ECE445 Design Document

Project Leaflink

Team #43

Praveen Natarajan (pn17)
Hassan Shafi (hashafi2)
Hannah Pushparaj (hsp5)

TA: Aniket Chatterjee
Professor: Craig Shultz

Spring 2026

Table of Contents

ECE445 Design Document

Introduction

Problem

Indoor plants are a common way to decorate one's living space. They require careful watering and monitoring to make sure they stay alive. However, it is easy to forget to water these plants due to our busy lives or for example, taking a vacation. According to an article published by Medium, around 50% of plant owners kill their plants by forgetting to water them. The culmination of these issues leads to people wanting plants that are less dependent on constant watering. This problem is addressed by watering systems. However, these systems are simply programmed without sensing important features such as soil moisture and thus run the risk of overwatering.

Solution

Our project provides a standalone device that will automatically sense the moisture level of the soil, and deploy a pump that supplies the plant with just the right amount of water to survive. It will use an onboard soil moisture sensor along with a water pump to supply the plant with water. We will implement a simple light that shows the system's status (normal, watering, or needs attention). It will also include basic safety limits that will prevent the pump from running indefinitely. It will warn the user if the water container is empty or if the device isn't able to pump water properly. The device will store a basic history of when it watered the plant so the user can see that it's working. It will also utilize a companion app that will allow the user to monitor the current soil measure as well as show a log of recent watering. It will allow the user to trigger a manual watering from their phone if needed as well as select the water level.

Visual Aid

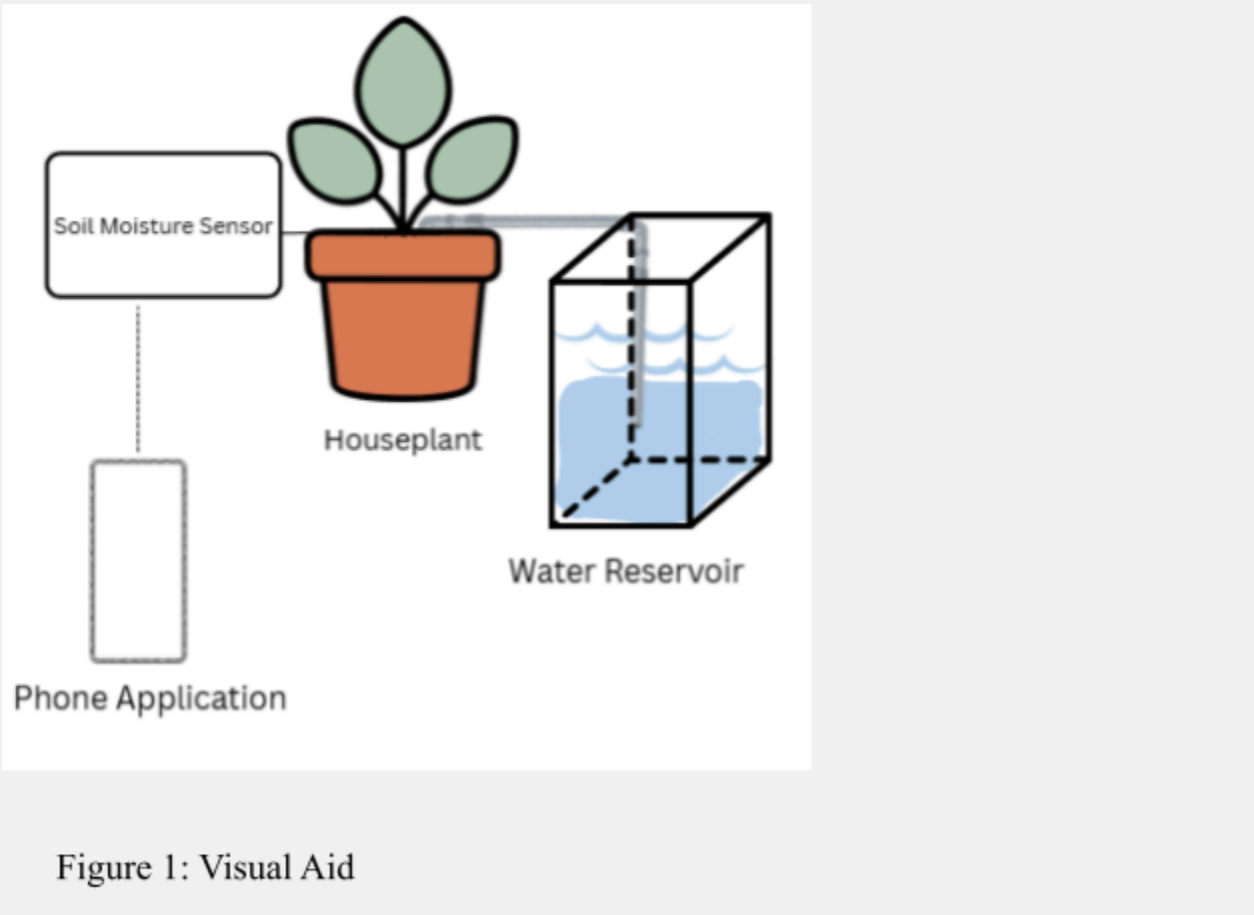


Figure 1: Visual Aid

High Level Requirements

- The ESP32 on our custom PCB correctly reads soil moisture data and determines when watering is required independently (requiring no supervision)
- Ensure proper functionality of the soil moisture sensor by ensuring moisture readings are accurate (for example if we add water the moisture percentage should get higher)
- The ESP32 reliably controls the relay to turn the water pump on and off based on soil moisture thresholds.

