Autonomous Soil Health Management System

S.A.G.E

Soil Analysis & Ground Evaluation

Eithan Capella
Emmanuel Gonzalez
Bryan Vega

Table Of Content

Introduction

Soil health lies at the very heart of agricultural productivity. When soil is balanced and nutrient-rich, crops flourish, yielding higher harvests with fewer inputs. Conversely, if soil becomes depleted or contaminated—whether by industrial pollutants, heavy metals, or excessive chemical fertilizers—farmers can face steep declines in production and quality. This not only hurts profitability but can also lead to long-term damage to local ecosystems. In many agricultural communities, especially those with limited financial or technical resources, assessing soil quality often comes down to guesswork or sporadic lab testing. Unfortunately, such testing can be slow and prohibitively expensive, forcing farmers to make critical decisions about fertilization, irrigation, and pest management without accurate, real-time data.

The cost and logistics of traditional lab-based soil analysis create a barrier to effective soil health management, particularly for small and medium-scale farmers. Transporting samples to distant labs and waiting days or even weeks for results can mean the difference between salvaging a crop and suffering a near-complete loss. In the meantime, undetected contamination or nutrient imbalances may continue to worsen, further jeopardizing productivity. These challenges are even more pronounced in areas with limited infrastructure, where unreliable transportation or a lack of nearby testing facilities can exacerbate delays.

Moreover, environmental concerns are becoming increasingly urgent. Excessive fertilizer application, for example, can lead to nutrient runoff into local waterways, harming aquatic life. Misuse of pesticides or over-irrigation can also degrade soil structure over time, making it more susceptible to erosion and reducing its capacity to support healthy plant growth. Without up-to-date information, farmers may inadvertently apply chemicals that are no longer needed or fail to address issues like soil acidification until the damage is evident in reduced yields or plant health problems.

In response to these issues, new approaches that integrate Internet of Things (IoT) technologies offer promising solutions. Real-time monitoring systems can detect small changes in soil conditions as they happen, alerting farmers before issues escalate. Low-cost sensors embedded in the field can measure critical parameters such as pH, moisture, and nutrient levels (NPK), transmitting that data to a central hub for easy analysis and visualization. This shift from periodic testing to continuous monitoring has the potential to dramatically improve farming efficiency. Farmers gain the ability to act immediately if soil tests indicate contamination or nutrient deficiencies—whether that means adjusting irrigation schedules, modifying fertilizer regimens, or investigating sources of pollutants.

Yet, despite the proven benefits of IoT-based soil monitoring, many commercially available systems remain too expensive or complex for smaller farms. Proprietary software, specialized equipment, and subscription fees can all add to the cost, while maintenance and troubleshooting often require technical skills or professional support. Consequently, there is a growing need for a simplified, affordable, and user-friendly system that provides the core benefits of continuous soil monitoring without imposing a heavy financial or technical burden. By focusing on essential sensors, easy-to-manage hardware like the ESP32 microcontroller, and a straightforward web interface, our project aims to fill this gap. Ultimately,

the goal is to equip farmers—regardless of scale—to make timely, data-driven decisions that not only boost productivity but also foster more sustainable, environmentally friendly practices in agriculture.

Problem Statement

Agricultural soil conditions can change rapidly, often without visible warning signs. Contaminants like heavy metals, pesticides, or excess chemical fertilizers can seep into the soil and disrupt its nutrient balance, while factors like erosion or over-irrigation gradually degrade its structure. Detecting these issues early is critical: delays in identifying contamination or imbalances can have severe consequences, including reduced crop yields, wasted resources, and long-term harm to the environment. In many regions, however, farmers rely on traditional soil tests that involve sending samples to commercial laboratories. This process can be slow and expensive, placing real-time soil monitoring out of reach for those who lack the necessary funds or logistical support—particularly small and medium-scale farmers. By the time lab results arrive, the damage may already be done, forcing farmers to bear the costs of corrective measures or accept yield losses.

The driving force behind this project is the urgent need for continuous, reliable, and cost-effective soil monitoring. If nutrient deficiencies or soil contamination can be spotted promptly, farmers can take immediate steps—adjusting fertilizer application rates, modifying irrigation schedules, or investigating possible pollution sources—before these problems escalate. With climate change intensifying weather extremes, the margin for error in farming has grown smaller, and traditional testing methods are too slow to provide a meaningful safety net. An autonomous soil health management system offers a powerful alternative, enabling timely decision-making that can save crops, protect soil fertility, and reduce environmental impact.

Our primary focus is on small and medium-scale farmers, who often face financial and logistical hurdles when it comes to soil testing. These farmers need a straightforward, dependable system that delivers near-instant feedback on soil conditions. While large commercial farms may afford complex proprietary solutions, smaller operations typically cannot justify high upfront costs or recurring subscription fees. By lowering barriers to entry, our project aims to democratize soil health insights and enhance food security in areas that might otherwise be left behind by expensive, specialized technologies.

Our project concentrates on the precision agriculture domain, specifically Smart Farming through IoT edge devices. We focus on small-scale farming operations, individual gardeners, shared agricultural spaces, possibly extending to Educational Institutions or Agricultural Researchers due to the ability to have specialized sensors. In general, farming is an expensive venture, and requires frequent monitoring if you wish to receive the most possible yield. For this reason, small farms often want affordable, on-site methods to monitor soil conditions without having to rely on lab tests. Gardeners and Community Gardens benefit from simple yet cost-effective practical tools to maintain soil health. Through the use of a user-friendly Mobile application and an alternate Web Application with logged results, we aim to help the users determine and understand their soil's composition and behavior.

To address these challenges, our project aims to develop a real-time, IoT-driven soil health management system that continuously measures critical parameters such as pH, moisture, and nutrient

levels (NPK). We will integrate affordable sensors with an ESP32 microcontroller, which can transmit data wirelessly to a web-based dashboard or app. This setup requires minimal installation overhead and leverages existing Wi-Fi or local hotspot networks, reducing the need for additional infrastructure. Unlike traditional, lab-based testing processes—or high-end proprietary solutions—this innovation provides an immediate, accurate snapshot of soil conditions, enabling farmers to respond to threats like contamination or nutrient depletion before they harm yields.

By prioritizing affordability and ease of use, this project seeks to bridge the gap between precision agriculture and real-world constraints, offering an accessible tool that can significantly improve on-farm decision-making. In doing so, we not only help individual farmers protect their livelihoods but also contribute to broader sustainability goals. Healthier soils mean stronger ecosystems, more efficient use of resources, and a reduced environmental footprint—benefiting communities and future generations alike.

Project Objectives

Project General Goal

- Develop a IoT Based Solution using the ESP32, that is capable of **logging** and **monitoring** different soil parameters such as moisture, temperature, pH, nutrient levels, and so on. With the aim of assisting farmers and gardeners into making more data-driven decisions into their crop management, yielding to better crop health and yield. The device will not replace lab results but rather try to realistically approximate their results.

Objective 1: Develop Sensor Module

- **Specific:** The sensor layout and wiring will be decided, ensuring each sensor (pH, moisture, humidity, temperature, NPK, etc.) is connected to the appropriate ESP32 pins (analog/digital). Calibration, scaling and filtering routines will be implemented to ensure accurate data.
- **Measurable:** Each reading of the sensors needs to be compared against a known reference, temperatures validated with a thermometer, moisture and humidity with a hygrometer and capacitance probes.
- **Achievable:** Create a Wiring Diagram of the sensors and the used ESP32 pinout. Connect and preprocess the Sensor's readings for calibration. The collected data must be formatted to CSV
- **Relevant:** Ties in with the general goal of developing an IoT based solution for smart agriculture. Having accurate sensor readings and calibrations are the foundational steps to ensure that clean, reliable field data is collected.
- **Time Based:**
    - Week 1: Finalize wiring Diagram and confirm preliminary sensor readings.
    - Week 2: Implement calibrations and scaling routines to rectify sensor accuracy
    - Week 3: Format the collected data for CSV export and test data logging.

Objective 2: Set-up Wi-Fi AP

- **Specific**: Set-up the ESP32 as a Wi-Fi Access Point to allow devices (desktop, ios, android) to transfer data from the ESP32 to a device.

