ourselves on track. We gave each group member a chance to lead a task. Each task is explained in detail. The software tasks were the only ones that

we had to prolong quite a bit because we underestimated the scope and intensity of the work that came with it.

Start Date:	Finish Date:	Weeks to Complete:	Start Week:	Lead Participant:	Task:	Explanation:
1/10/22	3/4/22	8	1	K	Order parts	Order all necessary parts for the project and any parts we decide we need as we continue the project
1/17/22	2/4/22	3	2	A	Program Arduino	Program arduino to control solenoids, sensors when necessary
2/7/22	2/11/22	1	5	A	Test Arduino	Test to verify all functions and devices work as intended
2/14/22	2/25/22	2	6	A	Create user interface	Program UI to show and function as according to the specs of the applications' functionality
2/28/22	3/4/22	1	8	A	Test user interface	Test UI for any bugs, and making sure it functions smoothly
3/7/22	3/18/22	2	9	J	Install outdoor system	Implement design plan on the 3 areas where the system will be used and tested

3/21/22	3/25/22		Spring Break		None	None
3/28/22	4/1/22	1	12	J	Have outdoor system and all electronic systems connected	Full implementation of the entire system, making sure it is all connected as designed
4/4/22	4/8/22	1	13	S	Final testing of entire system	Test for any final bugs between the hardware and software functionality
4/11/22	4/15/22	1	14		Create setup for expo	Build an appropriate setup for the expo presentation
4/18/22			EXPO	ALL	EXPO	EXPO

Contribution of Work

Joe Biscan - I helped contribute to the software design by writing the weather data code and assisting Anthony with the server and UI construction. I also helped set up the physical system such as wiring and laying out parts. Helped with making decisions and changes for the project.

Kyle Hansen - I helped code the microcontroller, wire the physical system, create the electrical schematic, connect the arduino to the wifi, make decisions on what parts to order, make decisions on design changes to our system, and helped with the physical layout of our system parts and how to get them all connected.

Anthony Ibarra - As of now the total contribution as far as I know are, designing the software, building the application, creating the server in the application, coding the microcontroller and making the server in the microcontroller. Assisted in building and constructing the system for the integration of the project. Also part of decision making and finding solutions to problems that have shown along the way.

Silas Chakravary -

Lessons Learned

Joe Biscan - It is extremely important to start the task with sufficient time available instead of waiting until the last minute. If we did not complete a certain task by our due date it basically messed up the rest of the schedule by pushing everything further back. It's easy to get stuck doing things at the last minute and then having it pile up in the end. It's also important to have good communication with group members to make sure everyone is on the same page, organized, and kept accountable. This project also showed me that it's useful to keep an open mind about what you're working on because changes occur often and quickly such as when we had to move our project into the greenhouse with only two months to finish. Senior design has prepared me to work effectively in a group environment.

Kyle Hansen - One of the most important takeaways from this project is that each individual task must be started and completed in a timely manner. If this does not occur the project will start to snowball and before you know it you are out of time and rushing everything at the last minute. Another big takeaway from this project is that communication and holding yourself and your teammates accountable for their tasks is key in creating a successful end product for the user. If one person is slacking and no communication is happening then the project will fall flat on its face and the target date of the project will not be met.

Anthony Ibarra - Even though what one might end up working on might not be what I hoped in this case plants, there was a lot to take from the project itself. The process of communicating between team members and knowing what someone is and how far they are at it, or if there are

any changes made. As a team information is extremely important, without it then the team is just lost without advancing in the right direction and ends up as a train wreck.

Silas Chakravarty -

Conclusions

The system that we constructed would work well on a small scale. We know that if needed, we could scale the system to any size needed to fit a wide variety of consumers' needs by adding more sensors and solenoids. The Arduino and UI communicated effectively on orders for the system to execute. The software for the project wouldn't change much depending on the size. We would basically have to copy and paste previous code and change variable names for each additional solenoid added to the garden.

The cost of the system would obviously depend on the size of the garden it is designed for. For the project we created, we recommend a starting price of about \$500.

We thought of a few improvements for the future. If we were designing it for a larger scale we would need to create a mobile app instead of a desktop app. A mobile app would just be easier for the user since most people always have their phone on them unlike a computer. We would also use PVC piping throughout the whole system to have more secure connections. For plants with deep roots, we may have to get a sensor that has sufficient range to work effectively for that plant. Our sensor now only had a range of six inches. We would need to find a more secure way to water the plants instead of just tying the hose down. We could do this by creating the whole system at plant height. Casing for the wires such as conduit would be helpful over long distances to prevent water from interacting with the wires.

We believe that our system was not able to work because of the outlet that we plugged our system into. The outlet had water dripping onto it when it rained outside and this was not something that we were aware of until we blew a fuse in our AC/DC converter. We believe that everything else was set up right. We checked our schematic multiple times doing circuit analysis to make sure all the values were correct. We wired our system exactly as we had laid out in the schematic. The fuse blew so that means it couldn't have been something in the system we constructed.

References

- [1] McHugh, Molly. (2019, May 23). Let It Grow: The Applification of Plants Is Helping Owners Keep Them Alive
- [2] J. Luo; H. Ni; M. Zhou. (2015). Control Program Design for Automated Guided Vehicle Systems via Petri Nets, pp. 44
- [3] Zhang, Yu Ming; Li, Pengjiu; Chen, Yiwei; Male, Alan T.. (2002). Automated system for welding-based rapid prototyping
- [4] A. S. Mohammed, O. Barkovska, S. T. Suganthi, D. Rosinsky, and B. Oleg “Automated Gardening System with Frost Prediction for Small Land Parcels,” 2020, pp. 416-420, doi: 10.1109/ICCKAM46823.2020.9051541
- [5] A. Natanzon, L. Natanzon, V. Roginsky and J. Shochat, "Intelligent gardening system and method," US9271454B1, March 1, 2016, pp. 2

Appendices

Appendix A: Final User Manual

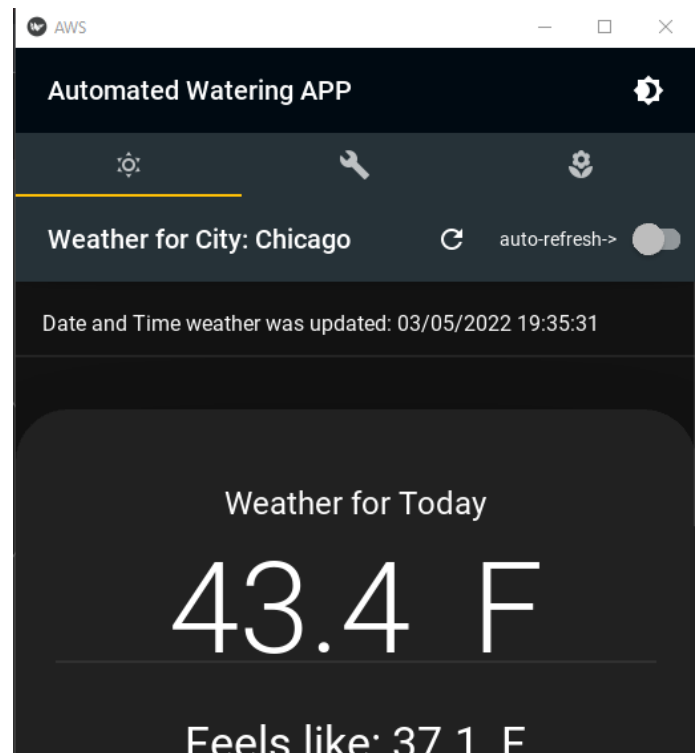
APP Manual

There are three tabs in total for the app, the weather tab, system tab, and plant tab.