- The water pump operates only through the relay and correctly distributes the required amount of water
- The multiple LEDs correctly indicate the current system states, including idle, watering, and error.
- Pressing the emergency stop button immediately cuts power to the water pump and halts any ongoing operation
- Remote monitoring system displays accurate real-time soil moisture data, logs watering events, and allows manual watering control

Design

Testing Overview:

To keep the design simple, the ESP32 should have the following profiles for 3 different plants corresponding to plants that need less, moderate, and a lot of water. To quantify the dosage, we will use the following table to dispense specific amounts of water when necessary to meet the moisture threshold of each plant. We will be using a standard 8in diameter flowerpot for testing. The date in the following table is approximated from the watering requirements of the listed plants found online.

Category	Plant	Moisture Sensor Threshold
Dry	Haworthia	10%

Moderate	Spider plant	25%
Wet	Nerve plant	50%

Table 1: Table for different moisture thresholds

If time permits, more plants can be added to the table. However, most plants should survive with the 3 different categories of watering (Dry, Moderate, Wet) unattended.

Block Diagram

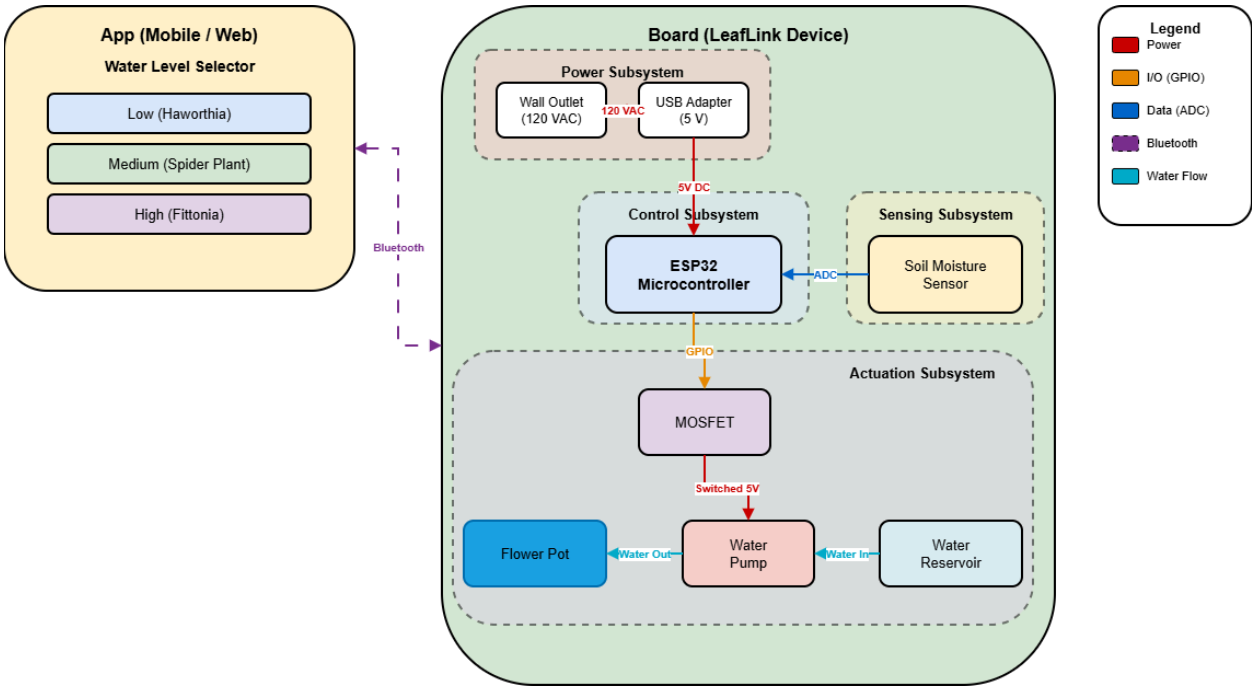
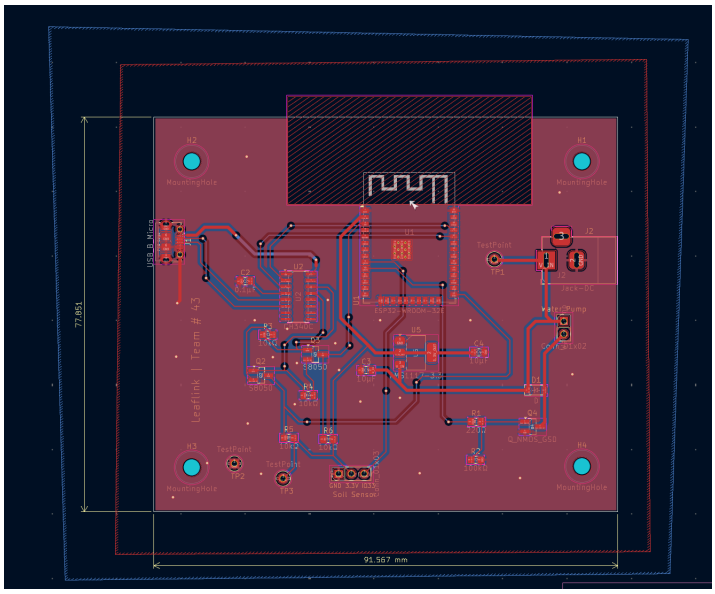


