# LEAFLINK

By

Praveen Natarajan

Hannah Pushparaj

Hassan Shafi

Final Report for ECE 445, Senior Design, Spring 2026

TA: Aniket Chatterjee

2 May 2026

Project No. 43

## Abstract

LeafLink is an automated plant watering system designed to monitor soil moisture and water plants as needed. The system utilizes a capacitive soil moisture sensor and a microcontroller to measure soil conditions and control a water pump for automated irrigation. By reducing the need for continuous user monitoring and maintenance, LeafLink simplifies plant care while helping prevent underwatering and overwatering. The system also includes a companion mobile application that allows users to view watering logs, adjust soil moisture thresholds, schedule watering events, and manually activate watering. Overall, LeafLink demonstrates a low-cost and user-friendly solution for smart plant maintenance.

# Contents

# 1. Introduction

Indoor plants are a popular way to decorate one's homes. Not only are they visually appealing, but they also provide many health benefits such as lower levels of stress, longer attention spans, and improved cognitive functions. [1] One survey found that approximately 37.6 US households have indoor plants. [2] This statistic demonstrates the significance of indoor plants in everyday life. Despite their prevalence, nearly 50% of plant owners forget to water their plants, due to busy schedules, traveling or plain forgetfulness. [3] On the other hand, some kill their plants by overwatering. Either way, many find it difficult to properly take care and maintain their plants.

Leaflink offers a solution by eliminating the need for constant monitoring by plant owners due to its automated watering system. The system employs a capacitive soil moisture sensor that works by outputting an analog voltage proportional to the dielectric permittivity of the surrounding soil. Water has a higher dielectric constant than air or dry soil. When soil moisture increases so does the dielectric constant thus increasing the capacitance between the two conductive electrodes connected to the sensor. When capacitance is low, the resonant frequency increases and vice versa. The sensor measures this change and converts it into an analog signal. [4]  This signal is sent to the microcontroller, in our case the esp32, and processed according to the calibration levels. If the soil moisture level is below the threshold set by the user in the companion app, the esp32 will trigger the pump to deploy water until it reaches the correct level. Additionally, an app allows users to monitor soil moisture levels, schedule watering events, set moisture thresholds, and manually trigger watering. The final design is shown in Figure 1.1.

Figure 1: Leaflink System

### 1.1 Block Diagram

The block diagram for Leaflink is shown in Figure 2. It consists of five subsystems; the power, control, sensing, actuation, and app subsystems. The power subsystem consists of a buck converter that steps down the wall outlet voltage to 3.3 V so that it can supply power to the control subsystem. The control subsystem receives a signal from the sensing subsystem, and the app via the wifi component. This subsystem also executes the control logic; if soil moisture is lower then the threshold than the pump is on and outputs a GPIO signal to the actuation subsystem. The latter employs a mosfet that acts as a switch to control the water pump which deploys water to the plant. Table 1 lists the high level requirements for the system.