Weather Tab: the first tab from the left will show the user the weather of the set location Chicago;

The weather tab will show the date and time that the weather information has been updated. The current temperature, what it feels like, the precipitation, Humidity, Pressure, Wind speed and its direction, along with cloud coverage.

You can refresh the tab by pressing the first button that look like an arrow moving counterclockwise,

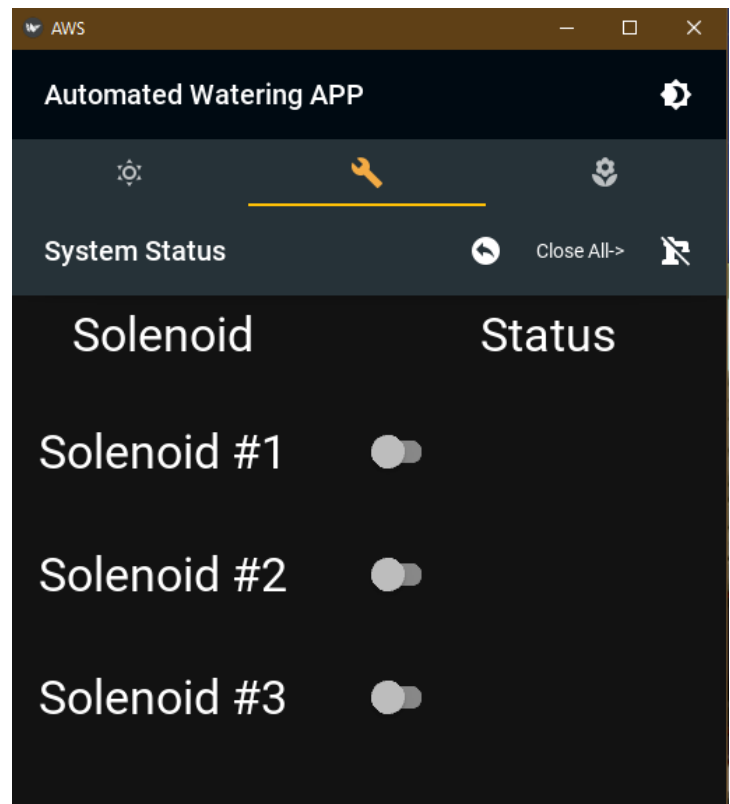


and beside that you can turn either the auto refresh on or off(refreshes every 30 minutes).

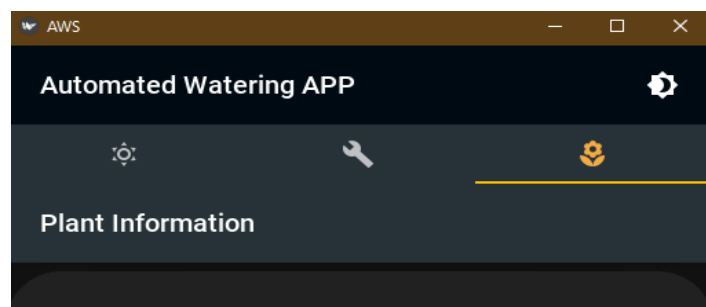
System Tab: In the beginning the 3 switches corresponding for each solenoid will be unavailable for the user until the control has been return to the device(microcontroller arduino), to return control press the button on the left side of Close ALL->

The user will now be able to see the status of each solenoid. Once the user presses either Close All->(crossed out pipe) or a switch below the status and beside the solenoids.

It is recommended that the user returns control every time the user has finished changing the solenoid's states.



Plant Tab: The plant information tab will show the user the plants that the system is



currently watering, which solenoid is doing the watering, and what is the current moisture level for that plant.

System Manual

- I. Installation
 - A. Download provided app/website on your phone or computer
 - B. Open the application
- II. Powering the Microcontroller
 - A. Plug power cord into a 120VAC outlet (standard wall outlet)
- III. Pairing the Controller to your phone or computer using Wifi
 - A. Turn on your mobile phone or computer and power on the system
 - B. Connect to the desired Wifi network on your device
 - C. Connect the Arduino to the same Wifi network as your device by accessing the Arduino IDE website
- IV. App/Website specifications
 - A. Viewing current weather data
 1. Scroll up and down on Weather Tab to view current weather conditions of the Plant Research Lab
 - B. Viewing current soil moisture level of each plant
 1. Navigate to the Plant Tab
 2. Scroll down beneath the picture to view the current soil moisture level of each plant. Also shows the solenoid watering the corresponding plant
 - C. Access control toward solenoids
 1. Navigate to the Solenoid Tab
 2. Choose solenoid you want to access and change accordingly
 - D. Updating microcontroller, weather from app
 1. Navigate to the Weather Tab

2. In the row that says “Weather Info for City: Chicago”, look to the right and there will be a refresh button. The refresh button is an arrow going in a clockwise direction with an arrow pointing
3. Click the refresh button

V. Shutting Down

- A. To power off the system, remove the power cord from the outlet
- B. Exit the application

Appendix B: Team Task Management Document

Team Info

Team Name	Automated Watering
List Members Here	
1	Joe Biscan
2	Silas Chakravarty
3	Kyle Hansen
4	Anthony Ibarra

A	B	C	D	E
Team Name	Automated Watering			
Week	2			
Manager	Joe Biscan			
Topic:				
Task #	Description	Deliverable	Assigned To	Status
1	Order all electronic parts for project w/Silas	BOM for electronic parts	Kyle Hansen	Complete
2	Have outdoor blueprint design completed	Will have a finished design done that so we	Joe Biscan	Partially Complete
3	Start brainstorming for app	Have an outline completed for the app with	Anthony Ibarra	Complete
4	Order all non electronic parts for project	BOM for non electronic parts	Silas Chakra	Complete

Team Name	Automated Watering			
Week	3			
Manager	Silas Chakravarty			
Topic:				
Task #	Description	Deliverable	Assigned To	Status
1	Take final measurements for outdoor	Have a completed blueprint specified with	Joe Biscan	Partially Complete
2	start building weather data collecting	Image of the code progress	Silas Chakra	Partially Complete
3	Start on UI of app rough version	an image of a rough draft of UI	Anthony Ibar	Partially Complete
4	Find correct solenoid	purchase order for solenoid	Kyle Hansen	Complete

Team Name	Automated Watering			
Week	4			
Manager	Kyle Hansen			
Topic:				
Task #	Description	Deliverable	Assigned To	Status
1	Partially complete coding for weather app	Picture of code	Joe Biscan	Partially Complete
2	Familiarizing himself with Arduino coding	Having code prepared for when part arrives	Silas Chakra	Partially Complete
3	Partially complete coding for weather app (2	Picture of code	Kyle Hansen	Partially Complete
4	Continue UI and assist with weather app	An image of continued progress	Anthony Ibar	Incomplete

