SONOMA STATE UNIVERSITY | ENGINEERING SCIENCE

Senior Design Project Progress Report
EE 493 Senior Design Project

Remote Soil Moisture Monitoring

By:

Chris Campbell
Josh Lyman
Charlie Sterling

May 15, 2025

Faculty Advisor: Dr. Farid Farahmand, Sonoma State University
Client: Cali Pearce, Preserves Manager at the Center for Environmental Inquiry

Project Website: https://ssupreservedashboard.com

Acknowledgments

We would like to give special thanks to Dr. Farid Farahmand for the expertise he has shared with us, as well as the continued guidance on this project. A thanks to Cali Pearce from the Center for Environmental Inquiry for her assistance with accessing and navigating the Fairfield Osborn Preserve. Thanks to Kerry Wininger for her help in securing funding for this project. And a final thanks to the Norwick Memorial Fund for providing us with funds for this project.

Abstract

In this project, we have designed and implemented a distributed system of devices that are used to monitor soil moisture at remote locations on the Fairfield Osborn Preserve. This system includes sensor devices that collect and transmit data, and includes a custom made gateway for LoRa packet forwarding and data processing. From this gateway, the data is sent to our custom website where it is visualized. The system has controllable parameters that users can set and update through an online dashboard. The user is able to change the frequency of data collection, the time data is uploaded to the online database, and monitor battery and connectivity status of deployed nodes. Aggregated data can be downloaded from the website, allowing for users to effectively get all of the data. The online database also includes location data for each node, as input by the user. This system allows users to collect and view soil moisture data without the need for manual data collection or the need for expensive alternatives.

Table of Contents

Abstract **2**
Table of Contents **3**
List of Figures **4**
List of Tables **6**
1. Problem Statement **7**
2. Introduction **8**
3. Previous Works **9**
4. System Overview **11**
4.1. Methodology 11
4.2. Requirements 11
Marketing Requirements 11
Engineering Requirements 12
4.3. Theory of Operation 14
5. Implementation **16**
5.1. Overview 16
5.2. System Architecture 17
5.3. Alternative Design Matrix 21
5.4. Budget 23
5.5. Project Schedule 24
6. Tests Conducted **27**
6.1. Summary of Test 27
6.2. Description of Tests 28
Function Tests 28
FT1 - LoRa 915 MHz Range Test 28
FT2 - Connect TTN server to our database and website 30
FT3 - Test web application for user interface 31
FT4 - Test communication from Gateway to Node 32
FT5 - Testing Device Enclosure 32
FT6 - Sensor Calibration 33
FT7 - Testing interruptions of Gateway service and testing LED status indicator 34
FT8 - Verifying reported ADC values align with measured 35
System Tests 36
ST1 - Testing Device communication with the Dashboard 36
ST2 - Parameter Changes for Nodes 38
ST3 - Power consumption and Battery life 38
ST4 - Full Autonomy Test 39
7. Ethics **40**

8. Challenges **41**
9. Conclusion **42**
References **44**
Appendix **45**

List of Figures

Figure 1: LoRa Class A visualization Page 14

Figure 2: RF Link Power Budget Page 15

Figure 3: System Overview Page 17

Figure 4: The communication between the sensor node and gateway Page 17

Figure 5: The hardware block diagram of the nodes Page 18

Figure 6: Hardware diagram of the Gateway Page 19

Figure 7: The software design of the sensor nodes Page 19

Figure 8: The software design of the gateway Page 20

Figure 9: The software design of the Dashboard Page 20

Figure 10: Critical Path Chart Page 26

Figure 11: Setup For FT1 Page 29

Figure 12: Map of 4 points Page 29

Figure 13: Setup for FT2 Page 30

Figure 14: Received dummy data from the test device Page 31

Figure 15: Setup for FT3 Page 31

Figure 16: Setup for FT4 Page 32

Figure 17: The calibrated sensor and our sensors in the soil for testing Page 33

Figure 18: The graph of the reported values of the Tensiometer and the 200SS sensor Page 34