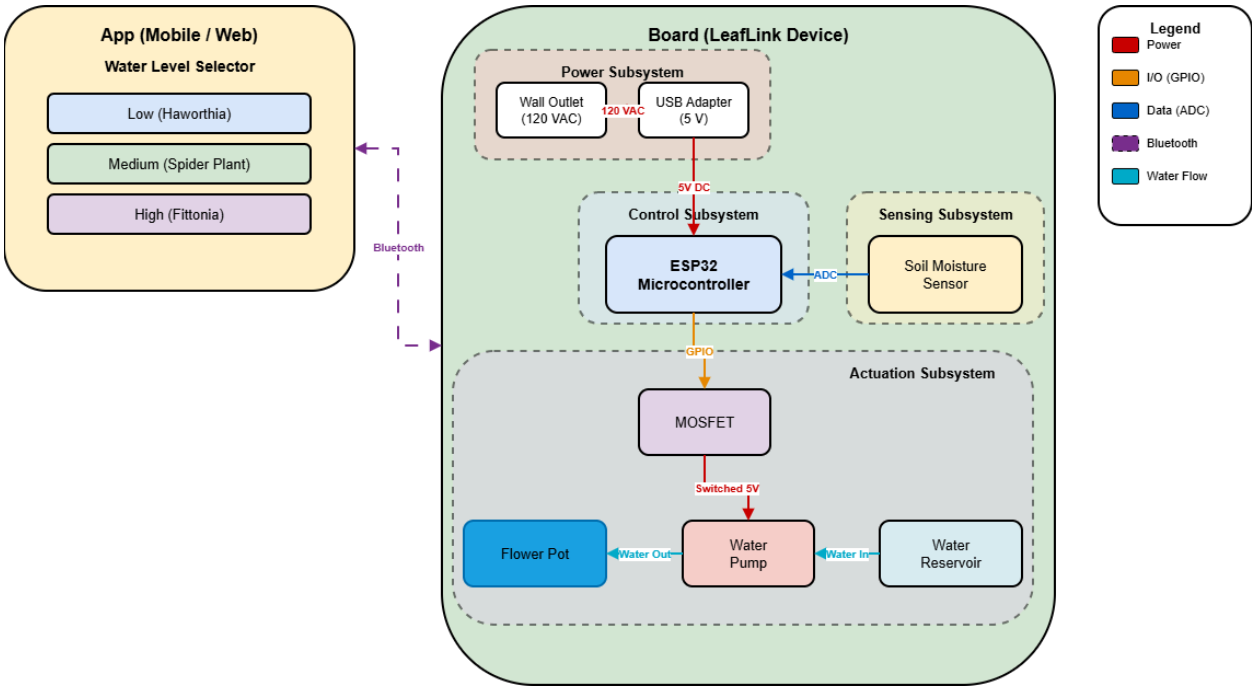


Figure 2: Final Block Diagram

| Requirement | Description |
|---|---|
| Autonomous Operation | The ESP32 on the custom PCB reads soil moisture data and determines when watering is required without supervision. |
| Sensor Accuracy | The soil moisture sensor provides accurate readings (e.g., moisture percentage increases when water is added). |
| MOSFET Control Reliability | The ESP32 reliably controls the MOSFET to switch the water pump on and off based on soil moisture thresholds. |

| | |
|---|---|
| Pump Operation | The water pump operates only through the MOSFET and delivers the required amount of water. |
| Remote Monitoring & Control | The system displays real-time soil moisture data, logs watering events, and allows manual watering control. |