- **Measurable**: The ESP32 (microcontroller)'s network can be accessed by any device with a network through the ESP32's network, utilizing its private IPv4 Address 192.168.4.1 (default).
- **Achievable**: Configuring the ESP32 Wi-Fi to set it up as an access point, verifying that the device can access the microcontroller.
- **Relevant**: Enabling access points is important to ensure connectivity in remote places like farms without conventional signals, by establishing its own network.
- **Time Based**: The configuration will be complete and ready to be accessed in the first week of receiving the ESP32.

Objective 3: Web application - Internal Tool

- **Specific**: Create an internal web based platform that logs and displays collected soil data. Communicates through the Wi-Fi AP.
- **Measurable**: The platform will be able to control the device (start & end logging) and display the resulting data entries (soil nutrients, moisture, etc) from the sensor.
- **Achievable**: Create device routines to generate the sensor logs 2 minimum start and end sensor logging. Allow the user to display log values on screen with a block for each sensor value.
- **Relevant**: The website will serve as an internal interface between the ESP32 and the user without the use of an external app.
- **Time Based**: The web application will be completed and accessed through ESP32's network at the second week of receiving the ESP32.

Objective 4: Mobile Application - Field Dashboard

- **Specific:** Create a mobile application in which the ESP32 will offload its tasks to. It will serve as a dashboard for the ESP32 devices, it will visualize the sensor logs, providing a historical view of the collected data.
- **Measurable:** The Dashboard will support:
    - Adding and managing multiple ESP32s
    - Visualization of sensor readings across time.
    - Basic File Management for the CSV logs.
    - Local SQLite Database to store all sessions.
    - (Possibly) Export graphs and logs.
- **Achievable:** The app must be developed with a cross-platform framework to ensure compability with a wide variety of devices (React Native). Libraries for graphing must be used to provide at least the basic visualizations of sensor reading vs time. An SQLite database will be used to store and handle relevant queries regarding data. The ESP32 will offload its data over the Wi-Fi access point.
- **Relevant:** Given the ESP32 does not have enough resources to host everything. This mobile app will function as the main core application for long-term data visualization and controls. It allows users to access soil data from multiple ESP32s allowing the comparison of different sessions and field trends over time.
- **Time-Based:** The expected timeline for a functional skeleton of the app with the core features of device connection, data download and visualization is around 4-5 weeks. It is likely to extend if we were to face challenges in the transfer of data.

Objective 5: Identify Real Soil Lab Test

- **Specific**: Identify a reliable soil lab test that accurately measures essential soil parameters such as nutrient levels, moisture, and pH.
- **Measurable**: Success will be measured by selecting a test that meets established quality criteria including precision, consistency, and documented accuracy.
- **Achievable**: Consulting with local farms and academic literature to assess available testing options.
- **Relevant**: The identification of real lab results is crucial to validate and calibrate the project's soil analysis model with real data.
- **Time Based**: The selection and verification process for the soil lab test will be completed within three weeks from the project's start date.

Objective 6: Testing and Accuracy

- **Specific**: Validate sensor readings against standard lab soil tests.
- **Measurable**: Conduct 10+ tests with known results such as soil with fertilizer, fruits, compost and lab results.
- **Achievable**: Calibration algorithms will be refined to minimize discrepancies.
- **Relevant**: Data accuracy is crucial for farmers and gardeners to trust and adopt the device.
- **Time Based**: Testing and calibration should be finalized by the last month of the semester.

Objective 7: Research Spectroscopy

- **Specific**: Research the feasibility of using spectroscopy or photoresistor sensors to detect any additional soil nutrient using their wavelength for which dedicated sensors are not available.
- **Measurable**: Add results obtained using spectroscopy detection to the core sensor result to increase accuracy against verified lab test results.
- **Achievable**: Review research papers and conduct experiment to evaluate if possible and the quality of the data,
- **Relevant**: Increase the soil analysis capabilities by detecting other nutrients that are harder to detect, because there are no dedicated sensors (hardware).
- **Time Based**: The initial research and experiment should be done by the end of March, for finding and to have enough time for integration with the core sensors.

Solution Approach

We propose a low-cost, real-time soil monitoring system that integrates IoT devices with a local web based dashboard. The system will use an ESP32 as the main microcontroller for wireless communication and data processing. Farmers will be able to use the web-app by connecting to the microcontroller's Wi-Fi access point accessing the hosted local web server. Measuring soil health parameters such as moisture, temperature, pH, and nutrient levels that includes nitrogen, phosphate, potassium. Log sensor data to an SD card that will enable historical analysis and offline data storage, the dashboard will be able to view this historical data. Since we are using an ESP32 it ensures low power consumption, allowing for long operation time between charges.

The current solutions offered in the market range tackle the problem from different perspectives and do not fall into the same category. Some focus on larger scale operations where the budget is larger. Whilst some focus on smaller scale operations, differing in the type of management the client seeks. The following are a few of the current solutions in the market:

- **Libelium** is the closest contender, offering an IoT solution for crop monitoring. Their devices might differ in sensors but they aim to measure different weather conditions, light levels, soil morphology, fertilizers and so on. An advantage of their solution is the use of wireless communication at large distances. This device whilst a contender is not something viable for small scale farms as some of these devices can cost up to $10,000, with some possible added licensing fees. It is a good product, but its price is not accessible for struggling farmers or smaller scale operations that cannot afford many of these devices.
- **OneSoil** aims to help manage agriculture based on satellite imaging, and providing weather data of the region in a self contained suite, though it is a superficial level of management, as it does not focus on soil, but rather on weather and crop monitoring through remote sensing techniques.
- **LandPKS** LandManagement Module is a more manual product aimed to track small farms/garden activity which aims to help the farmer to learn how to measure the different parameters to diagnose their soil/crop health.
- The **ComWinTop** USB portable soil tester can measure the basic information of moisture, NPK (Nitrogen, Potassium and Phosphate), this device includes a free Android app, with simple use, but can only be obtained using a qr code and downloading an APK from an unknown source.

As of now, in the market, there are few IoT solutions that are affordable for smaller scale operations like farmers in Puerto Rico. This present gap is what our solution aims to fill, by providing an affordable comprehensive on-site IoT-based Soil Monitoring, real-time data logging and ease of use all packaged within an individual device. Which allows us to bring forth a valuable tool for farmers. The benefits of having a system like this is that it reduces the time that farmers need to wait for a lab test to know the fertility of their farm's soil. It could also potentially reduce the amount of tests done long-term, reducing costs due to making the farmer more independent from lab tests. Collecting the information is also easy, simply requiring a connection to the ESP32 WiFi Access Point, to access the web-app in order to enable the sensors. Logging the data is also a quick process, allowing the user to view these files later on, seeing the soil composition changes across time. It also does not depend on location as it is all hosted locally in the ESP32's network. A big limitation of on-field testing is accuracy, the measurements will reduce the dependency of lab tests, but they do not replace laboratory tests which provide absolute accuracy to soil composition, but when farmers seek fast feedback during seasonal changes, or soil amendment or conditioning, it helps to give them that feedback they need.

The solution possesses a good commercial potential, Small-scale farmers or gardeners might depend on more manual tracking through the use of multiple sensors to take their measurements. Given the difference of previously mentioned expensive solutions or simplistic devices there is a market gap that can be filled. There is also an overall trend leaning towards Precision Agriculture due to rising needs in improving yields, more sustainable farming initiatives and so on. The solution can also have tiers as in a base model with the core sensors and other versions of the device with more specialized sensors and

features for the client. If bought in bulk the price for manufacturing it could be greatly reduced and be used to reduce price further creating less of a barrier of entry, which is appealing for all groups. Or another route could be to try Open-Source Kits, selling to hobbyists and agronomists looking to do their own projects and customizations be it for research or farming.

As mentioned, in order to remain competitive with the other products, we need to supply the basic sensors to achieve an accurate and comprehensive soil analysis.

- Core Sensors:
  - Soil Moisture Sensor (Capacitive preferred)
  - Soil pH Sensor (Potential Substitute Liquid pH Sensor)
  - Temperature Sensor
  - NPK Sensor

These are a set of potential sensors that could be considered:

- Additional Sensors (Stretch Goals)
  - Optical Transducers / Multiband Hyperspectral Camera (Detection of Nutrients)
  - Illuminance or Ambient Light Sensor
  - Water Level sensor (Rain measurement)
  - Ambient Humidity

List 1: Sensor Lists divided by Phase.