Figure 2: Project Block Diagram

Physical Design



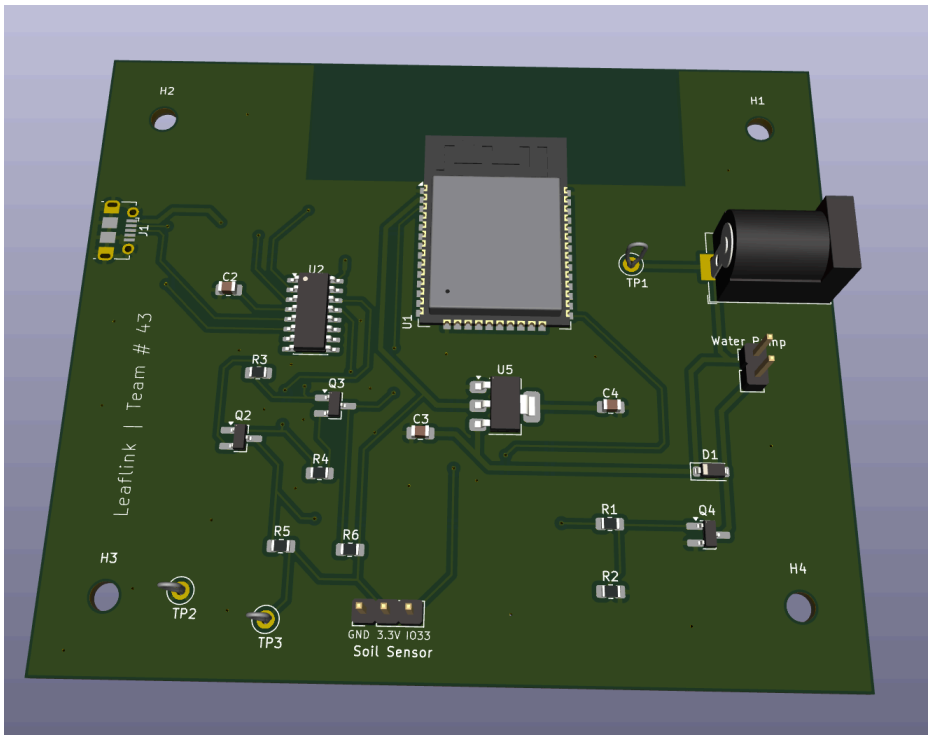


Figure 3 & 4: Render of PCB

Above is the Leaflink's PCB schematic and a rendering. We made sure to follow the PCB checklist to ensure our PCB will be functional and the berber files will go through PCBWay's audit.

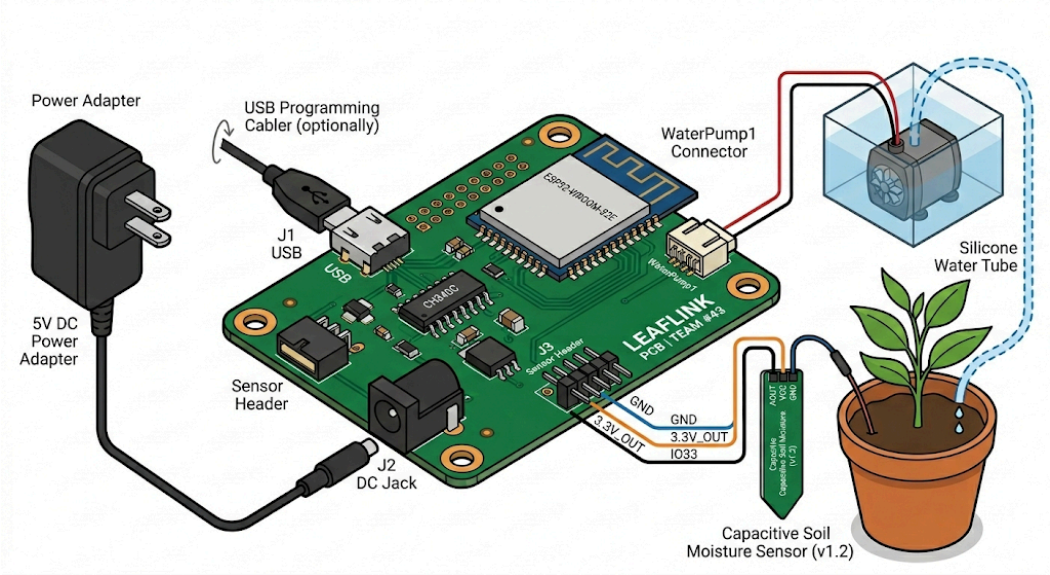


Figure 5: Full PCB Design

Above is an AI generated image of our entire system showing our numerous subsystems working together to power the Leaflink.

Subsystem Overview

Power Subsystem

The Power Subsystem converts power from the wall outlet into stable DC voltages needed by LeafLink. The ESP32 microcontroller and relay module are powered by a USB adapter that receives power from an AC wall outlet and outputs 5 V DC. This subsystem powers all of the other subsystems and connects directly to the Control Subsystem (5 V input), the Actuation Subsystem (switched 5 V to the pump), and indirectly supports sensing and wireless operation by delivering power in a controlled way.