Table 1: Requirements and Description

## 2.2 Design

### 2.2.1 Electronic Design

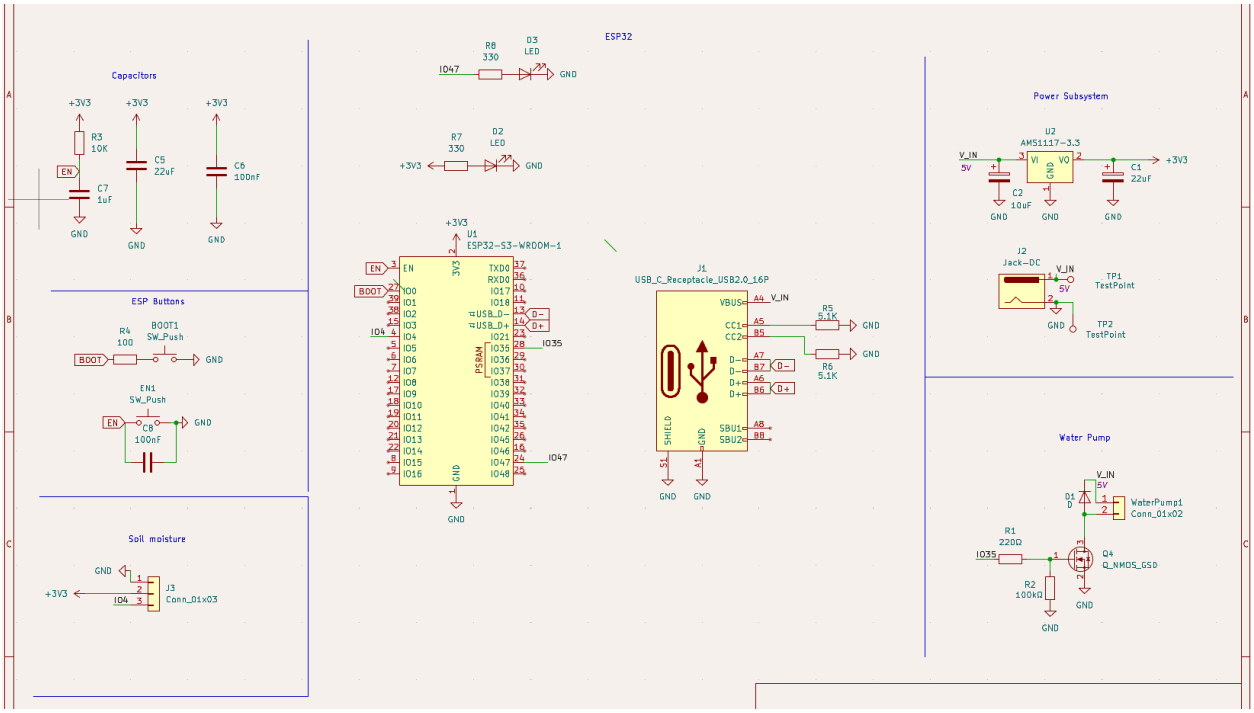


Figure 3: PCB Schematic

**Control Subsystem**

- Design Procedures

  As shown in figure 3, the ESP32 wiring was simplified by replacing the older ESP32-WROOM-32E and external CH340C USB-to-serial programming circuit with an ESP32-S3-WROOM-1, which supports native USB communication. In the previous design, the Micro-USB connector connected to the CH340C, and the CH340C then routed TXD and RXD signals to the ESP32. Additional DTR and RTS transistor circuitry was also required to control the ESP32 EN and boot/reset behavior. In the current design, the

USB-C connector routes D+ and D- directly to the ESP32-S3 USB_D+ and USB_D- pins, removing the need for the CH340C converter and its supporting components.

The USB connector was also changed from Micro-USB to USB-C to improve usability and simplify power/programming access. The USB-C VBUS pin connects to the board's V_IN net for 5 V input power, while the shield and ground pins connect to GND. The CC1 and CC2 pins each use 5.1 kΩ pull-down resistors to identify the board as a USB device. Overall, the updated wiring reduces the number of components, removes unnecessary UART converter and reset circuitry, simplifies routing, and makes the LeafLink PCB easier to assemble, program, and debug.

- Design Details

The current design uses an ESP32-S3-WROOM-1 as the main controller for the LeafLink system. The ESP32-S3 is powered from the board's 3.3 V rail, with its 3V3 pin connected to the regulated supply and its GND pins tied to the common ground. The EN and BOOT signals are broken out so the board can still be manually reset or placed into programming mode if needed. The USB data lines are connected directly to the ESP32-S3's native USB pins, with USB_D+ connected to the USB-C connector's D+ pins and USB_D- connected to the USB-C connector's D- pins. This allows the ESP32-S3 to handle programming and serial communication without an external USB-to-UART chip.

The USB-C receptacle provides both power input and communication for the board. Its VBUS pin is connected to the V_IN net, which supplies 5 V input power to the rest of the circuit. The connector ground and shield pins are connected to GND for a shared electrical reference and improved stability. The CC1 and CC2 pins each have a 5.1 kΩ pull-down resistor to ground, allowing the USB-C port to be recognized as a device when plugged into a computer or power source. The design also includes indicator LEDs: one LED is tied to the 3.3 V rail as a power indicator, while another is connected to IO47 to allow the ESP32 to control a status signal.

**Actuation Subsystem**

- Design Procedures

As shown in figure 3, this circuit controls the operation of the water pump that supplies water to the plant. The first prototype of the circuit used the relay module as it was easy to solder and could test the pump control logic rapidly. Nevertheless, the relay module was bulky, and the pump did not work stably when powered through the same source as the microcontroller. As a result, it became clear that there had to be a dedicated 5V line for the pump.

In the final circuit on the PCB, the relay module was substituted by a MOSFET low-side switch. This circuit is more compact and can easily be integrated into the circuit board. The ESP32-S3 microcontroller does not provide power directly to the pump. The GPIO pin of the controller controls the pump current supplied through the MOSFET.

