Independently Controlled Irrigation System for Agriculture Plots

Introduction

Problem:
Maintaining optimal environmental conditions for plant growth is essential for farmers, particularly those in rural farming lands where limited water supplies and lack of resources hinder productivity. Currently, both novice and experienced farmers/gardeners share a similar problem - different plant species require varying levels of soil moisture and ensuring each plant receives an ideal amount of water can be very time-consuming and error-prone. In modern day, most gardening enthusiasts manually cater their resources to each plant and this has proven to result in low-efficiency and lots of errors like under watering/overwatering plants. The concerns with modern-day irrigation plans is evident as underwatering/ overwatering plants cause significant issues like stunted plant growth and even plant death. Experiments highlighted by TreeNewal state that 80% of tree/plant problems can be attributed to the soil environment. According to the National Library of Medicine, 80-95% of the fresh biomass in plants is made up of water, highlighting the vital role water plays in a plant's growth, development, and metabolism. ScienceDirect further emphasizes that inconsistent watering and drought results in consequences like high salinity levels, heat stress, and attack of pathogens. This challenge becomes significantly more profound for farmers that have to monitor the specific water needs for different plant species.

The need for a precise, efficient, automatic irrigation solution is evident. Numerous countries across the world depend on the plant/crop yield of irrigated lands for resources like food. Clearly, inefficiencies and errors in the modern day manual irrigation have severe consequences as citizens of various countries are dependent on the plant's health and yield. According to the Food and Agriculture Organization (FAO), countries like India and Mexico depend on irrigated land for 55% of their agricultural output. As we can see, proper irrigation and watering systems are the key to high productivity in crop/plant outputs. The FAO also further highlights the need for a new automated watering system as currently, countries are spending a high amount of resources and money but are still not getting the results they want. For example, despite the high investments in the agriculture industry and improved irrigation, as much as 60% of the water diverted and pumped for irrigation is wasted. Smart, automated systems that can provide tailored irrigation for varying plant species are the next step in ensuring successful plant growth and assisting in combating real world crises like water scarcity.

References:
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7911879/
https://treenewal.com/6-common-tree-planting-mistakes-and-how-to-avoid-them/
https://www.sciencedirect.com/science/article/pii/S2666916121000190
https://www.fao.org/4/t0800e/t0800e0a.htm

Solution:

Our solution aims to combat the inefficiencies and inconsistencies of modern-day irrigation approaches by creating a smart,automated watering system designed to cater to the needs of varying plant species. At the heart of this solution is a central water supply that is connected to each of the plants through its own valve. To prevent overwatering/underwatering, the most common issues when manually watering plants, each valve dispenses water based on real-time soil moisture data. This approach ensures that each plant receives the optimal amount of water and the correct times, and combats the issues in water waste and manual labor. By utilizing real-time data and automating the irrigation, we can eliminate the inevitable errors that come with human labor and significantly improve plant health/growth.

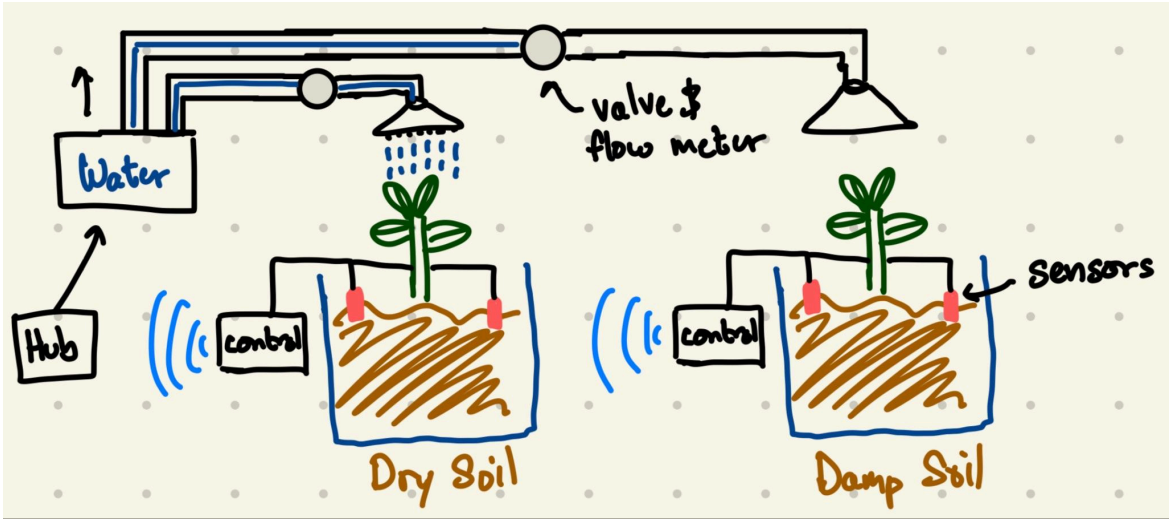
Our solution is split into 2 components, the irrigation system and the sensing system. The sensing system collects real-time data from the soil-moisture sensors to inform the irrigation system when and how much water each plant needs. Additionally, at the core, our solution includes a central microcontroller that will process the data from the soil sensors and can relay the information so the water valves provide the optimal amount of water to ensure precise and efficient crop growth. Our sensors will be distributed evenly across the soil in the plant environment and will be wirelessly transmitted to the control system via Bluetooth Low Energy. Our new irrigation solution will utilize Bluetooth Low Energy to transmit data as in rural area/farm lands WI-FI availability is minimal. In order to assist farmers that look to improve crop yield, we have taken account of the lack of available resources in the rural countryside. The control system processes data from the sensors and activates the irrigation system. The irrigation system includes pumps that use Pulse Width Modulation to regulate flow rate through the valves. Flow sensors will be integrated to deliver key information regarding water volume and flow rate to ensure the correct amount of water is delivered. We will include a user interface with React so users are enabled to adjust irrigation settings, track water usage in real-time, and customize watering schedules to their liking. This automated watering solution will ensure plants receive the perfect care without much human intervention, making it an ideal solution for managing diverse plant environments.