Many labs perform Spectroscopy on soil through the use of reagents which help to classify the concentrations of nutrients or heavy metals inside the soil. We believe that through this logic we could perform a similar task with simpler tools albeit at a reduced accuracy. Though it would still be useful given it could be used as an early warning or superficial overview of the soil composition.

This technique could generate a substantial intellectual property regarding the process pipeline of using wavelengths to detect the presence of nutrients, minerals or metals. Including any sensory arrangements of the possible electronics, such as Photodiodes, Photoresistors, Spectral Sensors or Hyperspectral Cameras. Up to algorithms used to filter or process the data obtained from the Absorption and Reflectance Signatures.

Outside this technique the way data from the sensors is processed and visualized into the dashboard could result in intellectual property in regards to the copyright of design, artwork, original code and so on. Any concerns regarding intellectual property would be managed in consideration of UPR's IP policies. Ensuring that all IP resulting from the capstone project is documented properly. If the techniques proved to be viable and commercializable, we will explore multiple options regarding the filing of patents or other Intellectual Property protections for such techniques.

Nutrient	Absorption Wavelength (nm)
Nitrogen	902, 1054, 1221, 1478, 1697, 1969, 2104 nm for different soil types
Phosphorus	213-215 nm (UV)

Potassium	766-770 nm (NIR) (reflectance since poor absorption in UV)
Calcium	422.7 nm
Magnesium	285.2 nm
Sulfur	180-250 nm (UV)
Heavy Metals	Absorption Wavelength (nm)
Lead	200-220 nm (UV)
Mercury	185-250 nm (UV)
Cadmium	228-240 nm (UV)
Arsenic	193-197 nm (UV)

Table 1: Wavelengths of Elements

Note: These Wavelengths vary as some studies note wavelength absorption peaks, or use other forms of the mineral. Further Research must be done for the Near Infrared Spectrum as they are the most researched regarding soil.

Technical Description

Our proposed IoT-driven soil monitoring system focuses on accessibility and affordability, distinguishing it from existing commercial and open-source solutions. While some high-end devices like Libelium Smart Agriculture provide robust wireless monitoring for large farms, they can cost upwards of $10,000 and impose recurring licensing fees. By contrast, our project uses low-cost sensors (e.g., capacitive moisture probes, pH) alongside an ESP32 microcontroller, aiming to serve smaller-scale farms that cannot justify the expense and complexity of proprietary tools.

Another area of differentiation lies in real-time, on-site soil measurement. Services such as OneSoil or Traction Field App rely heavily on satellite imagery or require sending soil samples to external labs for analysis, which delays feedback to farmers. Our system measures parameters like moisture, pH, and NPK directly in the field and streams that information via a built-in Wi-Fi access point, allowing farmers to react immediately. This approach not only reduces lab costs but also ensures that data collection continues uninterrupted, even in locations with limited internet connectivity.

The system's architecture will start by prompting the user to input a request for the sensor data. Once the ESP32 and the sensor are positioned in a correct manner, the user may start the main routine from the web application to extract the data from ESP32's sensors, creating a log file on which the user can visualize the data of the soil nutrients. The log file will be stored in a SD card, which allows the user to access it later to view their field's historical data.

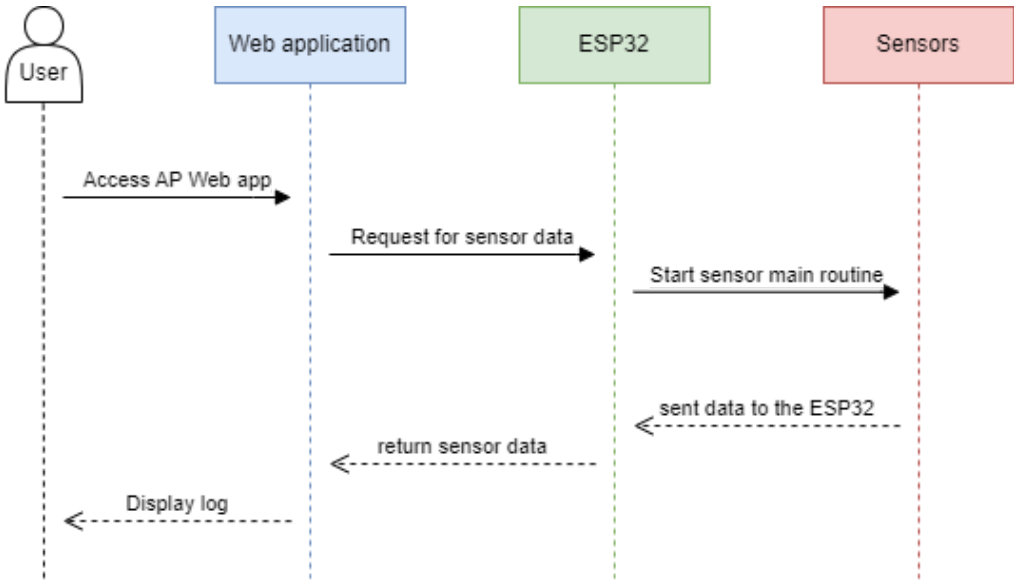


Figure 1: Sequence Diagram of System Architecture

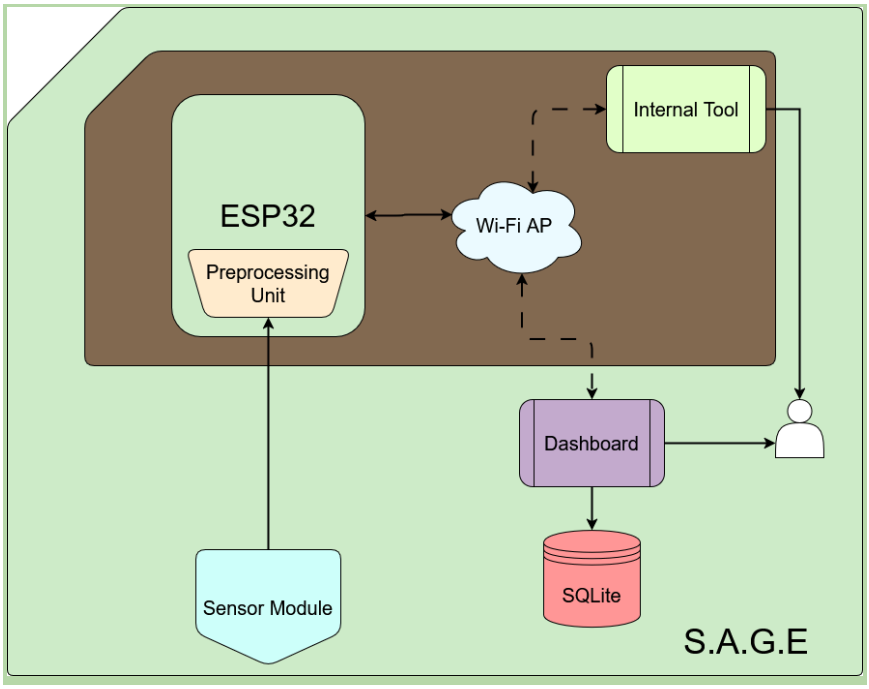


Figure 2: Updated General Architecture Diagram

- **Components**:
  - **Sensor Module**
    - **Sensor Hardware**: Moisture, Humidity, Soil & Ambient Temperature, pH and NPK (electrical conductivity). Connected to the ESP32
  - **ESP32**
    - Core device, responsible for data collection, preprocessing and Wi-Fi AP Hosting and serving both the **Internal Tool** & **Dashboard App**.
    - Signal Processing and communication with User Interfaces.
  - **Preprocessing Unit**
    - Prepares raw sensor data for use by calibrating analog signals using map_value() and constrain_value() with a known reference.
    - Average pooling is applied on collected data over time intervals to reduce noisy data and outliers.
  - **Wi-Fi Access Point**
    - The ESP32 creates a local Wi-Fi network which allows users to connect to the device. The access point supports both User Interfaces.
  - **User Interface**
    - **Internal Tool** - (Debugging & Live Testing) A locally hosted web interface. Provides core controls: Start & End Logging, and Log Viewer.
    - **Dashboard App:** Comprehensive Management app. It supports multiple device management for multiple ESP32s, Visual Analytics: Trends & Stats, a local database for long-term tracking and querying.
  - **SQLite Database**
    - Used in the Dashboard app to manage and store the collected data from multiple S.A.G.E devices. It facilitates complex queries and trend analysis to form improved insights into the soil.
- **External Components**
  - **Storage Module**
    - **SD Card**: Stores logs for historical analysis. Depending on resource usage an SD card could be installed within the ESP32 to hold more information relevant to the sensors.
  - **Power Management Module**
    - **Battery**: According to estimations the ESP32 consumes at most around 150-240mA when it is active (Processing and utilizing Wifi). The different sensors could take around 70mA at most. Assuming these worst case scenarios:

      A power bank that provides 10,000mAh at 5V would need to be stepped down to operate at 3.3V (ESP32 range).
      - Assuming an 85% efficiency we would have at least 8,500mAh usable.
      - Runtime would be calculated as in a high consumption scenario:
        - $Runtime\ (in\ hours)\ = Battery\ mAh/Total\ Consumption\ mA$
        - $Runtime\ (in\ hours)\ = 8,500mAh/310mA\ =\ 27.41hrs$

Database Schema

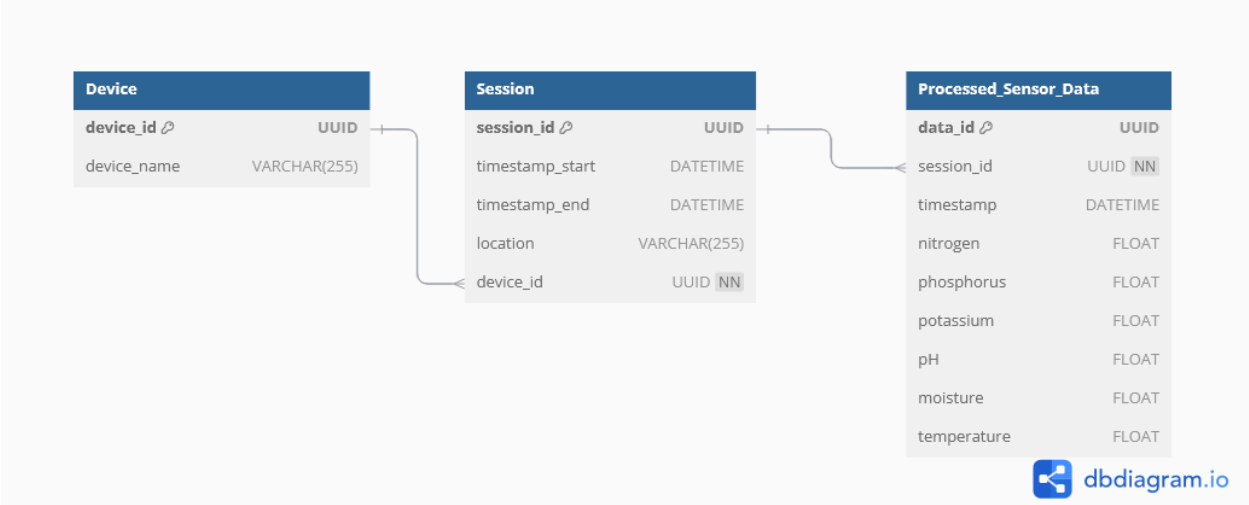


Table 2: Database Schema

The Schema for this device is simple, as it is all hosted locally there is no need for authentication. It could be abstracted even further if we were to assume the Client only had one device. This schema entails that one device can hold many sessions, the idea being that each session captures the sensor reading over a set of time. [Timestamp start and end]. We then use the session_id to query the Processed_Sensor_data and find all relevant data according to that session.

After the Access Point (AP) is configured, a web application will be created and hosted in a WebServer in the ESP32. This web application will serve as an user interface, where they will be able to start and get the data of the sensors, once the sensors have sent all the data the website is going to redirect to the new log's dashboard page, see **Figure 3** and **4** for both views in IOS and Android and **Figure 5** to see the flowchart. The logs are going to be stored in a SD card and the users will be able to view the previous logs and a user-friendly arrangement. The web application will be developed using vanilla JavaScript, HTML and CSS to make sure the website is responsive and with a lightweight bundle size, so it is easier to process in the ESP32.