- Design Details

The final actuation circuit employs an N-channel MOSFET as a low side switch. One end of the pump is connected to the 5 V supply, while the other end is connected to the MOSFET drain. Finally, the MOSFET source is grounded. When the ESP32-S3 GPIO pin switches the MOSFET gate on, the motor starts to operate; if it switches the gate low, then it is turned off.

In addition, the design utilizes a gate resistor, a pulldown resistor, and a flyback diode. The former helps reduce the peak current at the moment of switching the MOSFET using the GPIO pin. Moreover, the latter ensures that, during the boot process and while the GPIO pin is left floating, the MOSFET remains off, preventing any accidental activation of the motor. The flyback diode is required to avoid high voltages caused by switching the motor off.

**Sensing Subsystem**

- Design Procedures

  As shown in figure 3, the sensing subsystem measures the moisture level of the soil and provides the control subsystem with an analog signal. Two main types of soil moisture sensors were considered: resistive and capacitive sensors. A resistive sensor is simple and inexpensive, but it can corrode over time because it relies on exposed conductive probes in wet soil. A capacitive sensor was selected because it is better suited for repeated use in soil and provides an analog voltage that can be read directly by the ESP32-S3 ADC.

  The sensor output changes as the dielectric properties of the soil change. Wet soil produces a different sensor reading than dry soil, allowing the ESP32-S3 to estimate the relative moisture level.

- Design Details

  The capacitive soil moisture sensor is connected to the 3.3 V supply, ground, and an analog input pin on the ESP32-S3. As the soil moisture changes, the sensor outputs a different analog value. The ESP32-S3 reads this value and converts it into a moisture percentage using dry and wet calibration values. During testing, the sensor was placed in dry soil and wet soil to determine the approximate range of readings. The final value was clamped between 0% and 100% so the system could display a clear moisture percentage in the app and compare it to the selected watering threshold.

**Power Subsystem**

- Design Procedures

  The power subsystem provides the voltage levels needed for the LeafLink board and pump. Since the system is meant to stay near an indoor plant for long periods of time, we chose a wall-powered design instead of a battery-powered design. A battery-powered system would be more portable, but it would require charging circuitry and would limit how long the system could run without maintenance. The final design

uses a 5 V input to power the pump and then regulates that voltage down to 3.3 V for the ESP32-S3 and other low-voltage components.

- Design Details

  The board receives 5 V through the power input, which is used directly for the water pump. The same 5 V rail is also connected to a 3.3 V voltage regulator, which supplies the ESP32-S3, moisture sensor, LEDs, and control circuitry. Input and output capacitors were placed near the regulator to help keep the voltage stable and reduce noise or voltage dips during operation. The PCB also includes test points for 5 V, 3.3 V, and ground so the power rails can be checked easily during bring-up. This was important because the power subsystem had to be verified before testing the microcontroller, sensor, or pump.

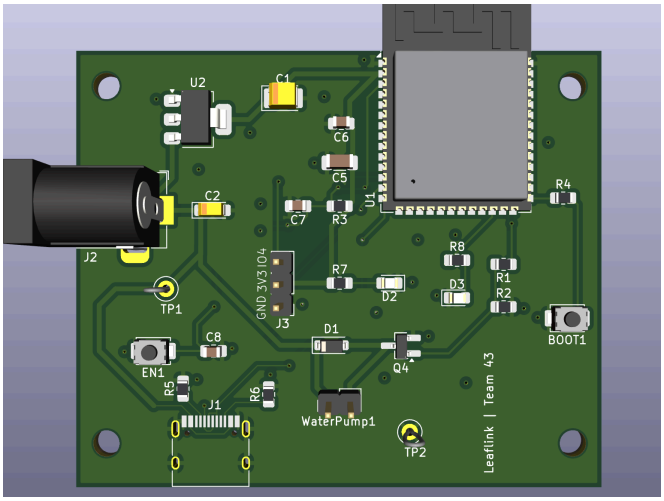


Figure 4: PCB Render

**Remote Monitoring Subsystem**

- Design Procedures

  The remote monitoring subsystem allows the user to view the plant's current soil moisture level and control watering from a phone or computer. We considered using an existing IoT platform for this feature because it would be quick to set up, but we decided to build a custom web-based interface instead. This gave us more control over the layout, user inputs, and how the data was displayed. The final design uses a React frontend for the user interface and a backend hosted on Render to handle communication between the app and the LeafLink device.