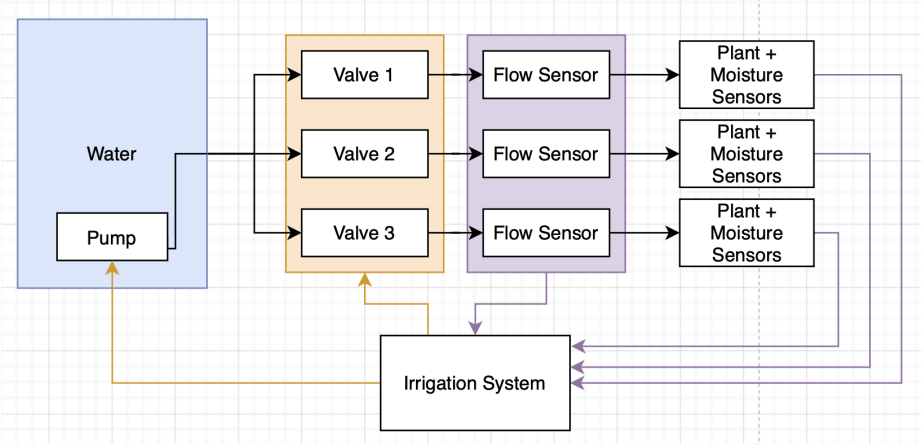
Our sensing system will also include humidity and temperature sensors to maintain the optimal soil moisture levels based on environmental factors. Given a plant's optimal soil moisture range (x% - y%), the microcontroller will use the readings from the soil moisture sensors and ensure water is dispersed so the current moisture level is in the middle of the optimal range. Then, considering the humidity and temperature sensors, our microcontroller will either adjust the water dispersed to make the moisture closer to x% or y%. If the environmental factors cancel out, then the microcontroller will disregard the humidity/temperature readings.

Example: We know that Basil has an optimal soil moisture range from 40-60%. Consider an example where our Soil Moisture Sensors show a current value of 50% moisture. If our humidity sensors depict a low humidity level and high temperature level, then we will disperse water so the moisture levels go closer to 60%. On the other hand, if our humidity sensors depict a high humidity level and low temperature level, then we will disperse water so the moisture levels go

closer to 40%. Lastly, if our humidity sensors depict a high humidity level and high temperature level, then these environmental factors cancel out and our microcontroller only disperses water based on the soil moisture sensor.