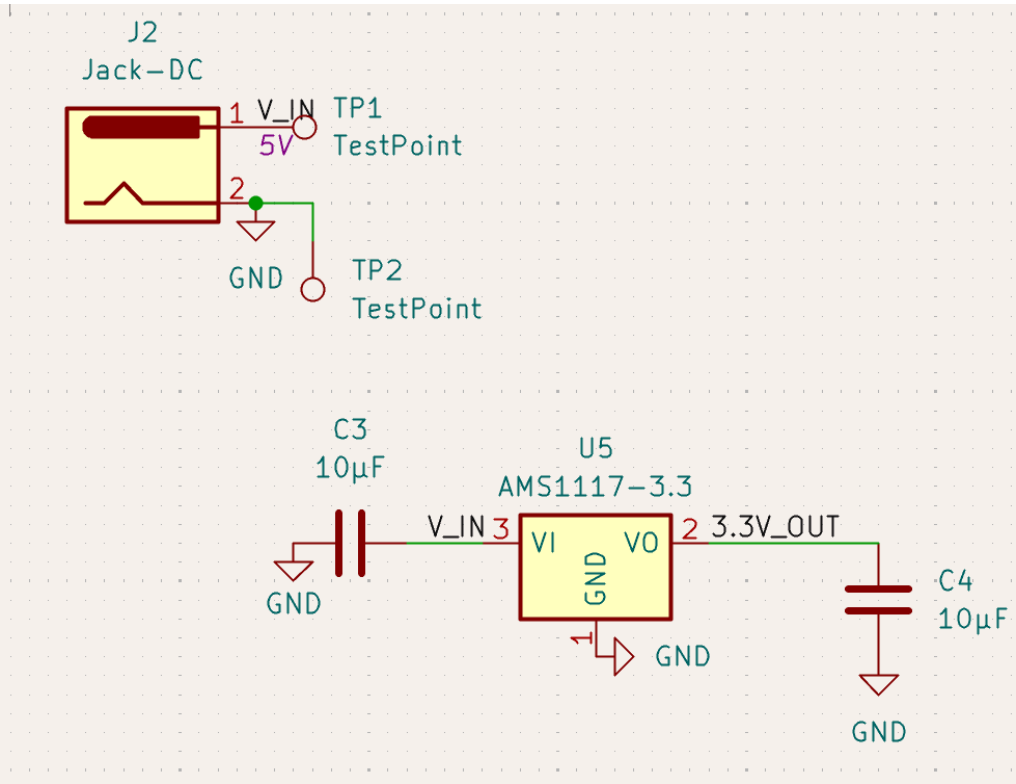


Figure 6: Power Subsystem

This block is our board's power input and 3.3 V regulation. The DC barrel jack (J2) brings in an external 5 V supply on the net V_IN (pin 1), with pin 2 tied to GND; TP1 and TP2 are test points so we can probe the 5 V rail and ground with a multimeter. The 5 V V_IN feeds the AMS1117-3.3 (U5) linear regulator, which outputs 3.3V_OUT for the rest of our circuit. C3 (10 µF) on the input and C4 (10 µF) on the output are decoupling/stability capacitors that smooth the voltage and help prevent the regulator from oscillating or dipping during load changes.

Requirements	Verification
Must be able to accept 5V as input and output 3.3 V	1. Connect a regulated 5.0 V supply to J2 (V_IN to pin 1, GND to pin 2). 2. Measure voltage at TP1 (V_IN) to confirm ~5.0 V. 3. Measure voltage at 3.3V_OUT (or the

	regulator output pin / 3.3V net) to confirm 3.3 V ± tolerance 4. Power the system and repeat steps 2–3 to confirm the 3.3 V rail stays in range under load. 5. Test to see if the power is sustained for long periods of time
Test points for measuring voltages	1. With power applied, probe TP1 and TP2 using a multimeter and confirm we can directly measure V_IN (~5 V) referenced to GND 2. Use TP2 as ground reference and measure 3.3V_OUT elsewhere on the board to confirm the test points are usable for debugging
Must include on/off switch	1. Toggle the switch ON and confirm the device powers up 2. Toggle the switch OFF and confirm the device fully powers down (no serial output/LED, pump cannot run). 3. With the switch OFF, measure 3.3V_OUT and confirm it is ~0 V

Table 2: Power Subsystem– Requirements & Verification

Control Subsystem

The Control and Processing Subsystem is essentially the core logic and decision-making component of LeafLink. The ESP32 microcontroller processes analog soil moisture inputs, executes watering control logic, and handles wireless communication with the app. The interfacing components of this subsystem are the Sensing Subsystem through the ADC, the Actuation Subsystem through GPIO signaling to the relay, the User Interface through status signals, and the Wireless Monitoring Subsystem through its interface.