- Design Details

The React frontend displays the current soil moisture percentage, pump status, watering threshold, and manual watering controls. The backend hosted on Render receives data from the ESP32-S3 and makes it available to the frontend. When the user changes the watering threshold or presses the manual watering button, the command is sent through the backend and then used by the system to update its watering behavior. This setup allows the user to monitor the plant remotely while still keeping the main watering logic on the ESP32-S3, so the system can continue operating even if the remote interface is not actively being used.

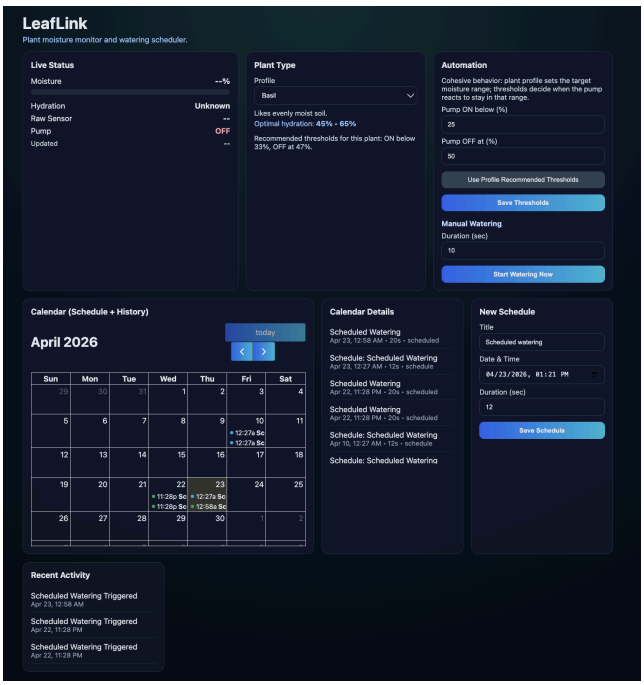


Figure 5: Remote Monitroing App

### 2.2.2 Physical Design

**Procedure and Design Details**

The LeafLink housing was designed as a two-part enclosure system: the reservoir, which stores water, and the housing/base assembly, which protects the PCB, wiring, pump connections, and cover. Separating the water storage from the electronics improves safety, organization, and maintainability. The CAD model shows the reservoir as the larger cylindrical component and the electronics housing as the smaller rectangular component attached to its side.

The main design goal was to create a compact, clean, and enclosed plant-watering system that could sit near an indoor plant. The reservoir provides enough water for long-term use, while the housing keeps the electronic components protected and accessible for testing, programming, and maintenance. Details about each component will be described below