Visual Aid:





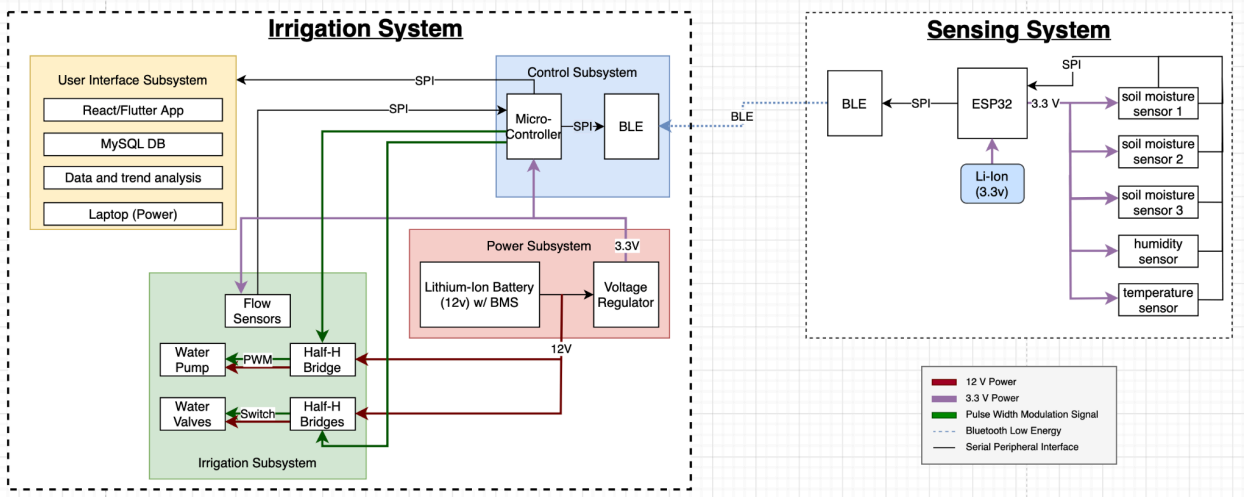
High-Level Requirements List:

1) **Precise Water Delivery :** Upon recognizing low moisture levels in the soil, the system must activate the correct valve and deliver the optimal amount of water required. The system must release water at a rate of .5 - 2 liters per minute, depending on the current soil moisture levels.The flow sensor must indicate and verify that the water released is within +/- 3% of the requirement indicated by the current moisture level.
2) **Real-Time System Response :** A key feature of our solution is that human error is eliminated and the system is automated in real-time. For our solution to be viable, we must ensure that the water delivery occurs with a fast response time to maintain optimal moisture levels. The system must process data from the sensors and adjust water flow within the valves within 5-10 minutes of receiving the new data.

3) **Accurate Moisture Detection :** At the foundation of our solution, we must first ensure that our sensors are able to accurately detect the moisture levels in the soil in order to begin the automatic watering system. The system must measure and transmit soil moisture data from three independent sensors providing coverage of 1sq meter each. The sensors must transmit data with an accuracy of +/- 3% compared to the actual soil moisture levels. Then, the sensors must trigger a response to the control unit within 5 minutes if the moisture levels reach below a predefined threshold to initiate the watering cycle. The system must send moisture data to the ESP with a latency of no more than 5 seconds for continuous monitoring and quick data processing.

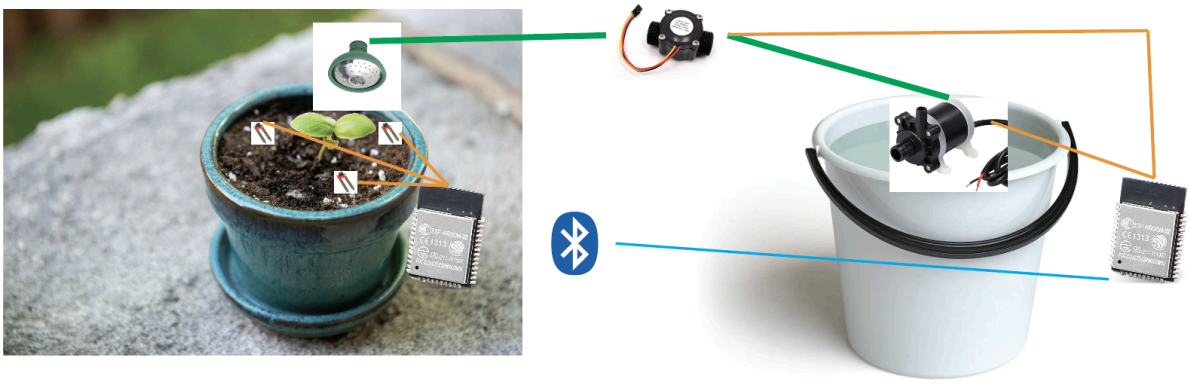
Design

Block Diagram:



* The User Interface System is powered by a mobile device or computer (not requiring system power).

PHYSICAL DESIGN



The physical design of the automated plant watering system is centered around a compact, modular setup that integrates sensors, actuators, and wireless communication. The system comprises a potted plant equipped with soil moisture sensors that are strategically embedded into the soil. These sensors continuously monitor the moisture levels and are interfaced with an ESP32 microcontroller, which serves as the central processing unit for data collection and transmission.