Team Name	Automated Watering			
Week	5			
Manager	Anthony Ibarra			
Topic:				
Task #	Description	Deliverable	Assigned To	Status
1	Continue coding of weather app	Picture of continued progress of code	Joe Biscan	Partially Complete
2	pick up parts from home depot if they have	reciept for home depot	Silas Chakra	Complete
3	3d printing files for encasing and PVC	Picture of files for said parts	Kyle Hansen	Partially Complete
4	Continue UI and assist with weather ap if ne	Picture of code progression	Anthony Ibar	Partially Complete
5	Assist Anthony with UI	picture of code progression	Silas Chakra	Partially Complete

Team Name	Automated Watering			
Week	6			
Manager	Joe Biscan			
Topic:				
Task #	Description	Deliverable	Assigned To	Status
1	Finish coding weather app	Picture of code	Joe Biscan	Complete
2	Begin Assembly Schematics Design	Scale Sketch of Assembly Design	Silas Chakra	Partially Complete
3	Pick up final part	Reciept	Silas Chakra	Complete
4	Design where components will go in	CAD or hand drawing	Anthony Ibar	Partially Complete
5	3d printing files for encasing and PVC angle	Picture of files for said parts	Kyle Hansen	Complete

Team Name	Automated Watering			
Week	7			
Manager	Silas Chakravarty			
Topic:				
Task #	Description	Deliverable	Assigned To	Status
1	Start coding the arduino to control moisture	Screenshot of coding progress	Kyle Hansen	Partially Complete
2	Build wifi capabilities for the app	Picture of code	Joe Biscan	Partially Complete
3	Make a detailed, professional schematic using simulation software	Picture of schematic	Silas Chakra	Complete
4	print 3d parts	having the parts printed	Kyle Hansen	Complete
5	connect the weather code to app	show functionality of the deliverable with	Anthony Ibar	Complete

Team Name	Automated Watering			
Week	8			
Manager	Kyle Hansen			
Topic:				
Task #	Description	Deliverable	Assigned To	Status
1	Code UI to check/control Solenoid status	Screenshot of code	Silas Chakra	Partially Complete
2	Check that physical parts connect well	Test connectivity of all phsyical components	Silas Chakra	Partially Complete
3	Continue coding arduino to control	picture of coding progress	Kyle Hansen	Complete
4	Gather plant information from garden and	picture of plant widget on ui	Joe Biscan	Complete
5	Add ability for user to interact with the app	list of succesfully working integrations	Anthony Ibar	Partially Complete

Team Name	Automated Watering			
Week	9			
Manager	Anthony Ibarra			
Topic:				
Task #	Description	Deliverable	Assigned To	Status
1	Connect arduino to ui	picture of code	Kyle Hansen	Partially Complete
2	Add how much precipitation into ui	picture of code	Joe Biscan	Complete
3	Workable notifications for user	picture of code	Anthony Ibar	Complete
4	Finish solenoid tab in code	picture of code	Silas Chakra	Partially Complete

Team Name	Automated Watering			
Week	10			
Manager	Joe Biscan			
Topic:				
Task #	Description	Deliverable	Assigned To	Status
1	Cut PVC into smaller pieces	Have 15 2ft pieces ready for assembly	Silas Chakra	Complete
2	Solder Electronic components	project soldered together	Kyle Hansen	Complete
3	Begin Assembly	Connectors, T's and PVC assembled	Silas Chakra	Complete
4	Collect moisture and solenoid data from	picture of code	Joe Biscan	Partially Complete
5	Set up http server	picture of server setup	Anthony Ibar	Partially Complete
6	Continue coding control of solenoids	GitHub Upload and Screenshot of code	Silas Chakra	Incomplete
7	Aquire housing for microcontroller	Casing of Micro-Controller		

Team Name	Automated Watering			
Week	11			
Manager	Silas Chakravarty			
Topic:				
Task #	Description	Deliverable	Assigned To	Status
1	Finish setup of system and work on expo	System is setup and expo video drafted	Joe Biscan	Partially Complete
2	Finish setup of system and work on expo de	System is setup and expo video drafted	Silas Chakra	Partially Complete
3	Finish setup of system and work on expo de	System is setup and expo video drafted	Kyle Hansen	Partially Complete
4	Finish setup of system and work on expo de	System is setup and expo video drafted	Anthony Ibar	Partially Complete

Team Name	Automated Watering			
Week	12			
Manager	Kyle Hansen			
Topic:				
Task #	Description	Deliverable	Assigned To	Status
1	Finish setup of system and work on expo de	System is setup and expo video drafted	Joe Biscan	Complete
2	Finish setup of system and work on expo de	System is setup and expo video drafted	Silas Chakra	Complete
3	Finish setup of system and work on expo de	System is setup and expo video drafted	Kyle Hansen	Complete
4	Finish setup of system and work on expo de	System is setup and expo video drafted	Anthony Ibar	Complete

Appendix C: Concept Maps for Design Alternatives

Table C-1: General Concept Map

Microcontro ller	Type of Piping	Valve System for Water	Power Supply	Soil Moisture Sensor	User Interface
---------------------	-------------------	------------------------------	-----------------	----------------------------	-------------------

Arduino	PVC	AC laminated solenoid	Battery	Capacitive soil moisture sensor	App
Tiva C	Soft hoses	Linear solenoid	Connected to greenhouse outlet	Humidity detection sensor	Website
Raspberry Pi	Metal piping	DC solenoid	Solar power	Soil moisture sensor with probe	Display directly on circuit shelter

Table C-2: Design Alternative I Map

Microcontroller	Type of Piping	Valve System for Water	Power Supply	Soil Moisture Sensor	User Interface
Arduino	PVC	AC laminated solenoid	Battery	Capacitive soil moisture sensor	App
Tiva C	Soft hoses	Linear solenoid	Connected to greenhouse outlet	Humidity detection sensor	Website
Raspberry Pi	Metal piping	DC solenoid	Solar power	Soil moisture sensor with probe	Display directly on circuit shelter

Table C-3: Design Alternative II Map

Microcontroller	Type of Piping	Valve System for Water	Power Supply	Soil Moisture Sensor	User Interface
Arduino	PVC	AC laminated solenoid	Battery	Capacitive soil moisture sensor	App
Tiva C	Soft hoses	Linear solenoid	Connected to greenhouse outlet	Humidity detection sensor	Website
Raspberry Pi	Metal piping	DC solenoid	Solar power	Soil moisture sensor with probe	Display directly on circuit shelter

Table C-4: Design Alternative III Map

Microcontroller	Type of Piping	Valve System for Water	Power Supply	Soil Moisture Sensor	User Interface
Arduino	PVC	AC laminated solenoid	Battery	Capacitive soil moisture sensor	App
Tiva C	Soft hoses	Linear solenoid	Connected to greenhouse outlet	Humidity detection sensor	Website

Raspberry Pi	Metal piping	DC solenoid	Solar power	Soil moisture sensor with probe	Display directly on circuit shelter
--------------	--------------	-------------	-------------	---------------------------------	-------------------------------------

Appendix D: Individual Pairwise Comparisons and Decision Matrix

Table D-1: Individual Pairwise Comparisons

Joe	Ease of product use	Development cost	Suitability for demo for 397	Time needed to complete	Resume attractiveness	Ease of completing the technical specifications
Ease of product use	1	5	3	1	7	5
Development cost	1/5	1	1	3	9	7
Suitability for demo for 397	1/3	1	1	1	5	7
Time needed to complete	1	1/3	1	1	9	5
Resume attractiveness	1/7	1/9	1/5	1/9	1	3
Ease of completing	1/5	1/7	1/7	1/5	1/3	1

the technical specifications						
------------------------------	--	--	--	--	--	--