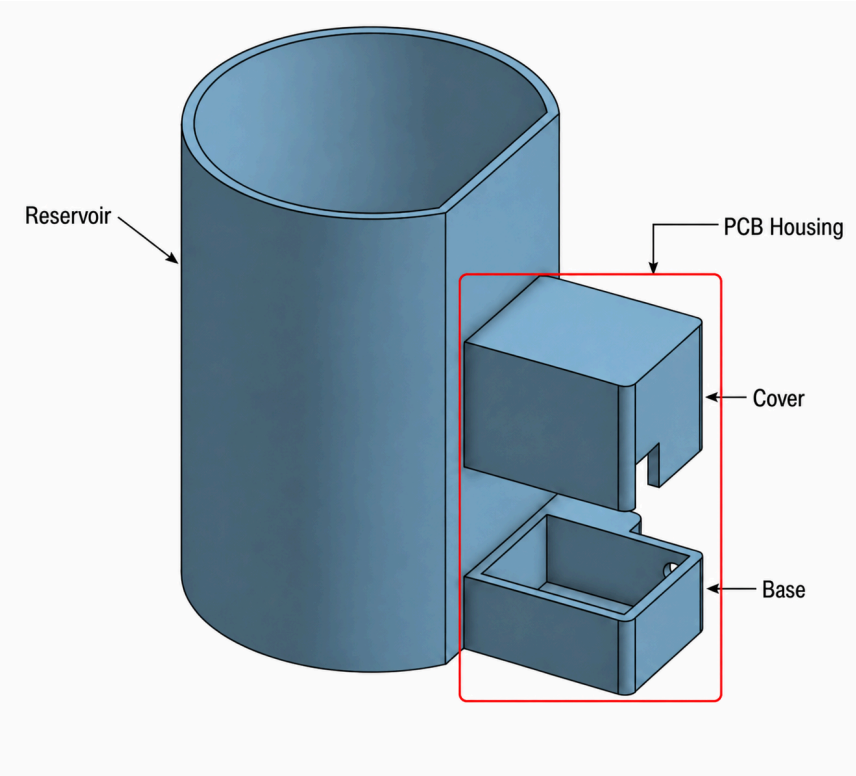


Figure 4: LeafLink Housing

### 2.2.2 Reservoir

The reservoir is the larger cylindrical section of the LeafLink housing. Its purpose is to store the water used by the pump to irrigate the plant. A cylindrical shape was selected because it provides high storage volume in a compact footprint and avoids sharp internal corners where water could collect or become difficult to clean. In addition, it should be big enough to have a plant fit inside of it, while still making it easy to refill the reservoir when the water runs out

The reservoir has an internal volume of approximately 2926 cm^3. Assuming that the plant takes about 1 m^3 inside of the reservoir, the volume left is about 1926 cm^3. Since water's density is 1 cm^3 = 1 mL, this reservoir can hold about 1.926 L.

Depending on the water cycle, this is more than enough to support an indoor plant. For example, on average each watering cycle uses about 150 mL of water (Varies based on moisture threshold)

**N = 1926 / 150 = 12.84 cycles**

This can provide several weeks of operation.

Waterproofing the reservoir is an important part of the physical housing design because it directly affects the reliability and safety of the LeafLink system. Since the reservoir holds the full water supply, any leakage could reduce the available water for the plant, damage nearby surfaces, or allow moisture to reach the PCB housing. To prevent this, the reservoir should have sufficient wall thickness, sealed seams, and a watertight interface where it connects to the rest of the housing. If the reservoir is 3D printed, the print may also need to be sealed with a waterproof coating or epoxy because small gaps between print layers can allow water to slowly seep through. A leak test should be performed by filling the reservoir and leaving it for several hours to confirm that no water escapes before placing electronics near the enclosure.

Alternative reservoir shapes were considered. A rectangular reservoir would be easier to print and assemble, but it would have sharper corners and a less polished appearance. A fully separate reservoir would improve water isolation, but it would make the product larger and require additional tubing. The final integrated cylindrical reservoir was chosen because it balances capacity, appearance, compactness, and ease of use.

**Housing/Base Assembly**

The housing is the smaller rectangular section attached to the side of the reservoir. It contains two main pieces: the base and the cover. The base holds the LeafLink PCB and wiring, while the cover fits over the base to protect the electronics from dust, water splashes, and accidental contact.

The base is designed with raised walls so the PCB can sit securely inside. It provides space for connectors, wiring, and cable routing. Cutouts or openings allow external connections to pass through the enclosure, including the soil moisture sensor, pump tubing or pump wiring, and programming/download cable.

The cover improves both safety and appearance. It hides exposed electronics while still allowing access when the system needs to be debugged or reprogrammed. A removable cover was chosen over a fully sealed enclosure because the system still needs to be accessible during development and maintenance.

Several housing alternatives were considered. An open electronics tray would make debugging easy, but it would leave the PCB exposed to water and soil. A fully sealed enclosure would provide better protection, but it would make programming, testing, and repairs harder. The final removable-cover design provides a balance between protection and accessibility.3. Design Details

Present the detailed design, with diagrams and component values. Show how the design equations were applied. Give equations and diagrams with specific design values and data. Place large data tables in an appendix. Circuit diagrams that are too large to be readable on a single page should be broken into pieces for presentation. The full diagram may be included in an appendix. Use photographs only as necessary and treat them, along with all other graphics except tables, as figures.