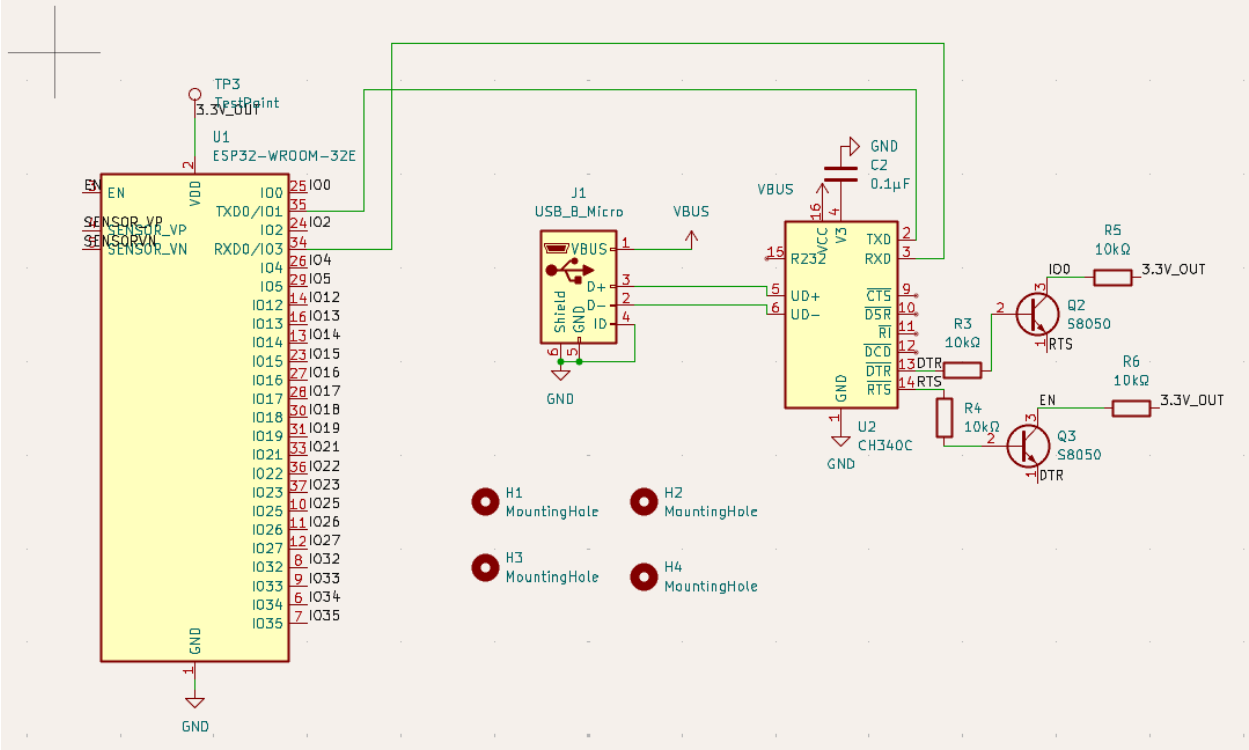


Figure 7: Control Subsystem

This section is our ESP32 core + USB programming interface. The ESP32-WROOM-32E (U1) is the main controller, powered from the 3.3V_OUT rail (with TP3 as a 3.3 V test point) and tied to a common GND. A micro-USB connector (J1) brings in USB signals (D+ and D–) and VBUS, which feed the CH340C (U2) USB-to-UART chip so we can plug the board into a computer for serial communication and uploading code. The CH340C's TXD/RXD lines connect to the ESP32's UART pins, and the S8050 transistors (Q2, Q3) with resistors form the auto-program/reset circuit that toggles EN and the boot pin (IO0) so flashing can happen automatically. The small capacitor (C2) provides local decoupling for stable operation, and the mounting holes are just mechanical features for securing the PCB.

Requirements	Verification

Must support USB programming and serial communication through the CH340C	1. Connect the board to a PC via micro-USB 2. Confirm the PC detects a USB serial (COM) device 3. Open a serial terminal at the expected baud rate and confirm readable boot/log messages 4. Confirm board is able to be programmed correctly with simple test program
Must provide stable 3.3 V power to ESP32 during USB connection and normal operation	1. Power the board normally and measure 3.3V_OUT at TP3 2. Connect board to phone, and determine if voltage stays stable
Must maintain connection to phone during operation	1. Pair the ESP32 with the phone over Bluetooth 2. Leave the system running for 10–30 minutes while streaming/periodically sending data 3. Verify the connection does not drop 4. Move the phone 5–10 m away, and confirm connection is still present

Table 3: Control Subsystem– Requirements & Verification

Sensing Subsystem

The Soil Moisture Sensing Subsystem detects the amount of moisture within the soil with the help of a capacitive soil sensor. The sensor generates an analog voltage signal proportional to the detected soil moisture, which is then detected by the ESP32's ADC. It acts as the primary input source for the system to water the plants and directly connects to the Control Subsystem via an analog signal wire and ground

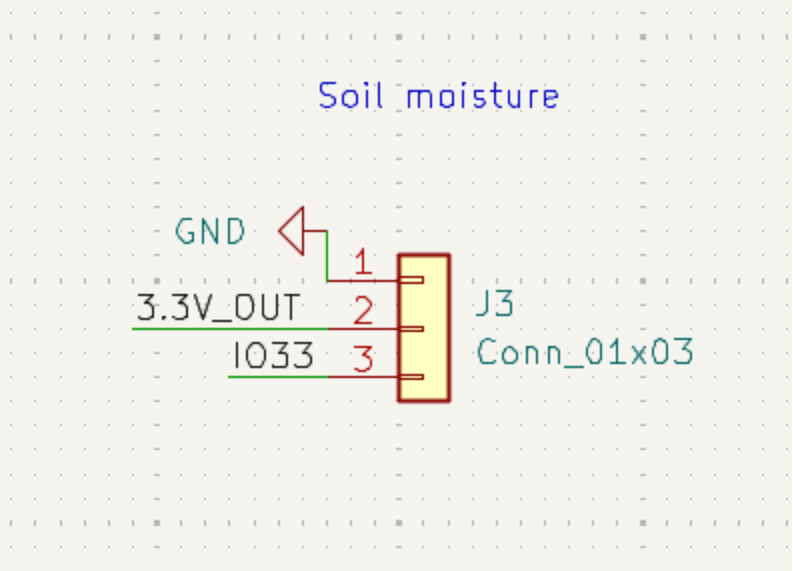


Figure 8: Sensor Subsystem

Above is the soil moisture sensor responsible for determining the moisture level of the soil for analysis. It is connected directly to the 3.3V pin and an IO to the ESP32. In addition, we will have a casing to house the moisture sensor so that it will not get damaged for prolonged periods of time.