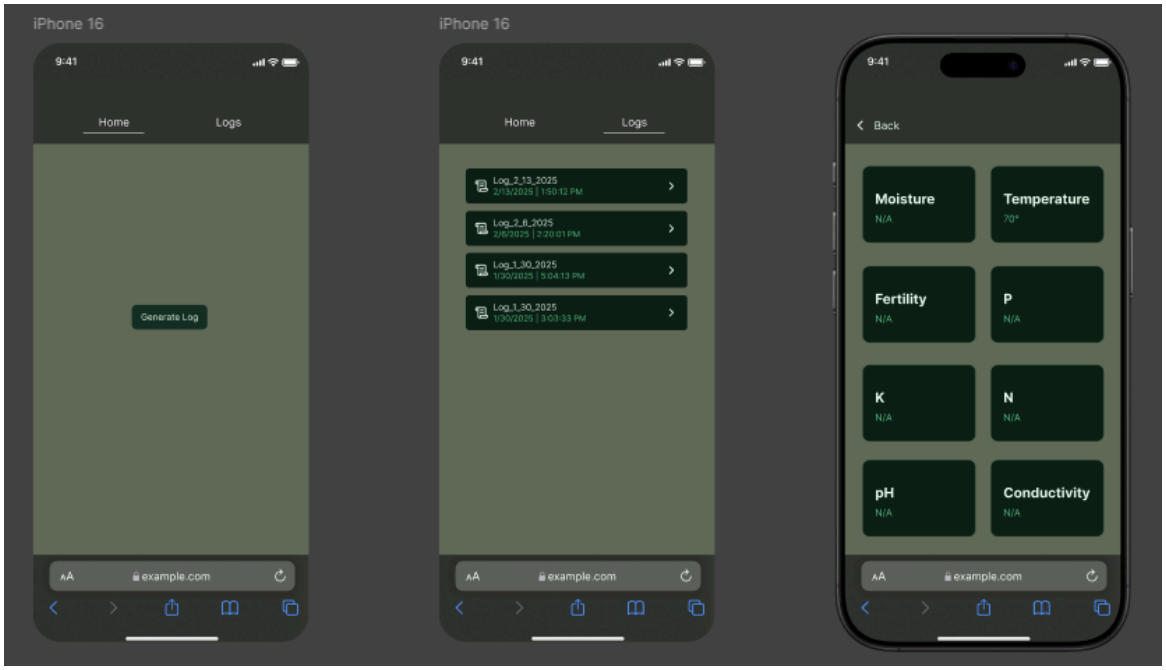


Figure 3: Wireframes for Web application for IOS

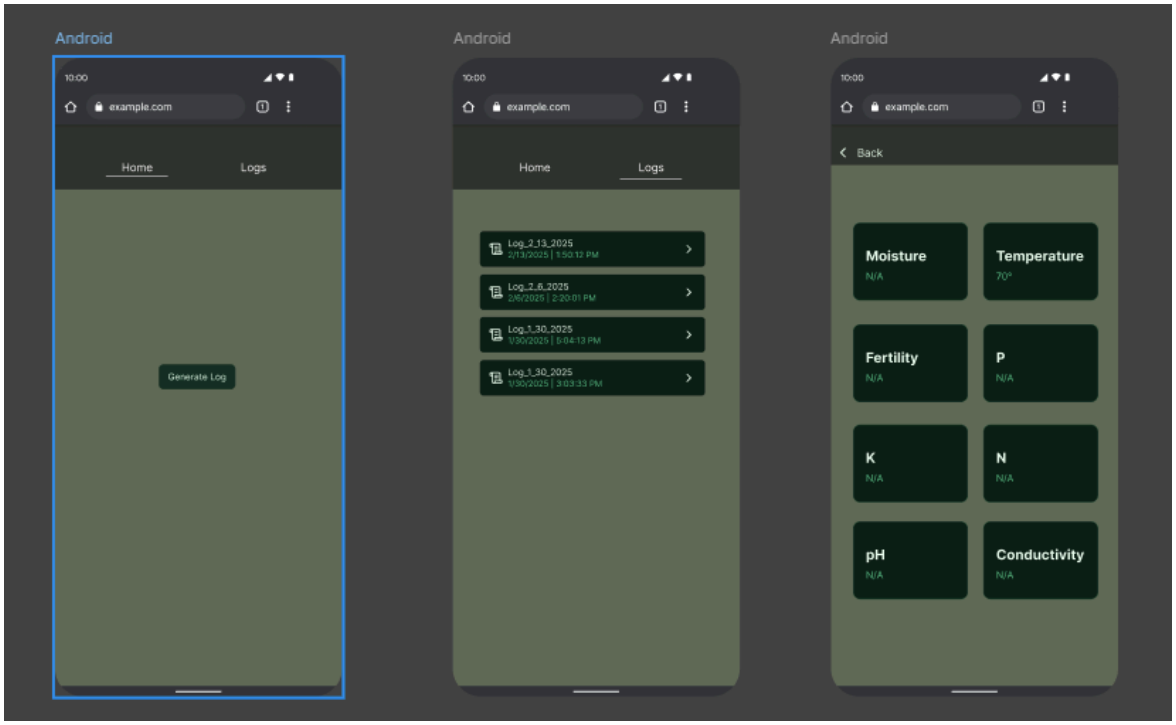


Figure 4: Wireframes for Web application for Android

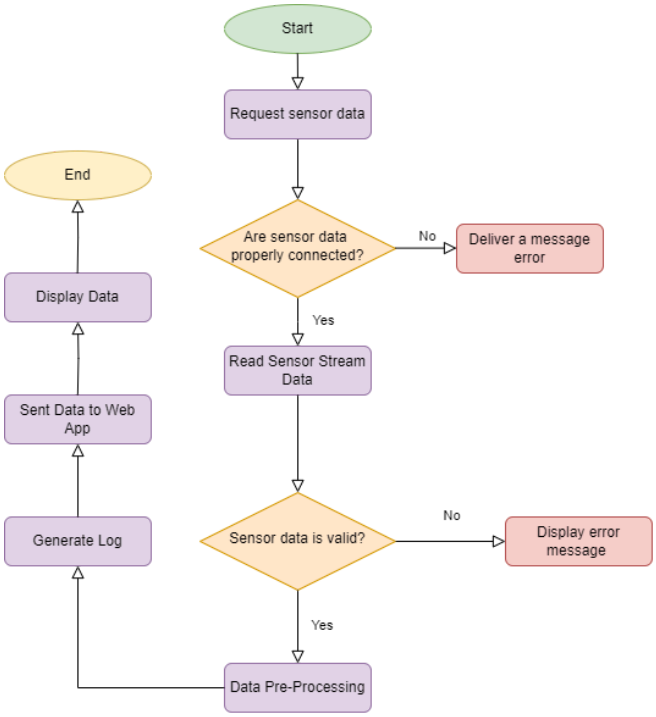


Figure 5: Flowchart for program behaviour

There are multiple constraints in the development of this device. Some of these fall into the environment this device will be subjected to. At a minimum this device needs to be dust and water resistant as it might rain during its activity. It needs to be able to withstand and perform accurately under the heat of the sun as well. The interconnection of hardware can be another constraint since the ESP32 has limited ports, it is a possibility that for the Stretch Goals a microcontroller could be added to have more ports in order to allocate more sensors. The biggest foreseeable constraint is the research and development of the spectroscopy aspects of this project. Expansive information of this process on soil is not readily available, and as such there is a need to investigate and compare known soil samples (lab results) against the sensors in order to determine the presence of the added nutrients or metals.

There are multiple engineering standards that apply for this project:

- **IEEE 802.11** (Wi-Fi): It entails the standard for Wireless Communication that the ESP32 needs to follow.
- **IEC 60529** Ingress Protection Ratings: It entails the standard and grading of the resistance of enclosures of electronic devices against dust and liquids.
- **ISO 10381**: If we considered sending soil samples for comparison this is the standard for soil sampling methods.
- **CSV**: Would be our standardized data format across devices.
- **ISO 11074**: Covers the list of terms used in the field of soil quality.
- **ISO 28258**: Describes how to exchange soil-related data across different digital systems (data compatibility)

Project Plan

The project can be divided into three main phases Preparation and Setup, Processing and Storing, lastly Web App and Visualization:

Phase 1

Once the ESP32 and the core sensors are received, set up can begin. This consists of connecting each Sensor to the appropriate port. The sensors have two different ways of transmitting data, ADC (Analog to Digital Converter), and RS-485, an industrial standard for long distances. In our case we might need conversions to make it compatible for the ESP32. Once all sensors are set up, we need to ensure all the data read by the sensor is reliable and consistent. Which entails calibration of each sensor against known values and conversion tables since some are based on conductivity. For example pH sensors can be tested by checking with a known acid/base, moisture check with water, temperature, and so on.

Phase 2

Once the sensor readings are known to be usable, the data processing phase begins. A good safety measure is to pad out the values read by the sensors, either by averaging them out or setting up a possible weight to create some inertia and smooth out the values in case any readings suddenly spike. After this the data can be logged, as of now the data will most likely be stored in the CSV format. As it is now ready for visualization.

Phase 3

Once the data is stored in our selected format. We begin making the Web App, creating the dashboard for every sensor reading, based on the CSV file. A section for selecting the logs. Once this selection is done, we can look into graphing the values in different ways be it bar charts or line graphs if we decide to measure across multiple logs giving insight over the changes over time of the soil.

Stretch Goals

Most of the stretch goals focus on the implementation of extra sensors, though a particular interest is using spectroscopy techniques to measure the presence of nutrients or heavy metals through their absorption wavelengths.

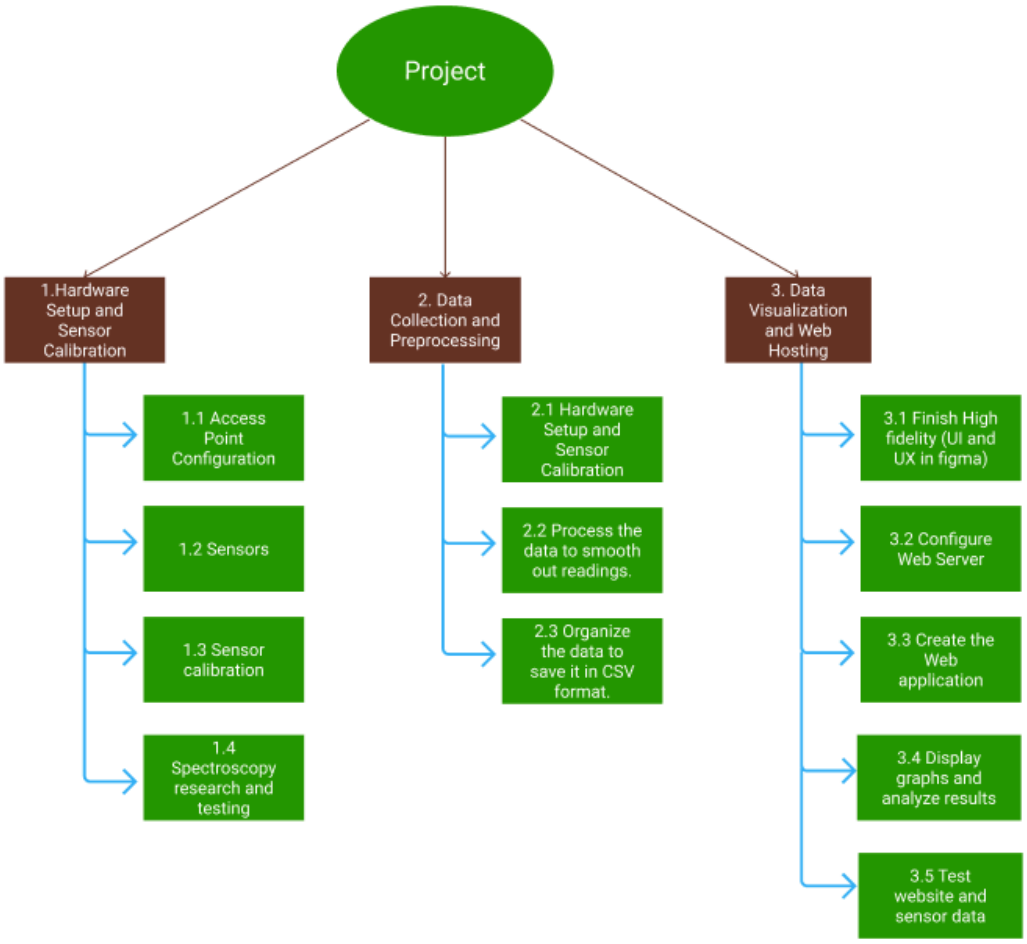


Figure 6: WBS Diagram of Milestones and Deliverables

The diagram consists of mainly 2 levels that compose the project. Level 2 Consists of the Milestones, the Phases that must be completed and entail a completed set of modules of the project. The third level describes each of the tasks that must be completed in order to deliver on a Milestone. The expected timeline depends on the date of arrival of all the electronic components which is as follows:

Important dates:
Apr 14, 2025 Progress report
May 12, 2025 Final Report

Phase 1 February to March
Feb 21, 2025 : ESP32s and initial Sensors Arrive
Feb 21, 2025 to Mar 7, 2025 : **1.1**, **1.2**, **1.3**, **1.4**

Phase 2 March to April
Mar 7, 2025 to Mar 21, 2025 **2.1**, **2.2**
Mar 21, 2025 to Apr 14, 2025 **2.2**, **2.3**

Phase 3 April to May
Apr 14, 2025 to Apr 21, 2025 **3.1**, **3.2**, **3.3**
Apr 28, 2025 to May 5, 2025 **3.4**, **3.5**
May 5, 2025 to May 12, 2025 **Stretch Goals**

Specific Tasks:

Emmanuel J. Gonzalez Morales: Will focus working with the front-end, improving the UI/UX of the application and hosting the Web Server in the ESP32. I'm also going to work with the Spectroscopy and photoresistor sensors, to find a way to improve data quantity and quality.

Bryan D. Vega González: Will work with the backend, utilizing the presented schema. Processing the data (smoothing the values) for the database. Handling the connection between the ESP32 and the sensors focusing on the multiple sensors.

Eithan M. Capella Muñiz: Will focus on the connection between the ESP32 and the sensors, using the conversion tables for the readouts of each sensor. Also investigating how the data is read through ADC and RS485 methods. Also will be investigating the process of Spectroscopy through the multiple sensors we are considering.

Progress Report

Completed Objective 1: Sensor Module

The development of the Sensor Module involved significant challenges. Starting With a resource analysis, verifying the available ports for the ESP32 (digital & analog). Assessing whether an expansion shield was needed, and estimating the power consumption.

Programming was done using C, through the ESP-IDF Framework which introduced difficulties due to sparse documentation; some of the sensors lacked widely used libraries. Meaning only the code itself was the reference. A major hurdle faced during this was a sensor failing due to a mismatch between the digital signature of the sensor and the manufacturers' default. Which required in-depth debugging.

To address some of these issues, we tried to use Arduino as an ESP-IDF component. This meant the ESP-IDF version needed to be downgraded, this was foreshadowing future issues. The Arduino component had poor compatibility with native Arduino libraries and hardware bugs, like incorrect clock speeds which needed to be fixed manually by changing the component. After fixing it, we opted instead to develop our own functions to remove the need for the Arduino component.

After sensor functionality was established and the data was preprocessed, we focused on building the logging framework. This included creating a custom partition stable using the SPIFFS filesystem,

allocating 768kb for csv_logs. We estimate that for a format of ~5 characters per value for 7 features, it could allow up to 17,000 rows.

Finally the last hurdle was establishing modularity and interface communication. This required a total rebuild of the initial code. Since the system needed to support both user interfaces while running simultaneously with the server and sensors. To achieve this, we used FreeRTOS for concurrency via context switching, and implemented FreeRTOS queues which gave added flexibility since it allowed intertask communications, messages between tasks and between interrupts and tasks. Which laid the groundwork for modularity and flexibility for future expansion.

Completed Objective 2: Set-up Wi-Fi AP

Created and configured the ESP32 as a Wi-Fi Access Point that broadcasts the network "SAGE-ESP32-AP," enabling nearby devices like phones or laptops to connect without requiring an external router. By initializing NVS (non-volatile storage), setting up event handlers, and calling wifi_init_softap(), the code starts the AP and registers callbacks to monitor when devices join or leave the network. Once powered on, the ESP32 logs its access point's IP address, which can then be used in a web browser (or any HTTP client) to interact with the device, this give the availability, so the user can enter in the internal tool or use the mobile application to record the logs.

Completed Objective 3: Web Application - Internal Tool

After the Sensor Module & Wi-FI AP were configured, a web app was created which would interface with all the components. As of now it allows the user to control the sensor module through interrupts allowing them to start & end logging events. It also has the initial portion where the users can view all stored logs, which will consequently allow them to view the results. The communication between the interface and sensor module was achieved through handlers in web_server.c, which would be fetched by the buttons through uri_handlers.

Completed Objective 4: Mobile Application

The Mobile application is being built in the React Native framework for cross-platform in IOS and Android. We are using the Tamagui library for better styling and customizable UI components. In **Figure 7**, you can see a collection of images of the Figma project, the splash screen, and home menu are already designed. The Home Screen contains the tabs Phone and Device, the Phone tab displays all the logs that are installed in the mobile device, the Device tab has all the logs that are located in the ESP32's SPIFFS Storage. The Device tab will be disabled when no device is connected, the device is going to be connected through the WiFi Access Point of the ESP32. The circle button in the navigation bar is going to be used to start the device "record", once the routine start is not going to stop until the device is not stopped through the mobile app or the Internal tool. This can bring memory utilization concerns if the device is never stopped, but since the amount of data is taken over a span of time (we do not need to log every second, rather we could log every 15-30+ minutes), it would be difficult to reach that problem. For example if we were to log every 30 minutes that would be 48 logs per day, which at 17000 estimated possible logs for our device, means we can log for roughly 354 days without running to memory issues.

The documentation of the application is going to be simple and direct to aid farmers connect their phone with the devices. **Figure 8**, shows the toolbar that is going to be used to edit and remove logs, this

toolbar will replace the navbar and will lock the feature of editing if the user selects more than one log. Furthermore, in **Figure 9**, we can see how the user is going to see the data if they open a log, and in **Figure 10** The Farmer is also going to be able to sort the logs by date.

We originally planned to use an on-device SQLite database within the ESP32's SPIFFS (SPI Flash File System) to store and manage sensor logs locally. However, we encountered significant hurdles related to partition sizes, memory limits, and installation. After spending considerable time reconfiguring partitions and troubleshooting SPIFFS write operations, we determined that embedding a full database engine in the microcontroller environment was impractical under our constraints. As a result, we decided to build the database inside the mobile application instead, relying on Prisma to model and create the necessary tables. Prisma's schema-based approach lets us define relationships among devices, sessions, and sensor readings in a straightforward and maintainable way. It also enforces type safety, which helps catch bugs early and streamline data handling across the entire stack. Because Prisma supports a variety of database back ends (including SQLite), we can run a self-contained database on the mobile device for local, offline-capable data storage. We implemented REST-style endpoints to handle data flow, making it straightforward for the mobile app to retrieve sensor readings and store them. Using a RESTful API brings significant advantages to our soil monitoring project because it neatly separates the microcontroller's data-acquisition role from the more resource-intensive tasks of storage and analysis. Rather than requiring the ESP32 to handle complex database logic, we can expose clear HTTP endpoints—like "GET /sensor" or "POST /session"—that the mobile app or any external client can call. This modular design not only simplifies development but also offers much greater flexibility. If we decide to update how sensor data is stored, add data visualization features, or even integrate with other apps, we can do so by extending the REST endpoints without modifying the firmware.