The ESP32 communicates via Bluetooth to another ESP32, which controls the water delivery system. The water delivery system includes a pump housed in a water reservoir, connected to a hose that leads to a small, perforated showerhead positioned above the plant. When the moisture levels fall below a predefined threshold, the microcontroller activates the pump to

water the plant. The modular design of the system, with distinct components for sensing and actuation, allows for scalability and easy maintenance, ensuring reliable operation in diverse environments.

IRRIGATION SYSTEM

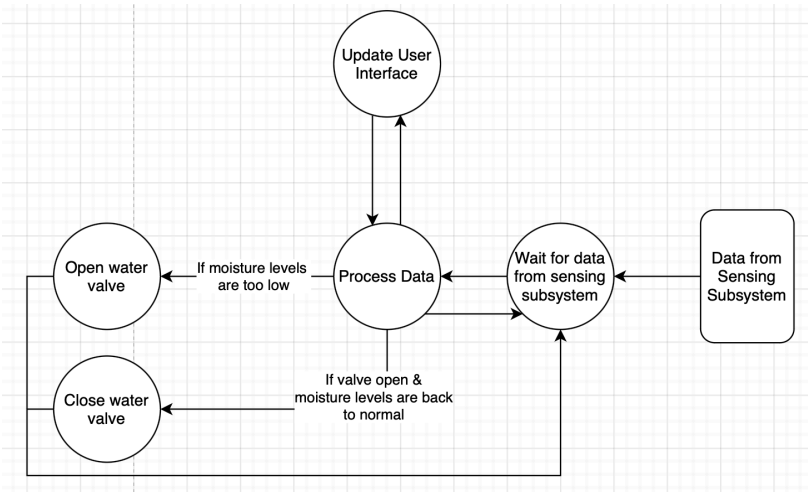
The irrigation system is the core of our project and includes subsystems that control the amount of water released, and when it is released. It also sends data to our user interface.

1. Control Subsystem

The control subsystem acts as the central hub of the irrigation system. It includes a microcontroller that processes inputs and outputs according to the program logic defined for irrigation. This subsystem uses the SPI (Serial Peripheral Interface) protocol to communicate with the user interface subsystem, and uses BLE (Bluetooth Low Energy) for wireless communication with remote sensor modules, receiving real-time soil moisture data. The control subsystem interprets these data points to manage the activation and deactivation of the water valves and pumps through the irrigation subsystem, thus maintaining optimal soil moisture levels across the garden. Finally, it will send real-time data to our user interface using SPI.

The state machine below describes the states of our control subsystem. Our start state is the “wait for data from sensing subsystem” state. The algorithm is described below:

Control System Algorithm: First, set a predefined threshold for soil moisture based on plant-type and environmental factors. We will define the acceptable water flow rates and precision requirements as described above. Our data collection will continuously monitor and collect data from the independent sensors and each will transmit data to the microcontroller with low latency. If one of the 3 sensors provides data that deviates significantly (+/- 5%) we will recalibrate that sensor and have that sensor reprocess data. If the average moisture levels drop below the specified threshold, we will calculate the deficit between current moisture levels and target moisture levels and begin the irrigation process. The flow sensors will ensure that the irrigation system is delivering an accurate amount of water. After the irrigation system completes, our sensors will recalibrate and confirm the current moisture levels have reached the target. Once the soil moisture reaches the specified threshold, our microcontroller will close the valves and shut off the water system.



Requirement	Verification
Must be able to receive and aggregate data every 5 seconds over BLE.	For a specific subsystem, start timing when it sends it sends it data up to when it is received and processed by the ESP32. These messages will be seen on the logs. Confirm that the time is less than 5 seconds
Must be able to calculate water necessary and respond with water for each plot within 30 seconds	Time the system from the moment it receives the data until the irrigation subsystem activates and the first water is dispensed. Verify the calculation process and water release timing using timestamps in the debug logs. Repeat this process 5 times to confirm stability.
Must be positioned within range of at least one sensing system device for BLE	Use a signal strength tool to confirm the BLE stable signal between the control subsystem and the nearest sensing device.
Must be supplied power of 3.3V with a tolerance of ±0.1V	Confirm power output with a multimeter.

2. User Interface Subsystem

The user interface subsystem provides an interactive platform for users to monitor and control the irrigation system. This subsystem is built with a React or Flutter application, enabling a robust and responsive user experience across various devices. Through this interface, users can customize watering schedules based on specific plant needs, view historical data to track water usage and soil moisture levels, and adjust settings to optimize water conservation and plant health.