Requirements	Verification
Calibrate the moisture percentage	1. Hook the sensor to a tub of water 2. Hook the sensor to dry soil 3. Map the readings in accordance to the following formula a. **percent = (dry - reading) / (dry - wet) * 100** b. Clamps the values between 0%-100% 4. Use these values for testing section below

Table 4: Sensor Subsystem– Requirements & Verification

Actuation Subsystem

The Water Delivery and Actuation Subsystem is what physically delivers water to the plant when commanded by the control logic. A relay module acts as an electrically isolated switch between the ESP32 and the water pump. When activated via GPIO, the relay supplies power to the pump, which draws water from the reservoir and delivers it to the flower pot. This subsystem interfaces with the Control Subsystem (GPIO), the Power Subsystem (pump supply), and the physical plant environment.

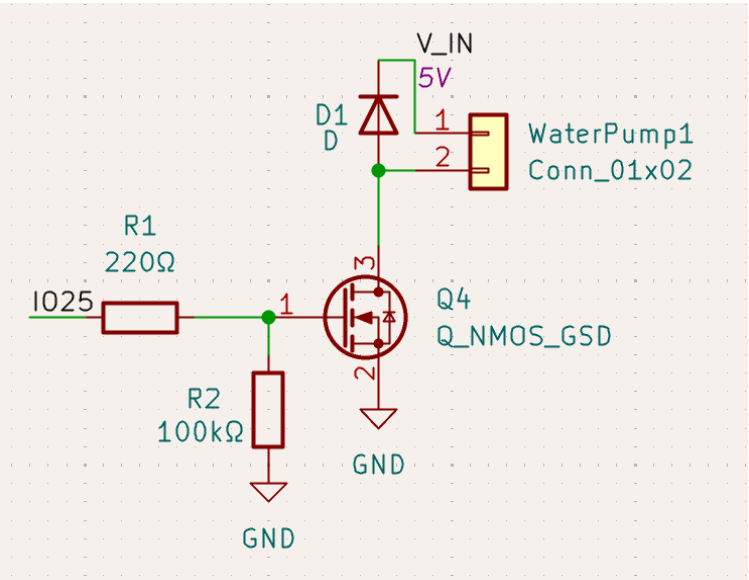


Figure 9: Actuation Subsystem

Above is the actuation block that lets the ESP32 safely switch the 5 V water pump on and off. The pump plugs into WaterPump1, with one side tied to V_IN (5 V) and the other side routed to the drain of Q4, an N-MOSFET used as a low-side switch (its source goes to GND). When the ESP32 drives IO25 high, the signal goes through R1 (220 Ω) to the MOSFET gate, turning Q4 on and completing the path to ground so current flows and the pump runs; when IO25 is low, R2

(100 kΩ) pulls the gate down so the pump stays off. D1 is a flyback diode across the pump that protects the MOSFET and the rest of the circuit from voltage spikes caused by the pump's inductive load when it switches off.

Requirements	Verification
Switch on the 5V pump using GPIO	1. Connect the pump to WaterPump1 and apply 5 V to V_IN 2. Flash firmware that sets IO25 HIGH for 3 s, LOW for 3 s in a loop 3. Confirm the pump turns ON when IO25 = HIGH and OFF when IO25 = LOW for at least 10 cycles
Maintain acceptable voltage to the pump	1. Turn pump on 2. Measure voltage across pump terminals and see if it is close to 5V
Must maintain repeated switching events without failure	1. Run a stress test toggle pump ON/OFF 20+ times 2. Confirm no missed activations, no ESP32 resets, and no component discoloration 3. Re-run a normal ON command after the test and confirm pump still runs normally

Table 5: Actuation Subsystem– Requirements & Verification

App Subsystem

The app subsystem allows users to monitor and interact with the LeafLink device remotely. Using the ESP32's built in Wi Fi, the system sends soil moisture data and watering history to our simple mobile/web interface. The app also allows users to manually trigger watering when needed and even select the type of plant to water accordingly. This subsystem connects directly

to the control subsystem through the ESP32 and does not affect core watering functionality if it is unavailable. To create the app, we will use the ESP32's bluetooth connectivity along with MIT AppInventor. There is a reference on how to do that here that we can consult.