## 2.3. Design Verification

Table 2: Power Subsystem

| 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).<br>2. Measure voltage at TP1 (V_IN) to confirm ~5.0 V.<br>3. Measure voltage at 3.3V_OUT (or the regulator output pin / 3.3V net) to confirm 3.3 V ± tolerance<br>4. Power the system and repeat steps 2–3 to confirm the 3.3 V rail stays in range under load.<br>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 using a multimeter and confirm we can directly measure V_IN (~5 V) referenced to GND<br>2. Probe 3.3V_OUT from AMS-1117 using TP2 GND |

Table 3: Control Subsystem

| Requirements | Verification |
|---|---|
| Must support USB programming and serial communication | 1. Connect the board to a PC via USB-C<br>2. Confirm the PC detects a USB serial (COM) device<br>3. Open a serial terminal at the expected baud rate and confirm readable boot/log messages (115200)<br>4. Confirm board is able to be programmed correctly with simple test program |
| Must provide stable 3.3 V power to ESP32 | 1. Power the board normally and |

| | |
|---|---|
| during USB connection and normal operation | measure 3.3V_OUT from the AMS-1117<br>2. Connect board to phone while streaming data to remote application, and determine if voltage stays stable |
| Must maintain connection to phone during operation | 1. Pair the ESP32 with the phone over Bluetooth<br>2. Leave the system running for 10–30 minutes while streaming/periodically sending data<br>3. Verify the connection does not drop<br>4. Move the phone 5–10 m away, and confirm connection is still present |

Table 4: Sensor Subsystem

| Requirements | Verification |
|---|---|
| Calibrate the moisture percentage | 1. Hooked the sensor to a tub of water<br>2. Hooked the sensor to dry soil<br>3. Mapped the readings in accordance to the following formula<br>**a. percent = (dry - reading) / (dry - wet) * 100**<br>b. Clamped the values between 0%-100%<br>4. Used these values for testing section below |
| Maintain moisture at set threshold | 1. Ran two week test with basil plant with input threshold of 70%. Average soil moisture was 71%. |

Table 5: Actuation Subsystem

| Requirements | Verification |
|---|---|

| | |
|---|---|
| Switch on the 5V pump using GPIO | 1. Connect the pump to WaterPump1 and apply 5 V to V_IN<br>2. Flash firmware that sets IO25 HIGH for 3 s, LOW for 3 s in a loop<br>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<br>2. Measure voltage across pump terminals and see if it is close to 5V |
| Must maintain repeated switching events without failure | 1. Ran a stress test by toggling pump ON/OFF 20+ times<br>2. Confirmed no missed activations, no ESP32 resets, and no component discoloration<br>3. Re-ran a normal ON command after the test and confirm pump still runs normally |

## 2.4. Costs

### 2.4.1 Parts

Table 6 lists the costs for each part needed for the final design.

**Table 6 : Final Design Parts Cost**

| Part | Manufacturer | Retail Cost ($) | Quantity | Actual Cost ($) |
|---|---|---|---|---|
| N-Channel MOSFET SOT23-3L | Digi-Key Corporation | 0.52 | 5 | 2.60 |
| Tactile Switch SPST-NO | Digi-Key Corporation | 0.37 | 5 | 1.85 |
| Mini Test Point | Digi-Key Corporation | 0.183 | 10 | 1.83 |
| 22µF Tantalum Capacitor 16V | Digi-Key Corporation | 0.256 | 10 | 2.56 |