Anthony	Ease of product use	Development cost	Suitability for demo for 397	Time needed to complete	Resume attractiveness	Ease of completing the technical specifications
Ease of product use	1	7	1	1	1/2	1
Development cost	1/7	1	1/3	5	1/2	2
Suitability for demo for 397	1	3	1	4	2	4

Time needed to complete	1	1/5	1/4	1	1/2	1
Resume attractiveness	2	2	1/2	2	1	1
Ease of completing the technical specifications	1	1/2	1/4	1	1	1

Kyle	Ease of product	Development cost	Suitability for demo	Time needed to	Resume attractiveness	Ease of completing
------	-----------------	------------------	----------------------	----------------	-----------------------	--------------------

	use		for 397	complete		the technical specifications
Ease of product use	1	5	5	7	9	1
Development cost	1/5	1	1	3	9	7
Suitability for demo for 397	1/5	1	1	1	5	7
Time needed to complete	1/7	5/7	5/7	1	9	5
Resume attractiveness	1/9	5/9	5/9	7/9	1	3
Ease of completing the technical specifications	1	1/9	1/5	1/7	1/9	1

Silas	Ease of product use	Development cost	Suitability for demo for 397	Time needed to complete	Resume attractiveness	Ease of completing the technical specifications
Ease of product use	1	6	7	5	9	3
Development cost	1/6	1	1/4	3	7	6
Suitability for demo for 397	1/7	4	1	6	8	8
Time needed to complete	5	1/3	1/6	1	8	4
Resume attractiveness	1/9	1/7	1/8	1/8	1	5
Ease of completing the technical specifications	1/3	1/6	1/8	1/4	1/5	1

Table D-2: Average Weights of Pairwise Comparisons

Average Ratings matrix	Ease of product use	Development cost	Suitability for demo for 397	Time needed to complete	Resume attractiveness	Ease of completing the technical specifications	Geometric Mean	Weights
Ease of product use	1.00	5.75	4.00	3.50	6.38	2.50	3.30	0.33
Development cost	0.18	1.00	0.65	3.50	6.38	5.50	1.56	0.16

Suitability for demo for 397	0.42	2.25	1.00	3.00	5.00	6.50	2.13	0.21
Time needed to complete	1.80	0.40	0.53	1.00	6.63	3.75	1.45	0.15
Resume attractiven ess	0.59	0.70	0.35	0.75	1.00	3.00	0.83	0.08
Ease of completin g the technical specificati ons	0.63	0.92	0.18	0.40	1.64	1.00	0.64	0.07
								1

Table D-3: Rating of Each Design Alternative Relative to the Criteria

Rate design alternatives	Alternative I	Alternative II	Alternative III
-----------------------------	------------------	----------------	-----------------

relative to criteria			
Ease of product use	5	3	3
Development cost	5	3	3
Suitability for demo for 397	3	3	3
Time needed to complete	5	5	3
Resume attractiveness	5	5	3
Ease of completing the technical specifications	5	5	5

Table D-4: Decision Matrix

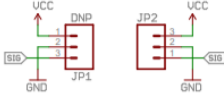
Scores for alternatives		Alternative I	Alternative II	Alternative III
Ease of product use	0.33	5	3	3

Develop ment cost	0.16	5	3	3
Suitabilit y for demo for 397	0.21	3	3	3
Time needed to complete	0.15	5	5	3
Resume attractive ness	0.08	5	5	3
Ease of completi ng the technical specifica tions	0.07	5	5	5
Score		4.58	3.60	3.14

Appendix E: Data Sheets

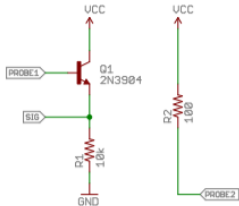
SparkFun Soil Moisture Sensor Datasheet

Connectors:
JST Jumper 3 Wire Assembly - PRT-09915
Screw Terminals 3.5mm Pitch (3-Pin) - PRT-08235
Vcc = 3.3V-5V



DO NOT POWER CONSTANTLY
It is recommended that you use a digital GPIO pin on whichever microcontroller or IC you're using to control the sensor to power the sensor.

Probe Circuit




Test different values for R1 to get lower power consumption while still getting a good ADC reading.

Rod length and spacing were not the most significant variables.
In general you want the probes long enough to reach the moist soil and not so close together that they are likely to touch accidentally.
Keeping them about an inch apart works great.
The big variable is the composition of the soil itself (especially salts), so ideally you would calibrate for each type of soil.
-Rob Faludi

Based off the Soil Moisture Circuit found at: <http://www.faludi.com/2006/11/02/moisture-sensor-circuit/>

PCB design inspired by the Soil Moisture Sensor from DFRobot
http://www.dfrobot.com/index.php?route=product/product&product_id=599

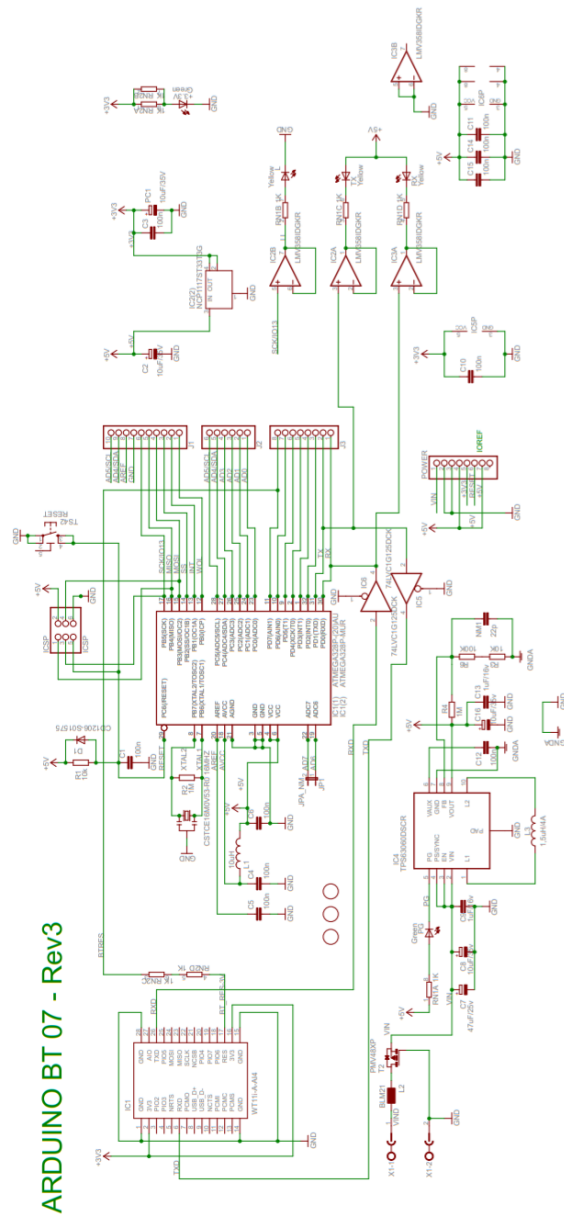


Released under the Creative Commons
Attribution Share-Alike 4.0 License
<https://creativecommons.org/licenses/by-sa/4.0/>

TITLE: SparkFun_Soil_Moisture_Sensor

Design by: Joel Bartlett	REV: v10
Date: not saved!	Sheet: 1/1

Arduino BT Datasheet



Reference Designs ARE PROVIDED "AS IS" AND "WITH ALL FAULTS. Arduino DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Arduino may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Arduino reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this information.