Requirement	Verification
Must update within 30 seconds of receiving data for each associated plot from control subsystem via SPI	Monitor the system's data update process by recording timestamps for when the control subsystem sends data and when the user interface updates. Ensure that the update occurs within 30 seconds across multiple tests for consistency.

3. Irrigation Subsystem

The irrigation subsystem is responsible for the physical delivery of water to the plots. It consists of a network of water pumps and valves controlled by the microcontroller in the control subsystem. The pumps use PWM (Pulse Width Modulation) signals to regulate the flow rate, while the valves, operated by switches, control the distribution of water to different sections of the garden. Flow sensors integrated within this subsystem provide feedback on water volume and flow rate, ensuring that each section receives the precise amount of water as instructed by the control subsystem. This feedback loop enables the system to adjust the water delivery in real-time, preventing overwatering or underwatering.

Requirement	Verification
Water pump must operate within a voltage range of 10V to 12V and provide flow rates from 0.5 to 2 liters per minute.	With the 12V power supply, run the pump at full power for one minute, and check volume of water output.
Valves must open and close in less than 1 second upon receiving a command via PWM to allow timely distribution of water	Use microcontroller to send a PWM signal to the valve and measure the time between sending the signal and the valve reaching a fully open or fully closed position. Repeat this process multiple times to ensure that the valve consistently responds within 1 second. Document the average time and any deviations.
Flow sensors must have an accuracy of within 3% to correctly report water delivery metrics back to the control subsystem for real-time adjustments with plants	To verify, run water through the system for a set time, such as 1 minute, and record the flow sensor's output. Measure the actual volume of water collected in a container. Compare the flow sensor’s reading to the measured volume. If the sensor's reading is within 3% of the actual measured volume, it passes the accuracy test. Repeat the process to ensure consistent results.

4. **Power Subsystem**

The power subsystem supplies energy to the entire irrigation system. It includes a 12V lithium-ion battery with a capacity of 500mAh, regulated by a Battery Management System (BMS) to manage charging and discharging. The BMS prevents overcharging and over-discharging, extending battery life and ensuring safe operation. A voltage regulator steps the battery output down to 3.3V to power the microcontroller and sensors.

Requirement	Verification
The lithium-ion battery must provide a continuous output of 12V ±0.6V with a capability to supply at least 500mA to the rest of the system	Confirm output with a multimeter.
The BMS must provide a supply of 12V ±0.6V to the pump and valves	Confirm output with a multimeter.
The voltage regulator must provide a stable output of 3.3V ±0.1V to ensure sensitive components like the microcontroller and sensors operate within safe electrical thresholds	Confirm output with a multimeter.

SENSING SYSTEM

The sensing system is an edge system placed in every pot/plot that needs to be monitored. It collects the data from the three sensor subsystems, averages the moisture levels, and broadcasts that data via BLE.

Example of Sensing System:

We know that Basil has an optimal soil moisture range from 40-60%. Consider an example where our Soil Moisture Sensors show a current value of 50% moisture. If our humidity sensors depict a low humidity level and high temperature level, then we will disperse water so the moisture levels go closer to 60%. On the other hand, if our humidity sensors depict a high humidity level and low temperature level, then we will disperse water so the moisture levels go closer to 40%. Lastly, if our humidity sensors depict a high humidity level and high temperature level, then these environmental factors cancel out and our microcontroller only disperses water based on the soil moisture sensor.

Given a plant's optimal soil moisture range (x% - y%), the microcontroller will use the readings from the soil moisture sensors and ensure water is dispersed so the current moisture level is in the middle of the optimal range. Then, considering the humidity and temperature sensors, our microcontroller will either adjust the water dispersed to make the moisture closer to x% or y%. If the environmental factors cancel out, then the microcontroller will disregard the humidity/temperature readings.

1. **Sensor Subsystem**

The sensor subsystem is centered around a capacitive soil moisture sensor, temperature, and humidity sensor connected to an ESP32 microcontroller.

The capacitive soil moisture sensor operates by measuring changes in the dielectric constant of the soil. Water has a significantly higher dielectric constant compared to dry soil, so as the water content in the soil increases, the sensor detects this change and outputs a corresponding analog voltage. The sensor does not have direct contact with the soil, reducing corrosion and increasing longevity compared to resistive sensors. The analog output voltage from the sensor ranges between 0V (dry soil) and approximately 3.3V (saturated soil), depending on the moisture content. This analog output is connected to an analog input pin (e.g., GPIO36) on the ESP32 microcontroller. The ESP32 uses its built-in 12-bit ADC (Analog-to-Digital Converter) to convert the analog signal into a digital value, which is then processed and scaled to represent the moisture level as a percentage. This sensor requires a stable 3.3V power supply, connected through the ESP32's VCC pin, and is grounded to the system's GND pin.