| | | | | |
|---|---|---|---|---|
| 1µF Ceramic Capacitor 50V | Digi-Key Corporation | 0.064 | 10 | 0.64 |
| 10µF Tantalum Capacitor 16V | 3Digi-Key Corporation | 0.313 | 10 | 3.13 |
| 0.1µF Ceramic Capacitor 50V | Digi-Key Corporation | 0.038 | 10 | 0.38 |
| ESP32-S3 Module | Digi-Key Corporation | 5.88 | 4 | 23.52 |
| Green LED 0805 | Digi-Key Corporation | 0.19 | 10 | 1.90 |
| 5.1kΩ Resistor 0805 | Digi-Key Corporation | 0.032 | 10 | 0.32 |
| 330Ω Resistor 0805 | Digi-Key Corporation | 0.032 | 10 | 0.32 |
| 100Ω Resistor 0805 | Digi-Key Corporation | 0.032 | 10 | 0.32 |
| 100kΩ Resistor 0805 | Digi-Key Corporation | 0.032 | 10 | 0.32 |
| 220Ω Resistor 0805 | Digi-Key Corporation | 0.032 | 10 | 0.32 |
| 10kΩ Resistor 0805 | Digi-Key Corporation | 0.032 | 10 | 0.32 |
| USB-C Connector 16P | Digi-Key Corporation | 1.00 | 3 | 3.00 |
| 22µF Ceramic Capacitor 1206 | Digi-Key Corporation | 0.406 | 10 | 4.06 |
| Total + Shipping + Tariff | | | | 55.53 |

### 2.4.2 Labor

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.

# 3. Conclusion

Over the course of the semester, we successfully designed and built an automated self-watering system called LeafLink. The system was able to accurately sense soil moisture levels and activate watering based on a user-defined threshold. The companion mobile application allowed users to manually trigger watering, schedule watering events, adjust moisture thresholds, and monitor real-time soil moisture data.

## 3.1 Ethical considerations

LeafLink aligns with the Code of Ethics by emphasizing public safety and responsible engineering practices. All electrical components were assembled according to standard safety practices, with properly insulated wiring and grounded connections where applicable. Additionally, the PCB and electronic components were enclosed in a protective housing to minimize the risk of water-related electrical damage.

LeafLink also follows the IEEE principles of honesty and transparency by providing reliable sensor data to users and by not sharing user data with third parties [5]. The system contributes to public welfare and environmental sustainability by simplifying plant maintenance and helping users maintain healthy indoor plants. Furthermore, the automated watering system reduces unnecessary water consumption by minimizing overwatering.

## 3.2 Future work

Although the breadboard prototype maintained relatively stable soil moisture levels, the final design occasionally watered the plant continuously until the moisture reading reached 100%. In future iterations, a timing-based control system should be implemented so that the pump activates for a short interval, such as ten seconds, before rechecking the soil moisture level. This would help prevent overwatering and improve overall system efficiency.

Additional improvements could include integrating a water-level sensor to notify users when the reservoir requires refilling. Future versions could also incorporate environmental sensors, such as a sunlight sensor, to help users optimize plant placement based on lighting conditions. Finally, the system could be expanded to support multiple plants simultaneously, allowing a single controller to manage several watering zones.

# References


[1] A. Bringslimark, T. Hartig, and G. G. Patil, “Effects of indoor plants on human functions: A systematic review with meta-analyses,” *Journal of Environmental Psychology*, vol. 29, no. 4, pp. 422–433, Dec. 2009.

[2] “U.S. houseplant participation 2020,” *Statista*. [Online]. Available: https://www.statista.com/. [Accessed: May 6, 2026].

[3] “How many people forget to water their plants,” *Plants.com*. [Online]. Available: https://www.plants.com/. [Accessed: May 6, 2026].

[4] “How does a capacitive soil moisture sensor work? – Easy explanation,” *GardenerBible*. [Online]. Available: https://gardenerbible.com/. [Accessed: May 6, 2026].

[5] IEEE, “IEEE code of ethics,” 2023. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: May 6, 2026].

[6] OpenAI, “ChatGPT,” OpenAI, 2026. [Online]. Available: https://chatgpt.com/. [Accessed: May 6, 2026].

[7] DFRobot, “Capacitive Soil Moisture Sensor SKU:SEN0193,” datasheet. [Online]. Available: https://media.digikey.com/pdf/data%20sheets/dfrobot%20pdfs/sen0193_web.pdf. [Accessed: May 6, 2026].

[8] Espressif Systems, “ESP32-S3 Series Datasheet,” datasheet. [Online]. Available: https://documentation.espressif.com/esp32-s3_datasheet_en.pdf. [Accessed: May 6, 2026].