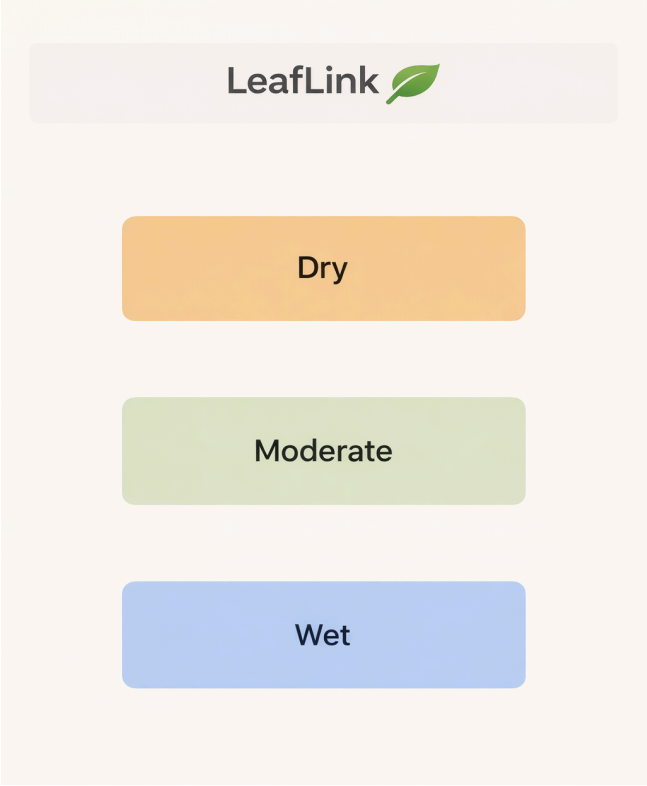


Figure 10: App Mock-up

Above is a mockup of the app screen. The user can select one of the 3 buttons to select the moisture level of the plant, and this will determine how frequently the plant will get watered.

Subsystem Requirements

Power Subsystem

- Converts wall outlet power into a steady 5 V DC supply for the system.
- Provides enough current to support the ESP32, relay, and pump during normal operation.
- Keeps the 5 V output stable while the pump turns on and off.

- Uses a shared ground so all parts of the system reference the same voltage level.

Control Subsystem

- Reads soil moisture data at least once per second using the ESP32.
- Runs watering decisions automatically without needing user input.
- Sends control signals from the ESP32 GPIO pins to the relay.
- Limits how long the pump runs during each watering cycle to prevent overwatering.
- Continues operating even if the wireless feature is unavailable.

Sensing Subsystem

- Produces an analog signal that the ESP32 can read reliably.
- Gives consistent readings when soil conditions stay the same.
- Changes output clearly as soil becomes wetter or drier.
- Operates safely from the system's 3.3 V supply.

Actuation Subsystem

- Relay responds correctly to control signals from the ESP32.
- Relay keeps the ESP32 electrically separated from the pump power circuit.
- Pump turns on only when the relay is activated.
- Pump delivers enough water to noticeably increase soil moisture.
- Emergency stop immediately shuts off pump power when pressed.

App Subsystem

- Displays accurate and real time soil moisture readings from the ESP32.
- Must have a history page where it shows recent watering events statistics.
- Allows the user to manually trigger watering through the interface.
- Sends user commands to the ESP32 through Bluetooth and can specify the specific type of plant being watered to adjust watering accordingly.
- Does not interfere with autonomous watering if the connection is lost.

Tolerance Analysis

Sensor Measurement Error

The LeafLink system measures soil moisture using the DFRobot SEN0193 Capacitive Soil Moisture Sensor. This sensor outputs an analog voltage proportional to soil moisture content. Unlike precision industrial probes, the manufacturer does not specify a fixed accuracy. However, calibration studies and laboratory testing show that low-cost capacitive sensors of this model typically exhibit moisture measurement errors on the order of:

$$\Delta Msensor \approx \pm 4\% \; to \; \pm 12\%$$

depending on soil composition, temperature, and calibration.

For tolerance analysis, a ±5% representative uncertainty is used when the sensor is calibrated for the target soil, which aligns with reported RMSE values near 4–5%.

Let

$$Mactual = true\ soil\ moisture$$

$$Mmeasured = Mactual \pm \epsilon sensor$$

Where

$$\epsilon sensor \approx \pm 5\%$$

Watering is triggered when

$$Mmeasured < Mthreshold$$

Therefore, worst-case activation occurs at:

$$Mactual = Mthreshold \pm 5\%$$

This defines the uncertainty band around each threshold.

10% Threshold (Haworthia – Dry)

$10\% \pm 5\% \rightarrow 5\% \; to \; 15\%$

Relative uncertainty: $5/10 = 50\%$

This is the limiting design case, since tolerance is large relative to the threshold.

25% Threshold (Spider Plant – Moderate)

$25\% \pm 5\% \rightarrow 20\% \; to \; 30\%$

Relative uncertainty: 20%

Acceptable for moderate-moisture plants.

50% Threshold (Nerve Plant – Wet)

$50\% \pm 5\% \rightarrow 45\% \; to \; 55\%$

Relative uncertainty: 10%

Least sensitive to measurement uncertainty.

System accuracy is primarily limited by sensor calibration accuracy and soil variability and placement depth. Low-cost capacitive sensors provide medium accuracy suitable for irrigation

control but require calibration. Thus, system performance is governed primarily by sensor calibration and control strategy, rather than precision hardware.

Cost and Schedule

Cost Analysis

The total cost before shipping, as seen below is **$64.10.** Approximate 5% shipping cost, the total would come out to **$66.53.** A 10% sales tax brings the total to: **$74.51**. We can expect a salary of $40 / hr. The total for one person would be 40 * 2.5 * 40 = $4000. For all 3 members, the total would be 4000*3 = 12000. Therefore, the total cost for this project would be **$12074.51**

Itemized Component List

Description	Manufacturer / Vendor	Qty	Unit Price	Extended Price	Link
0.1 uF Capacitor	Adafruit	1	$0.195	**$1.95**	Link
10 uF Capacitor	Adafruit	2	$0.195	**$1.95**	Link
Diode	Adafruit	1	$0.150	**$1.50**	Link
USB Micro Connector	Adafruit	1	$1.95	**$1.95**	Link
DC Jack	Adafruit	1	$2.95	**$2.95**	Link
Capacitive Moisture Sensor	Digikey	2	$5.90	**$11.80**	Link
N-channel power MOSFET	Adafruit	1	$2.25	**$2.25**	Link