The temperature sensor in this system is responsible for monitoring the ambient temperature in the environment around the plant. This data can help in adjusting irrigation patterns, as high temperatures can increase the rate of water evaporation, requiring more frequent watering. The temperature sensor is powered by the ESP32's 3.3V line and is connected to a digital GPIO pin for communication. The sensor reads temperature values and sends them to the ESP32 using a simple digital signal protocol.

The humidity sensor tracks the relative humidity (RH) of the air surrounding the plant. Changes in humidity can affect plant transpiration and soil moisture retention, and monitoring these levels helps refine the irrigation system's water delivery. The humidity sensor outputs digital data representing the percentage of relative humidity (ranging from 0% to 100%), which is transmitted to the ESP32 over a single GPIO pin. Like the other sensors, the humidity sensor is powered by the 3.3V supply and grounded to the system's GND pin.

Requirement	Verification

Soil Moisture Sensors: Must measure soil moisture content within a 3% tolerance for accurate irrigation processes.	Test soil moisture sensors in three different levels of moisture (not saturated, somewhat saturated, extremely saturated).
Humidity Sensors: Must measure the humidity levels/ water vapor present in the air accurately to help maintain optimal soil requirements. If there is low humidity, we will add additional water due to high evaporation and vice versa.	Print humidity level reading from ESP32 in Arduino IDE to confirm using weather app
Temperature Sensor: Must measure the temperature levels present in the environment to help maintain optimal soil environments. If there is a high temperature, we will add additional water due to evaporation due to heat.	Print temperature reading from ESP32 in Arduino IDE and confirm using weather app
All sensors: Must operate at 3.3V and pull a maximum of 300mA (together)	Use a multimeter to confirm each sensor is operating at a maximum of 300mA

2. Power Subsystem

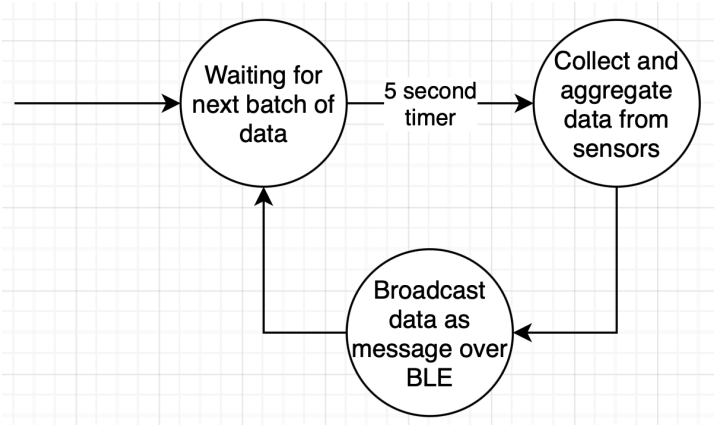
The power subsystem utilizes a 3.3V lithium battery (e.g., a 18650 Li-ion battery) to supply power to both the ESP32 microcontroller and the soil moisture sensor. The battery is connected to the ESP32 through its VIN or 3.3V input pin, benefiting from the ESP32's built-in power management. In case the battery outputs a higher voltage, a voltage regulator is included to maintain a steady 3.3V supply. The soil moisture sensor shares this power supply with the ESP32, receiving power through its VCC pin. This ensures that both components are powered consistently throughout the operation.

Requirement	Verification
The lithium-ion battery must provide a minimum continuous voltage output of 3.3V and a current output of 200 mA to adequately power the microcontroller and sensor subsystem (with three sensors)	Confirm output with a multimeter.

3. Control Subsystem

The control subsystem is managed by the ESP32, which serves as the main control unit. It reads analog soil moisture data from the sensor and transmits the processed information to a central hub using Bluetooth Low Energy (BLE). The ESP32 operates as a BLE peripheral device, sending data either via advertisement packets or through GATT-based communication every 30 minutes. For scenarios involving multiple sensors, SPI communication may be employed for more complex data handling. The system is timed by the ESP32's internal clock, ensuring data transmission is spaced evenly. Additionally, TDMA scheduling can be implemented to avoid interference by ensuring that each sensor transmits data in its own designated time slot.

The state machine for the Sensing System's control subsystem can be seen below. The peripheral systems have short, but important tasks, making the state machine simple.