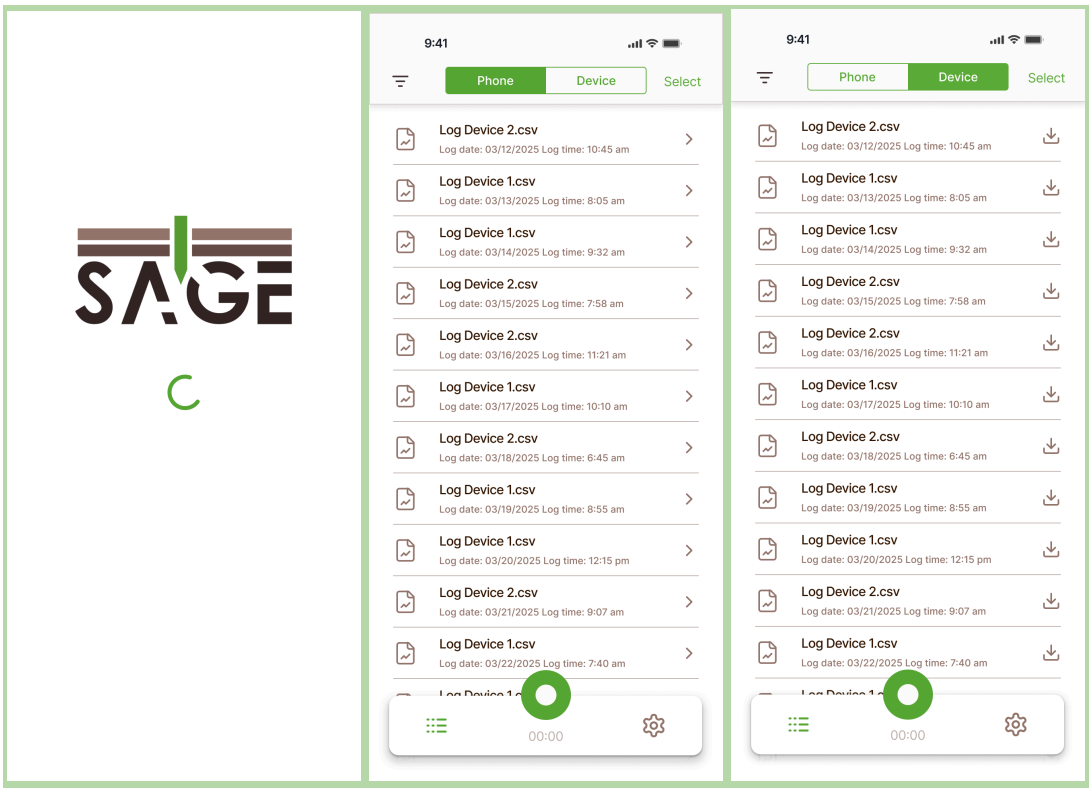


Figure 7: Mobile Application Figma

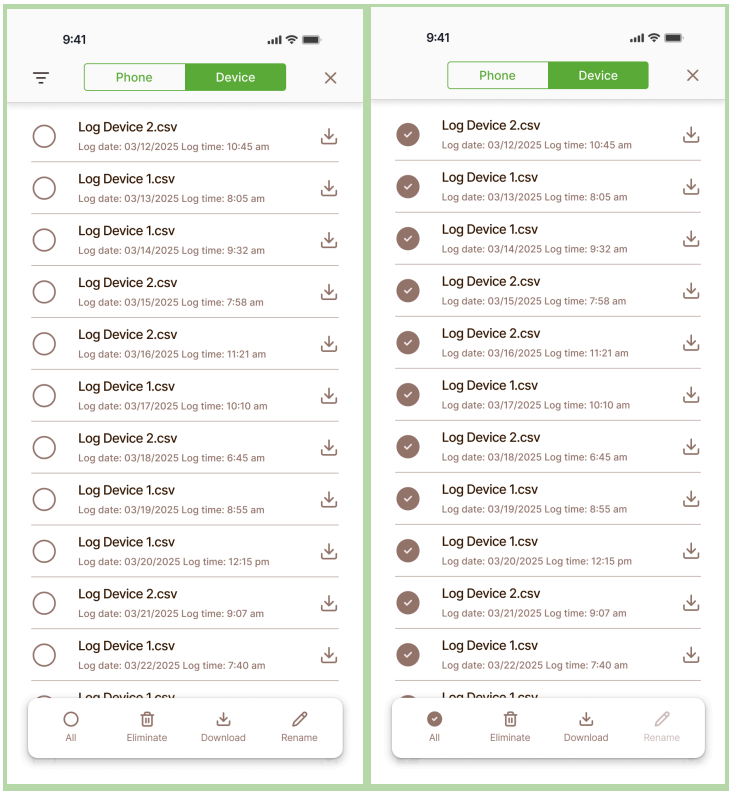


Figure 8: Mobile Application Toolbar

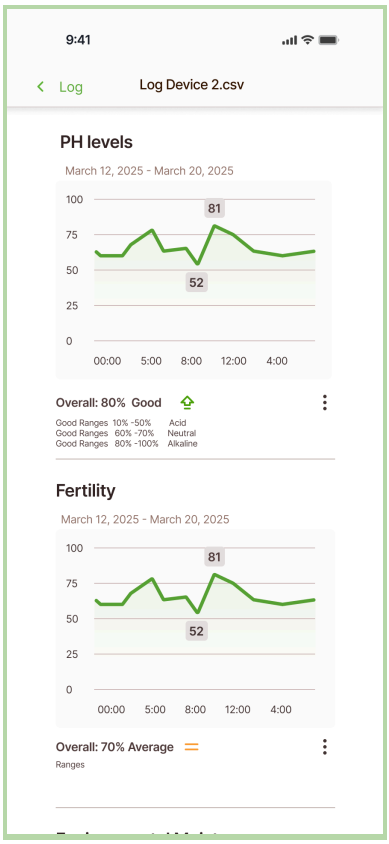


Figure 9: Mobile Application Log Analysis

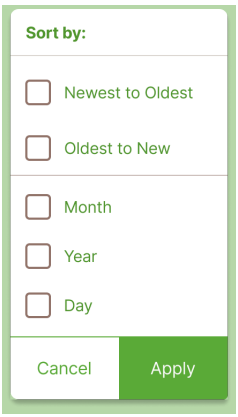


Figure 10: Mobile Application Sort by modal

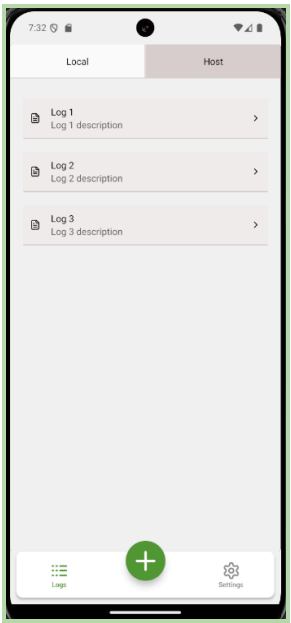
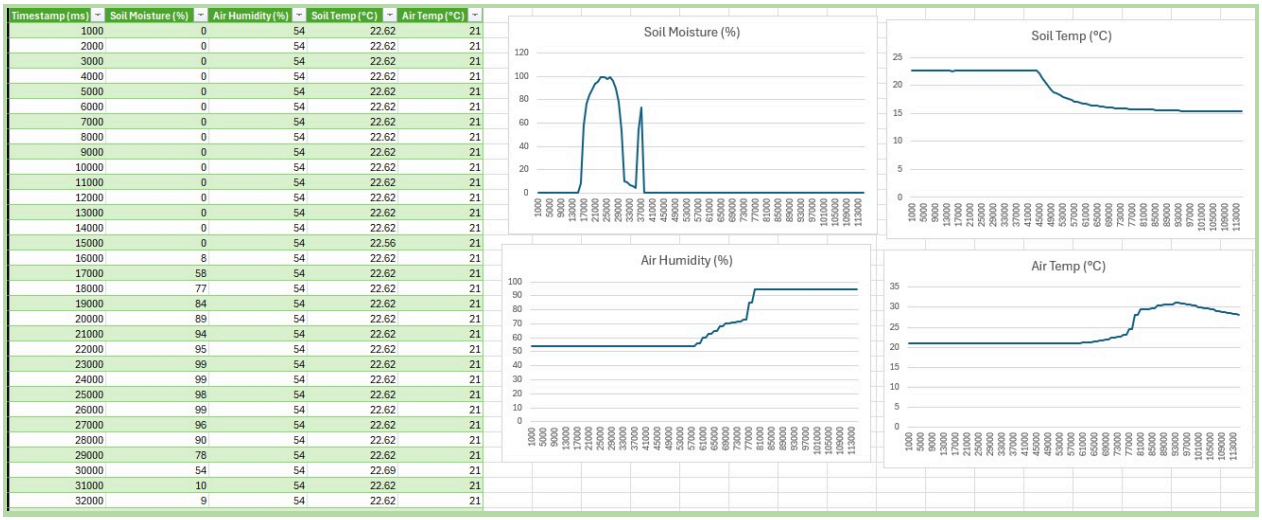


Figure 11: Current Progress in the Mobile Application

Discussion of Preliminary Performance Results

Overall our sensors have proven to be accurate, the temperature probes for ambient and soil temperature were compared with the values of digital and analog thermometers. The soil probe could be some degrees off due to its metal housing, but the values were closely related. At most the values could differ up to ±2.0°C, but on average they would differ around ±1.0°C. The ambient humidity also proved to be accurate, when compared to a hygrometer. Most of the digital sensors have been proven to be reliable by their manufacturer and wider sources. The only ones that require calibration are the analog sensors, like the soil moisture sensor which was calibrated using known samples, one at half saturation and at full saturation to establish the limits. The sensors are also proved to be sensitive to rapid fluctuations and relatively noiseless. So the initial concerns of noisy data were quelled, refer to **Figure 12**.