S8050 NPN Transistor	DigiKey	2	$0.08860	**$0.1772**	Link
220Ω Resistor	Adafruit	1	0.75	**$0.75**	Link
100kΩ Resistor	Adafruit	1	0.95	**$0.95**	Link
10kΩ Resistor	Adafruit	4	0.75	**$0.75**	Link
ESP32-WROOM	Adafruit	1	3.25	**$3.25**	Link
CH340C Serial Breakout	Amazon	1	8.95	**$8.95**	Link
AMS1117-3.3 Voltage Regulator	Adafruit	1	1.25	**$1.25**	Link
3.3V water pump	Adafruit	1	2.95	**$2.95**	Link
ESP32 Dev Board	Adafruit	1	15.00	**$15.00**	Link
Solderless Breadboard	Adafruit	1	4.95	**$4.95**	Link

Table 5: Component List

Schedule

Week	Task	Person
March 2 – March 9	Order components (soil moisture sensor, pump, MCU, LEDs, reservoir parts)	Everyone
	Finalize schematic design	Everyone
	Select moisture sensor & pump specifications	Hannah
	Define app requirements & communication protocol	Hassan

	Finish PCB layout	Praveen
	PCB ORDER	Everyone
March 9 – March 16	Assemble prototype board	Everyone
	Test power regulation & pump driver circuit	Hannah
	Integrate soil moisture sensor (basic read functionality)	Praveen
	Develop basic firmware structure	Hassan
	PCB ORDER	Everyone
March 16 – March 23	Implement automatic watering control logic	Praveen
	Add safety timeout to prevent continuous pump operation	Hassan
	Implement LED status indicators (normal / watering / error)	Praveen
	Begin mobile app development (display moisture level) Hardware debugging and PCB revisions	Hannah Everyone
March 23 – March 30	Integrate water level detection (empty reservoir warning)	Praveen
	Test pump reliability & flow rate calibration	Praveen
	Implement watering history storage	Hassan
March 30 – April 6	Full system integration (sensor + pump + app)	Everyone
	Implement manual watering feature via app	Praveen
	Optimize moisture threshold calibration	Hassan
	Long-duration watering tests	Hannah
April 6 – April 13	Fix integration bugs	Everyone
	Validate safety features (timeout, dry pump protection)	Hannah
	Finalize app interface (log display + moisture monitor)	Praveen

	Enclosure assembly	Everyone
April 13 – April 20	System validation testing against requirements	Everyone
	Data logging verification	Praveen
	Stress testing (multiple watering cycles)	Everyone
April 20 – April 27	Final debugging & polish	Everyone
	Demo preparation & test documentation	Everyone
	Final presentation & submission	Everyone

Table 6: Schedule

Ethics and Safety

This project involves the design of an automated plant watering system using an ESP32, soil moisture sensor, water pump, and companion mobile application. While the system is intended for benign household use, several ethical considerations arise during both development and deployment.Leaflink aligns with the IEEE code of ethics by respecting the importance of public safety through safety measures. We will provide an emergency stop button to prevent damage to the plant, water waste, and etc. Furthermore, our system will be honest by providing reliable data according to its sensors. This is in accordance with the IEEE code of ethics that requires honesty and transparency. The collected data will be stored securely and not shared with anyone but the consumer. All electrical components will follow industry standards in order to ensure public safety. There will be no exposed wires and they will all be properly grounded.

During development, we will also make sure to unplug the system every time we need to work on it to avoid contact with direct power. We will accept criticisms from peers as well as TA and others that oversee our work. In accordance with campus policy and the IEEE code of ethics, we will be respectful of each other and everyone we interact with.

Since plants improve air quality, Leaflink is a useful contribution to the welfare of the public as well as to the environment by making it easier to maintain plants. According to a study published in the Journal of Physiological Anthropology, indoor plants have been shown to reduce stress. Leaflink also prevents these plants from dying due to under or overwatering and eases the responsibilities of plant owners.

Works Consulted

Gardener's Supply Co. "How to Care for a Nerve Plant." *Gardeners Supply*, 2025, www.gardeners.com/blogs/gardening-tips/nerve-plant-care-9725.

Grossman, Karen Brewer. "The Right Way to Water a Spider Plant so It Will Thrive." *Southern Living*, Southern Living, 3 Feb. 2026, www.southernliving.com/right-way-to-water-spider-plant-11898236.

"How to Care for the Haworthia Succulent: Plants 101." Edited by The Still, *The Sill*, The Sill, 7 Aug. 2019, www.thesill.com/blogs/plants-101/how-to-care-for-haworthia.

Planter, Better. "Houseplant Statistics. Top 25 Houseplant Statistics | by Better Planter | Medium." *Houseplant Statistics*, 2024, medium.com/@betterplanter/houseplant-statistics-b105e4e276a3.

OpenAI. *ChatGPT*, 12 Feb. 2026, https://chat.openai.com

Adafruit. Adafruit Industries, 27 Feb. 2026, https://www.adafruit.com/

Stanborough, Rebecca Joy. *"7 Science-Backed Benefits of Indoor Plants." Healthline*, 18 Sept. 2020, www.healthline.com/health/healthy-home-guide/benefits-of-indoor-plants. Accessed 27 Feb. 2026.