Requirement	Verification
Must be able to send data every 5 seconds over BLE once system is initialized	Monitor the system logs and timestamps to verify that data packets are sent over BLE every 5 seconds. Use a BLE debugging tool to confirm consistent transmission intervals over multiple cycles.
Must be able to receive data every 15 seconds over SPI from sensor subsystem	Log every time data is received from a sensor subsystem, and measure the time in between each broadcast of data.

TOLERANCE ANALYSIS

The power system needs to be carefully designed to ensure stable voltages for all the subsystems. The power subsystem contains a 12V battery but since our PCB requires a 3.3 V we need to use the LD1117 voltage regulator.
Since: $Vin > Vout + Vdropout$
$Vdropout = 1V\ (LD1117\ datasheet)$

$Vout = 3.3V$
$Vin = 12V$
Since V_{in} is greater than 4.3 V but lesser than 15V, we ensure that the output voltage will be 3.3V

The AQT15S solenoid valve operates at 12V with a 15% voltage range which means it can operate between 10.2V - 13.8V.
The valve will be connected to the microcontroller which operates at 3.3V but will need an Hbridge. Below is a table from the L9110 H bridge

Symbol	Parameters	minimum	Typical	maximum	units
VH_{out}	Output high	7.50	7.60	7.70	V
VL_{out}	Output low	0.35	0.45	0.55	V
VH_{in}	Input high	2.5	5.0	9.0	V
VL_{in}	Input low	0	0.5	0.7	V

On analyzing the table we can calculate the internal resistance.
$Vcc = 9V$
$Iout = 750mA$
Output high voltage $VHout = 7.6V\ (typical)$
$Vdrop = Vcc - VHout = 9V - 7.6V = 1.4V$
Using Ohm's Law: $Rinternal = \frac{Iout}{Vdrop}$
$Rinternal = 1.87\Omega$

The Hbridge outputs about 750mA ~ 800mA
The maximum voltage drop will be $Vdrop = IR$ (Ohm's Law): $800mA * 1.87\Omega = 1.496V$
$Vsolenoid = 12V - 1.496V = 10.504V$ which is more than 10.2V which will allow the solenoid to work as expected

Irrigation System:
1. SPI Clock Frequency Analysis

- The ESP32 can operate SPI at a maximum clock speed of 80 MHz when using SPI IO_MUX pins. However, for the purpose of this analysis, we are considering using a 10 MHz clock.
- The SPI protocol relies on clocking in data bits at each clock pulse. With a 10 MHz clock, the transfer rate is: Data Rate = $10\,MHz = 10 * 10^6 bits/second$

2. Data Transfer Calculation

- Assume each soil moisture sensor provides 16 bits (2 bytes) of data per sample.

- If we have 3 sensors connected, the total data per transmission cycle will be:
- Total Data = $16\,bits \times 3 = 48\,bits$
- The time required to transfer this data at 10 MHz is:
- Transfer Time = $\frac{48\,bits}{(10*10^6)} = 4.8\mu s$
- This means it takes only 4.8 microseconds to transfer data from 3 sensors using SPI at 10 MHz, which is extremely fast compared to the time needed for other operations.

3. Tolerance on Clock Frequency

- SPI requires a frequency tolerance to avoid data corruption. A ±5% tolerance on 10 MHz clock means the actual clock speed can vary between:
- Minimum Clock=10 MHz−(10×0.05) MHz=9.5 MHz
- Maximum Clock=10 MHz+(10×0.05) MHz=10.5 MHz
- The transmission time for 3 sensors at these frequencies would be:
    - At 9.5 MHz: Transfer Time=48 bits /(9.5×106 bits/sec)=5.052 μs
    - At 10.5 MHz: Transfer Time=48 bits/(10.5×106 bits/sec)=4.57 μs
    - The difference in transfer time is very small (±0.57 μs), meaning even with a 5% variation in clock frequency, the system can still relay data reliably.

Even with a ±5% variation in the SPI clock frequency (ranging from 9.5 MHz to 10.5 MHz), the data transfer time difference is only ±0.48 microseconds. Since the system is designed to transmit data every 5 seconds, this small variation in transfer time will not impact overall system performance. The ESP32 can easily communicate with the central PCB in a timely and reliable manner, making the system robust for its irrigation tasks.