The NPK and PH sensors are the only ones where we have concerns. As of now the sensor was received recently so extensive testing hasn't been done yet, but from our understanding the NPK could have some reliability concerns as they measure the total amount of nutrients in soil, and use known trends to estimate the amount of each nutrient, which means there could be cases where outliers could occur. A potential way to mitigate this is to establish a baseline once soil conditions are stable, and then compare it across time. If overall nutrient availability is reduced as an average across time. Then the farmer should replenish the nutrients through the appropriate fertilizer.



Timestamp (ms)	Soil Moisture (%)	Air Humidity (%)	Soil Temp (°C)	Air Temp (°C)
1000	0	54	22.62	21
2000	0	54	22.62	21
3000	0	54	22.62	21
4000	0	54	22.62	21
5000	0	54	22.62	21
6000	0	54	22.62	21
7000	0	54	22.62	21
8000	0	54	22.62	21
9000	0	54	22.62	21
10000	0	54	22.62	21
11000	0	54	22.62	21
12000	0	54	22.62	21
13000	0	54	22.62	21
14000	0	54	22.62	21
15000	0	54	22.56	21
16000	8	54	22.62	21
17000	58	54	22.62	21
18000	77	54	22.62	21
19000	84	54	22.62	21
20000	89	54	22.62	21
21000	94	54	22.62	21
22000	95	54	22.62	21
23000	99	54	22.62	21
24000	99	54	22.62	21
25000	98	54	22.62	21
26000	99	54	22.62	21
27000	96	54	22.62	21
28000	90	54	22.62	21
29000	78	54	22.62	21
30000	54	54	22.69	21
31000	10	54	22.62	21
32000	9	54	22.62	21

Figure 12: Sensor Performance Testing

Project Documentation

GitHub Repository:

1. Project: emmagonz22/S.A.G.E
2. Website: emmagonz22/S.A.G.E-Website

Website link:

- https://groundtruthlts.xyz/
- Alternate: https://emmagonz22.github.io/S.A.G.E-Website/

References

[1] “Soil Health Monitoring System” Research Gate, 2020. https://www.researchgate.net/publication/341780058_Soil_Health_Monitoring_System (accessed Feb. 1, 2025)

[2] “MSU soil expert says not all soil test kits provide accurate results,” *Montana State University*, 2025. https://www.montana.edu/news/18714/msu-soil-expert-says-not-all-soil-test-kits-provide-accurate-results (accessed Feb. 1, 2025).

[3] “Temperature Moisture EC PH NPK Soil Tester Portable USB Type-C 8 in 1 Soil Sensor with Free Android App,” *ComWinTop*, 2024. https://store.comwintop.com/products/temperature-moisture-ec-ph-npk-soil-tester-portable-usb-type-c-8-in-1-soil-sensor-with-free-android-app (accessed Feb. 1, 2025).

[4]“Ingress Protection (IP) ratings,” IP ratings, https://www.iec.ch/ip-ratings (accessed Feb. 1, 2025).

[5] A. Gomstyn and A. Jonker, “What is smart farming?,” What is smart farming? , https://www.ibm.com/think/topics/smart-farming (accessed Feb. 3, 2025).

[6] “SparkFun Triad Spectroscopy Sensor - AS7265x (Qwiic),” *Sparkfun.com*, 2025. https://www.sparkfun.com/sparkfun-triad-spectroscopy-sensor-as7265x-qwiic.html (accessed Feb. 3, 2025).

[7] “Photoresistor Light Sensor Module” *Amazon.com*, 2025. https://www.amazon.com/WWZMDiB-Resistor-Comparator-Intensity-Detection (accessed Feb. 3, 2025).

[8] “Photoresistor Photo Light Sensitive Resistor” *Amazon.com*, 2025. https://www.amazon.com/eBoot-Photoresistor-Sensitive-Resistor-Dependent (accessed Feb. 5, 2025).

[9] “8 Precision Agriculture Trends in 2025,” StartUs Insights, https://www.startus-insights.com/innovators-guide/precision-agriculture-trends/ (accessed Feb. 5, 2025).

[10] H. Wang and S. Xiao, “Rapid detection of soil heavy metal pollution using hyperspectral data and Multiscale Spatial Network,” Environmental Technology & Innovation, https://www.sciencedirect.com/science/article/pii/S2352186425000173?via%3Dihub (accessed Feb. 5, 2025).

[11] S. Kummerl, D. Mantle, N. Patel, R. Munje, and P. Chen, “Dust and water resistance testing methodology for environmental sensors,” International Symposium on Microelectronics, vol. 2019, no. 1, pp. 000423–000427, Oct. 2019. doi:10.4071/2380-4505-2019.1.000423

[12] L. Maa, A. Li, H. Yu, and G. Chen, “Hyperspectral remote sensing estimation of soil nutrients in the black soil region based on Computer Vision Model,” ScienceAsia, vol. 48, no. 3, p. 287, 2022. doi:10.2306/scienceasia1513-1874.2022.035

[13] “Soil Test Results (Agronomic Crops) | Soil Testing Laboratory,” *Umn.edu*, 2024. https://soiltest.cfans.umn.edu/soil-test-results-agronomic-crops (accessed Feb. 10, 2025)

[14] Y. Peng et al., “Estimation of soil nutrient content using hyperspectral data,” MDPI, https://www.mdpi.com/2077-0472/11/11/1129 (accessed Feb. 10, 2025).

[15] “Interpreting your soil test results,” *ontario.ca*, 2021. https://www.ontario.ca/page/interpreting-your-soil-test-results (accessed Feb. 11, 2025).

[16] “Soil Testing and Soil Testing Labs | University of Maryland Extension,” *extension.umd.edu*. https://extension.umd.edu/resource/soil-testing-and-soil-testing-labs/

[17] IFAS Communications, “Soil Testing - UF/IFAS Extension,” *Ufl.edu*, 2019. https://sfyl.ifas.ufl.edu/agriculture/soil-testing/

[18] “Near-infrared spectroscopy for soil analysis - smart farming: Nutrient testing,” AgroCares, https://agrocares.com/near-infrared-spectroscopy-for-soil-analysis/ (accessed Feb. 11, 2025).

[19] J. Wetterlind, B. Stenberg, and R. A. Rossel, “Soil analysis using visible and near infrared spectroscopy,” Methods in Molecular Biology, pp. 95–107, Oct. 2012. doi:10.1007/978-1-62703-152-3_6

[20] “Analytical techniques for analysis of heavy metals,” Odinity, https://www.odinity.com/analytical-techniques-analysis-heavy-metals/ (accessed Feb. 12, 2025).

[21] “Interpreting Your Soil Test Reports,” *Penn State Extension*. https://extension.psu.edu/interpreting-your-soil-test-reports

[22] Y. Kulkarni, K. Warhade, and S. Bahekar, “Primary nutrients determination in the soil using UV ...,” IJEERT, http://www.ijeert.org/pdf/v2-i2/33.pdf (accessed Feb. 13, 2025).

[23] R. Jaiswal, “Soil Fertility Dataset,” *Kaggle.com*, 2023. https://www.kaggle.com/datasets/rahuljaiswalonkaggle/soil-fertility-dataset?select=dataset1.csv (accessed Feb. 13, 2025).

[24] S.A.G.E Group “Soil Analysis & Ground Evaluation”, groundtruthlts.xyz, 2025. “https://groundtruthlts.xyz/” (accessed April 14, 2025)