ARDUINO is a registered trademark.

Solenoid Datasheets

TECHNICAL DETAILS

SKU	APLO-1/2-12VDC
Position	Normally Open
Port Size	1/2" Male NPT
Voltage	12V DC
Body Material	POM Plastic
Components	Stainless Steel
Seal Material	EPDM Diaphragm
Orifice Size	8.5 mm
Temp Range	32 to 125° F / 0 to 50°C
PSI Range	3 - 115 PSI (Minimum Required)
Flow Rate	Cv 0.6 (Appx 4.5 GPM @ 60 PSI)
Power	6 Watts
Coil Connection	Terminal
Response Time	Fast Acting (Less than one second)

Duty Cycle	100% but not indefinitely
*Suitable Media	Water – Air -*Etc
Weight	4.10 oz
Height	2.55"
Length	2.85" port to port
Width	1.65"

PRODUCT OVERVIEW

Model APLO-1/2-12VDC of our plastic valve line is a two-way, direct acting valve with a normally open operating position. This means that there are two ports in which media flows, the valve will close when energized and prevent media from passing through. Taking the energy away from the valve will open it and continue the flow. It works across various media such as water, air, oil or other low viscosity fluids.

The port size is 1/2" inch NPT male threaded. It can withstand pressure up to 115 psi (8 bar) and temperatures as low as 32°F and as high as 125°F. The seal material is EPDM Rubber. This valve is capable of flowing 6.5 Gallons per minute at the max rated PSI.

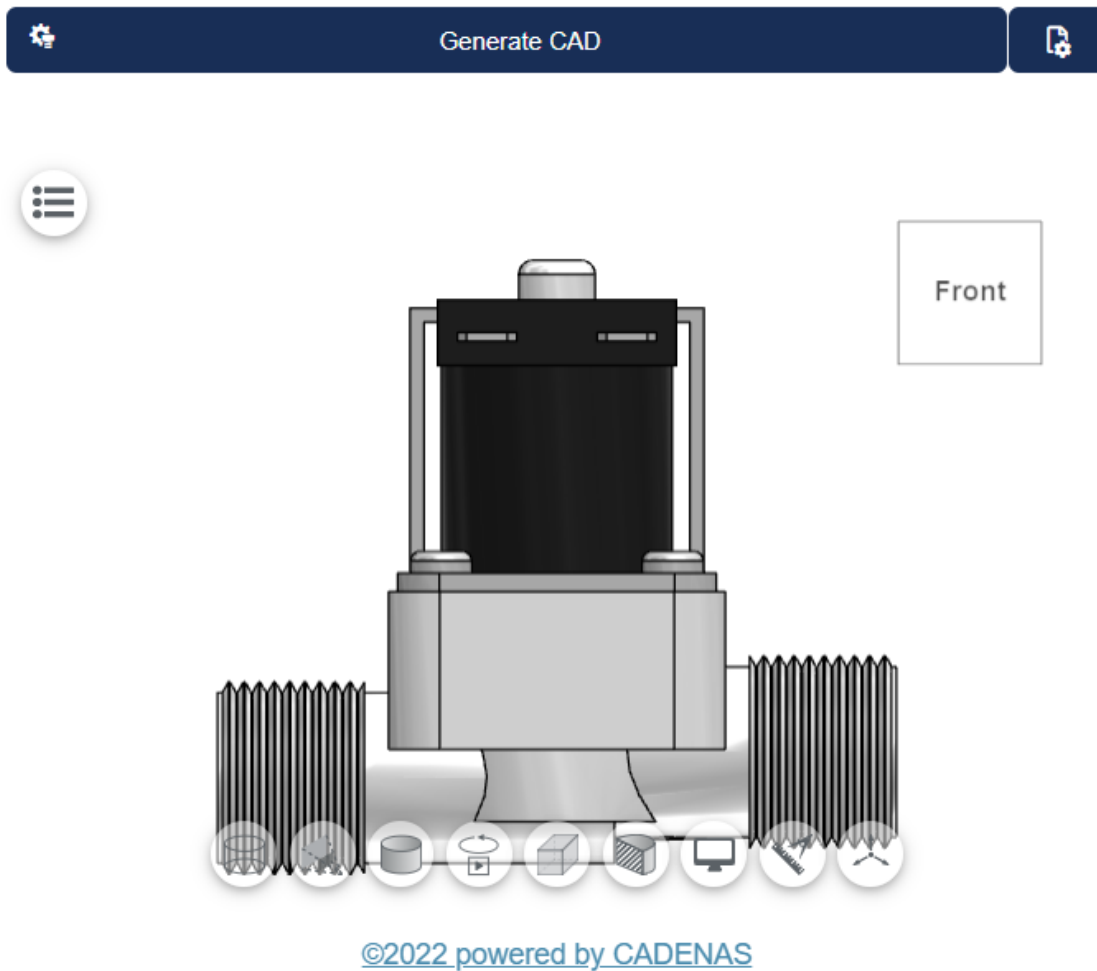
This specific plastic valve has a 12 Volt DC coil with a voltage range that is +- 10%. The coil head can easily be connected with your own terminal connectors or directly soldered. Its response time is very low; in fact, it acts in less than one second, making it a great choice for all types of applications.

Common uses include but not limited to;

- Water Filters
- Ice Makers
- Drinking Fountains
- Dental Tools
- Fish Tanks
- Beer Brewing
- Hydroponics
- Etc.

This model is mountable in any position and features a conveniently positioned arrow on the body to indicate the direction of flow. Included is a Built-in filter screen to prevent debris from entering the valve body.