COST ANALYSIS:

Description	Manufacturer	Quantity	Extended Price	Link
Ferry-Morse 200MG Basil Sweet Italian Herb Plant Seeds Full Sun	Ferry-Morse	2	2.88$	Link
Element Live Plant Aloe Vera Plant in 4in Pot	Altman Plants	2	9.94$	Link

Everwilde Farms - 200 Organic Watercress Herb Seeds - Gold Vault Jumbo Bulk Seed Packet	Everwilde Farms	1	3.96$	Link
LithiumWerks ANR26650M1B 3.3 V 2.6 Ah Lithium Iron Phosphate (LiFePO4) Battery (Pre-Order)	StorTronics	2	13.4$	Link
KBT 12V 1200mAh Rechargeable Li-ion Battery, Bare Leads Wire Replacement Battery Pack with 12V Charger Compatible for 12V Devices RC Car, Boat, Robot, DIY, LED Light Kit	KBT KEEP BETTER TECH	2	33.98$	Link
Voltage Regulator - 3.3V LD1117V33	Sparkfun	2	4.20$	Link

SCHEDULE:

Week	Task	Person
Sept 30 - Oct 6	Revise Proposal	Everyone
	Complete Design Document	Everyone
	Find temperature and soil moisture sensors	Sneh, Aditya
	Order all the parts needed	Ary
Oct 7 - Oct 13	Design Review	Everyone
	Code ESP32 for sensor data for temperature and moisture	Sneh
	Design PCB	Ary, Aditya
	Debug the Sensing subsystem	Sneh
Oct 14 - Oct 20	Test data from ESP32 to PCB	Sneh
	Code the microcontroller using Arduino	Ary
	Make a prototype for the Irrigation Subsystem	Aditya
Oct 21 - Oct 27	Test and Debug Irrigation subsystem	Sneh
	Make second prototype of Sensing subsystem	Aditya

	Make revisions to PCB design	Ary
Oct 28 - Nov 3	Test data coming from both sensing subsystem	Sneh, Aditya
	Test closed loop feedback for both	Sneh, Aditya
	Start building the User Interface	Ary
Nov 4 - Nov 10	Optimize user interface by adding helpful features	Ary
	Schedule data for BLE communication	Sneh
	Debug any subsystem not working	Aditya
Nov 11 - Nov 17	Finetune algorithm based on temperature and soil moisture	Aditya
	Add a third sensing subsystem	Sneh, Ary
Nov 18 - Nov 24	Mock Demo + Debug	Everyone
Nov 25 - Dec 1	Spring Break	Everyone
Dec 2 - Dec 8	Final Demo + Documentation	Everyone
Dec 9 - Dec 15	Final Presentation	Everyone

ETHICS AND SAFETY

Our optimized irrigation system should take into consideration the principles contained in the IEEE Code of Ethics including data confidentiality. The sensor system collects moisture levels as well as the amount of water consumed within a given period and this information may be sensitive and private to each farmer/gardener. We intend to use Bluetooth Low Energy (BLE) to transmit information ensuring safety of the communication and users will be sure that their data is classified. Furthermore, we want our solution to be available not only to gardeners looking to improve their plant growth but also for farmers looking to improve their crop yield in rural areas. This adheres to the principles of IEEE non discrimination and provision of equal opportunity. Lastly, considering the environmental impact of our solution, IEEE's principle is to comply with ethical design and sustainable development practices. Our solution inherently looks to reduce water waste and contribute to the sustainable development of agriculture. We will ensure that

the components used in our solution are energy-efficient and environmentally safe. Our project will adhere to the Restriction of Hazardous Substances (RoHS) directives to eliminate hazardous materials from our system. We will do our due diligence to incorporate electronic equipment that doesn't have hazardous substances like lead,mercury, and cadmium.

According to the IEEE Code of Ethics, our design must hold paramount the safety, health, and welfare of the public. If there are faults in our irrigation system, crops may get damaged and lead to economic losses for farmers and gardeners. To abide by the IEEE Code of Ethics, we will install alert systems that will instantly notify users if a sensor or valve malfunctions in order to prevent over/under watering. Additionally, in our 3 Requirements to ensure our solution is successful, we have included tests to ensure the fail-safe precise nature of our solution. Finally, another concern may be the electrical safety of our project as we use microcontrollers and lithium-ion batteries. To avoid hazards like short circuits and water damage, we will ensure our project meets industry standards and certifications for electrical safety. We will make sure our solution is water resistant by including proper casing and isolation for components that may be exposed to moisture. To ensure electrical safety, we will follow the International Electrotechnical Commission (IEC) as they publish global standards for electrical and electronic technologies. We will follow the IEC standards to implement overcurrent protection circuits/fuses to prevent electrical overloads. We will also follow IEC standards and implement a Battery Management System to safeguard our lithium-ion batteries against overheating and thermal risks.