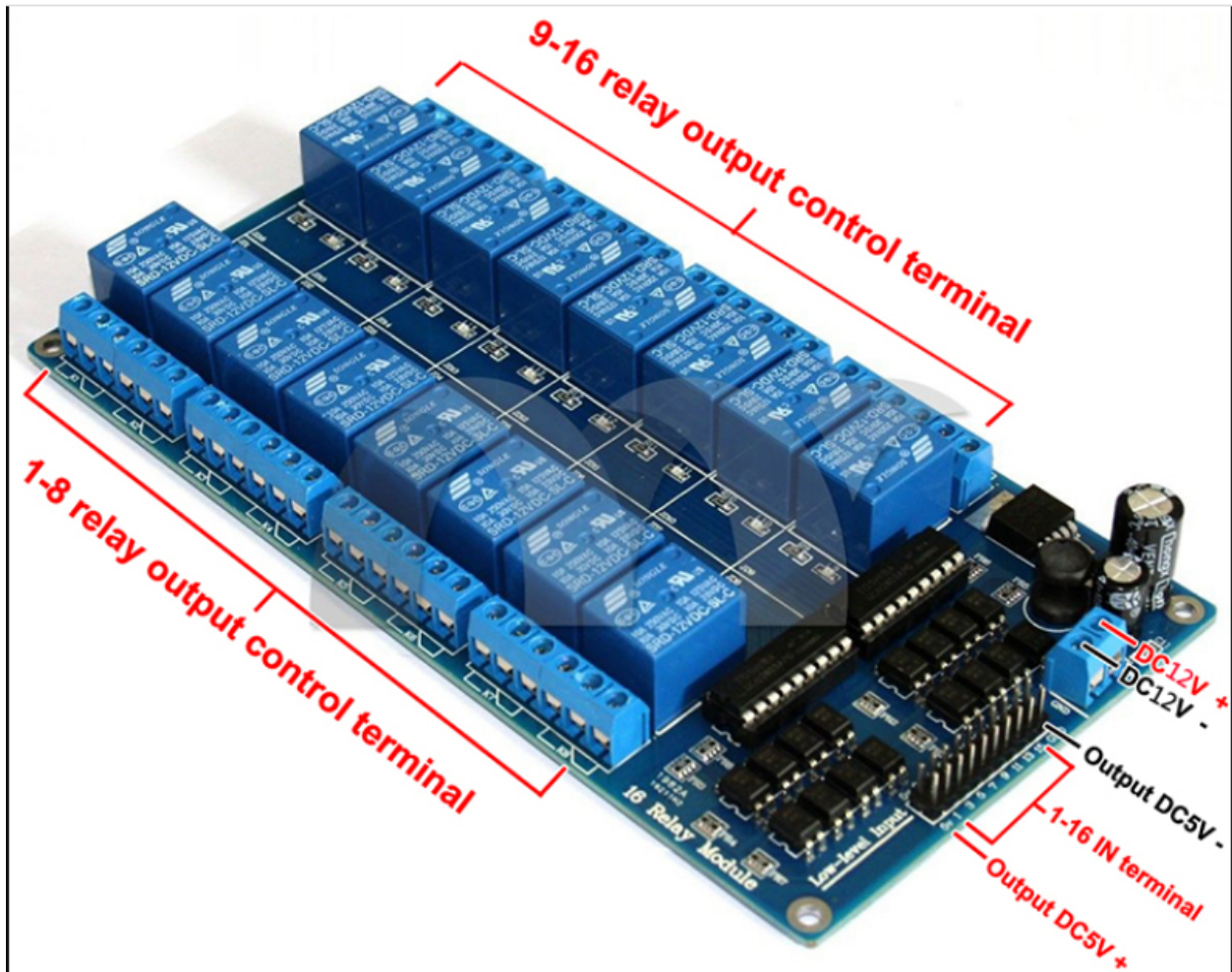
3D MODELS



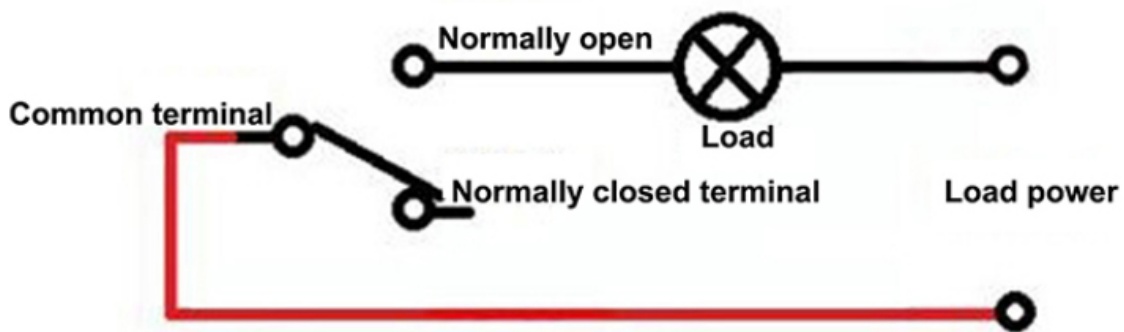
16 Channel 12VRelay Module

SPECIFICATION

- The use of the industry's top quality optocoupler isolation, strong anti-interference ability, stable performance
- 12V 16-Channel Relay interface board, and each one needs 15-20mA Driver Current
- 1-16 road can be any full on / off, or any road.
- Equiped with high-current relay, AC250V 10A ; DC30V 10A
- Standard interface that can be controlled directly by microcontroller (Arduino , 8051, AVR, PIC, DSP, ARM, ARM, MSP430, TTL logic)
- Indication LED's for Relay output status



Relay output wiring instructions



AC/DC Converter 12V 65W



Switch Mode Power Supplies Single Output AC/DC Power Supply

AEU65-120

Description:

The AEU65-120 is a single output power supply. This power supply is designed for a wide variety applications where high reliability is desired, including applications for the industrial and telecommunications markets. Excellent performance specifications are provided, together with compliance to European EMC (EN55022, Class B and EN61000-3-2), and Low Voltage directive.

Specifications (@25C)

Input Characteristics:

Input Voltage:	90-264VAC, 127-373VDC
Input Frequency Range:	47-63Hz
Input Current:	1.6A @ 115VAC, 0.8A @ 230VAC typ.
Max Inrush Current:	30A@115VAC, 60A@230VAC at cold start
Leakage Current:	<2.4mA/240Vdc

Output Characteristics:

Output Voltage:	12.0VDC \pm 1.5%Vdc
Output Current (Convection):	0-5.42A
Output Power (Convection):	65W
Adjustable Output Range:	11.4 – 12.6V. Output voltage can be adjusted at VR51
Ripple & Noise ¹ :	120mVp-p
Load Regulation:	\pm 0.5%
Line Regulation:	\pm 0
Efficiency:	87.0%
Start-up Time:	1000ms/230VAC, 2000ms/115VAC, full load
Rise-up Time:	30ms/230VAC, 30ms/115VAC, full load
Hold-up Time:	24ms/230VAC, 12ms/115VAC, full load
Over Current Protection:	110 – 160%. Hiocup mode. Resets automatically once the fault condition is removed.
Over Voltage Protection:	13.8 – 16.2VDC.

General Specifications:

Dimension (LxWxH):	99(3.9) x 75(3.0) x 35.0(1.38) mm (in)
Weight:	200g
Cooling:	Natural Convection
Isolation Resistance:	I/P—O/P, I/P—FG, O/P—FG: 500VDC/100M Ohms
Dielectric Strength:	I/P—O/P: 4.3KVDC; I/P—FG: 1.5KVAC; O/P—FG: 0.5KVAC
Warranty:	3 years
MTBF:	250K hrs. min. MIL-HDBK-217F (25°C)

Environmental Specifications:

Operating Temperature:	-20° to 50°C at full load (Refer to output load derating curve)
Operating Humidity:	20 to 90% RH, non-condensing
Storage Temperature:	-40 to 85°C
Storage Humidity:	10 to 95% RH, non-condensing
Temperature Drift:	<0.03%/°C (0-50°C)
Vibration:	10-500Hz, 2G 10min/cycle, period of 60min, each X, Y & Z axis

EMC & Safety Specifications²:

EMI Emissions:	Compliance to EN55022, CISPR22 Class B (Conducted & Radiated)
Harmonic Current:	Compliance to EN61000-3-2, 3
EMS Immunity:	Compliance to EN61000-4-2, 3-6, 8 & 11; EN55024 heavy, light industry level, criteria A
Safety Approval:	UL 60950-1, (insulation class -1)



¹ Ripple and noise are measured at 20MHz of bandwidth by using a 12" twisted-pair wire termination with a 0.1uF & 47uF parallel capacitors.

² The power supply is considered a component which will be installed into a final equipment. The final equipment must be re-confirmed that it still meets EMC directives.

Web: www.TriadMagnetics.com
Phone 951-277-0757
Fax 951-277-2757

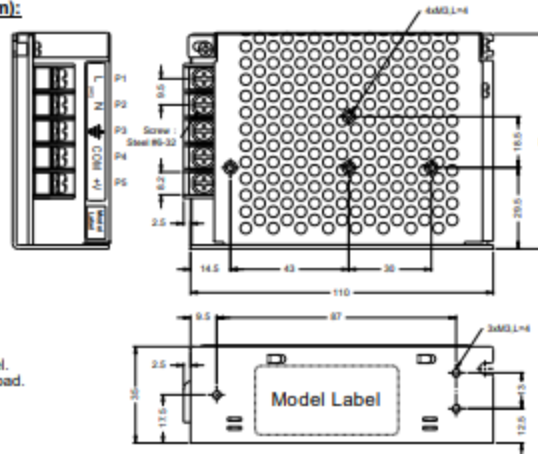
460 Harley Knox Blvd.
Perris, California 92571

Publish Date: September 20, 2021



Switch Mode Power Supplies Single Output AC/DC Power Supply

Outline Dimensions (mm):



NOTE :

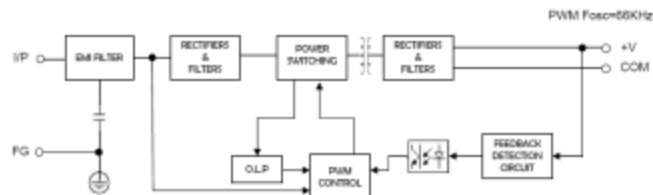
1. All I/O connection shall follow specified Model Label.
2. Temp = +50°C (max) at full load.

Connections:

AC Input Connector	
Pin	Assignment
P1	AC/L
P2	AC/N
P3	FG

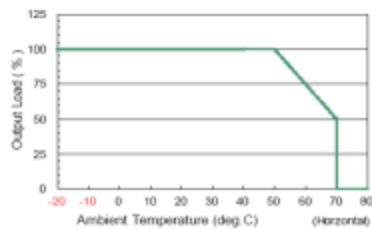
DC Output Connector	
Pin	Assignment
P4	COM
P5	V+

Block Diagram:

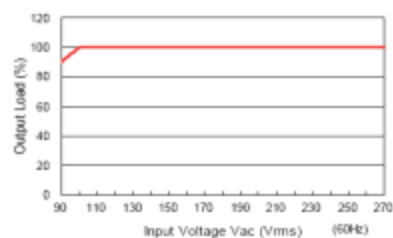


Derating Curve:

■ Output Derating VS Ambient Temperature : (HORIZONTAL MOUNTING)



■ Output Derating VS Input Voltage :



RoHS Compliance: As of manufacturing date February 2016, all standard products meet the requirements of 2015/863/EU, known as the RoHS 3 initiative.

* Upon printing, this document is considered "uncontrolled". Please contact Triad Magnetics' website for the most current version.

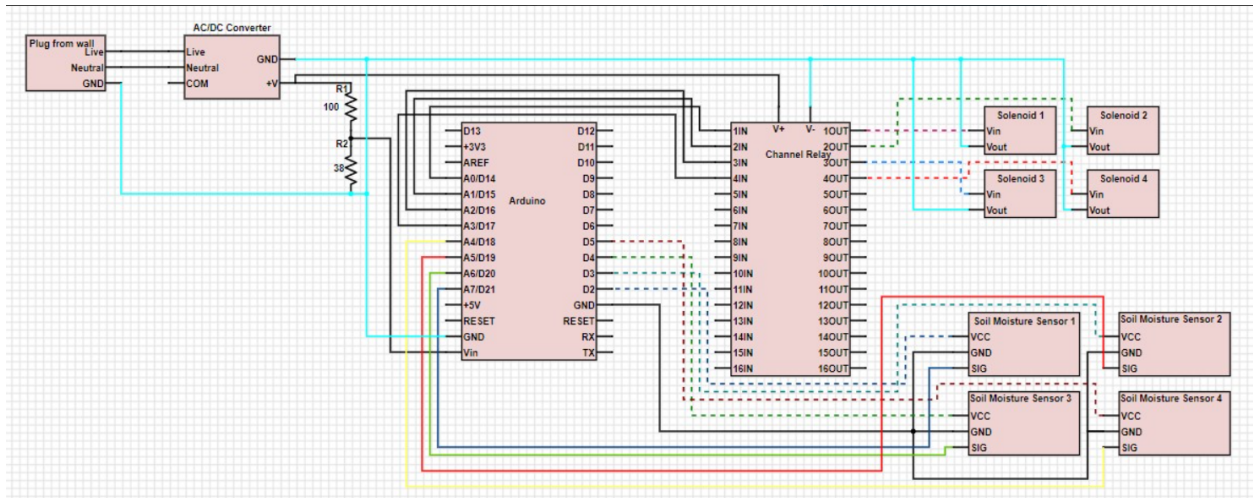
Web: www.TriadMagnetics.com
Phone 951-277-0757
Fax 951-277-2757

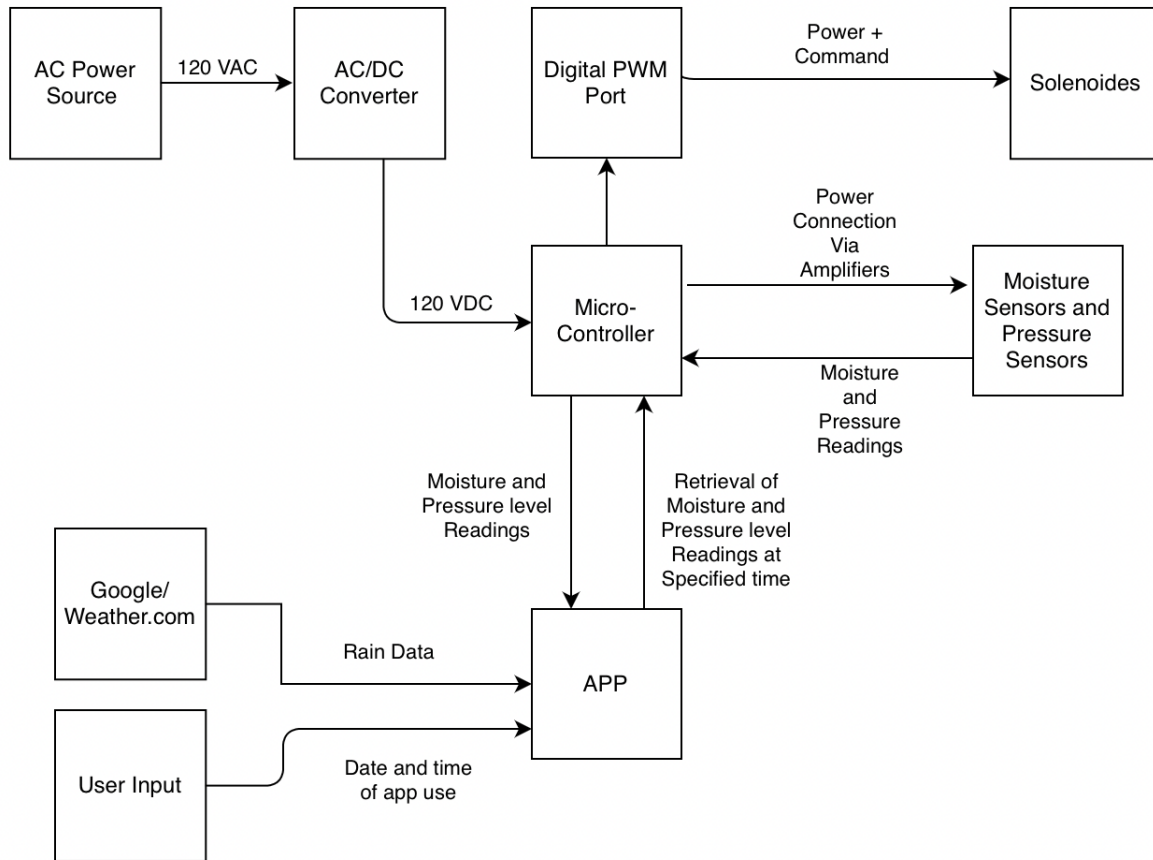
460 Harley Knox Blvd.
Perris, California 92571

Publish Date: September 20, 2021

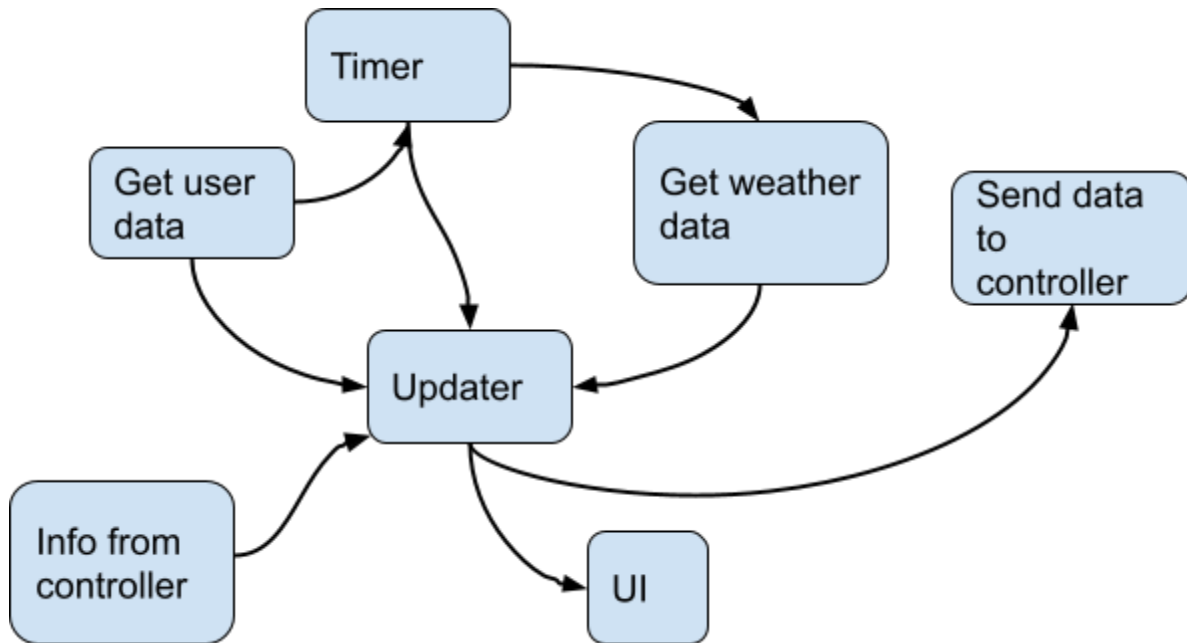
Appendix F: System Diagrams

Electrical Schematic of System

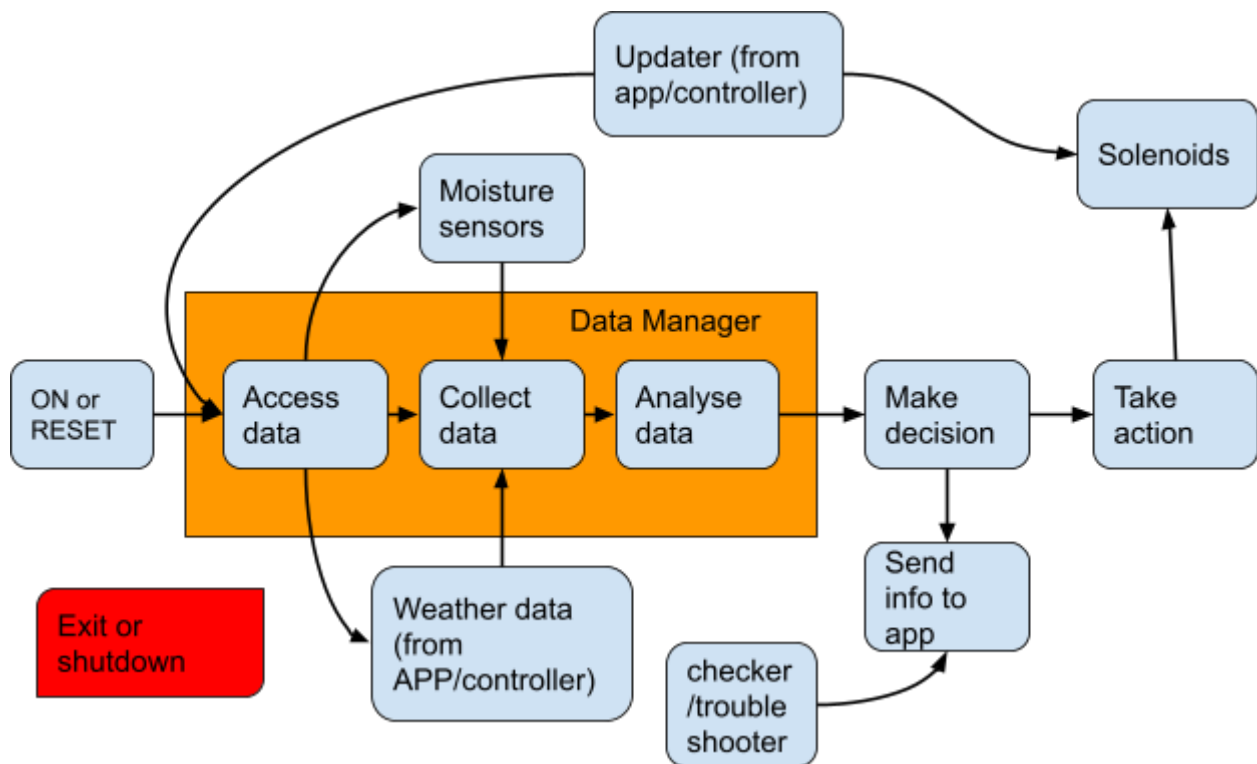


Level 1 Schematic of System Diagram

Software Design for App Diagram



Software Design for Microcontroller Diagram



Arduino Nano 33 IoT Pinout