Figure 19: The setup for FT7 Page 34

Figure 20: Setup for FT8 Page 35

Figure 21: Uncalibrated ADC measurements Page 36

Figure 22: Calibrated ADC measurements Page 36

Figure 23: Setup for ST1 Page 37

Figure 24: Analysis of missed communication Page 37

Figure 25: Setup for ST2 Page 38

Figure 26: The Setup for ST4 Page 39

Figure 27: Graphed data from autonomy test Page 39

Figure 28: Large map with all data points Page 45

Figure 29: Larger Version of water tension graph Page 47

Figure 30: Fast Drying Curve Page 48

Figure 31: Readings of water tension across 5 days Page 48

List of Tables

Table 1: Comparison of Wireless Communication Technologies Page 22

Table 2: Comparison of Microcontrollers Page 23

Table 3: Comparison of LoRa Gateways Page 23

Table 4: Parts list with prices and a short description Page 24

Table 5: Gantt Chart Page 25

Table 6: Summary of conducted tests. Page 27

Table 7: Values associated with the points on the map Page 30

Table 8: Analysis of sleep duration Page 38

Table 9: Values for all map data points Page 45

1. Problem Statement

The problem the Center for Environmental Inquiry is facing at the Fairfield Osborn Preserve on the Sonoma Mountain is the lack of ability to measure data about the preserve remotely. One such measurement that our client wished to conduct is a study on how the presence of grazing cattle affects the soil in terms of its ability to hold water. If the client wished to measure the amount of water it would require people to go out into the preserve and measure each location individually and repeatedly over time. This is completely impractical and thus limits the research that can be conducted on the preserve.

Our team has addressed this issue by creating a system on the preserve that allows for remote sensors to be installed in isolated locations and aggregates the data into a single location for the staff of the preserve. The sensors are measured by battery powered microcontrollers inside waterproof enclosures that use LoRaWAN to communicate wirelessly to a Raspberry Pi installed in the education center. The Pi acts as a LoRaWAN gateway and facilitates the uploading of measurement data to a website our team created. By using this website, preserve staff can view the data measured in aggregate and control how frequently measurements are taken.

2. Introduction

For the Fairfield Osborn Preserve collecting accurate data at remote locations is imperative for research. The issue with this is that collecting data from those remote locations would require regular hikes off the trail, adding to the time commitments of researchers and the overall cost of the project. Additionally there are no current sensors on much of the preserve monitoring soil condition. The areas that data would be collected from are remote and off-grid, meaning that any sensing or transmitting device would have to be either self-sufficient with solar panels, or would have to be designed to last on a battery for the duration of the data collection.

In this work our goal was to create a network of nodes that are capable of measuring and transmitting data to a gateway through the forest cover and elevation changes of the area. The system we implemented sends sensor data from the remote part of the preserve to the Education Center with LoRa modulation and the LoRaWAN technology. The collected data is then uploaded to a database we created and can be visualized in graphs allowing for the users to see the data plotted across time. Additionally we created and provided the clients with thorough documentation detailing all levels of system design, installation, and maintenance.

In the next section we review some of the products that already exist that fulfil a similar role to the system we created. These other implementations gave us great context for what solutions may work and what solutions may be ineffective for us for our product while we were in development.

3. Previous Works

While in the planning stage of our project's development, we looked at existing literature to examine what types of solutions to our problem already existed. This gave us insight into where we should start development and what had already been tried and failed. The system we created did not end up being a mesh network as described in the articles we examined, though the details of the project remained similar. As a result of this and the fact that the literature review was one of the first tasks we did for this project, the views expressed are not necessarily the same as those held at time of the project's completion.

Similar systems to the one we are developing have been created in the past and used to great effect. They provide an alternative to using traditional communication infrastructure when doing so would be cost prohibitive or impossible and can dramatically increase the amount of data that can be gathered by a group or individuals. [3]

John Porter et al. discuss a very similar system they created in the paper "Wireless Sensor Networks for Ecology." In their paper, they address a desire for better research access to a remote area in Taiwan that has a wealth of information about remote regions that are much less impacted by human interference. As a necessary part of that however, the area is difficult to access and measure, especially when the type of research being conducted is on quickly occurring events, such as storms, or when data should be collected faster than people can measure manually. Though the nature preserve we are conducting research at is not as remote as the area studied in the paper, it poses the same problem for data collection.

The solutions discussed in the paper are somewhat applicable to the application we have as they are primarily focused on using serial interfacing or the Ethernet protocol for communication between edge sensors and the location where data is aggregated for study. Unlike with a remote location like that discussed in the paper, the location we are working at has no need for on site data storage as there is a persistent internet connection available at the preserve, it is just too far away from the location that is being studied for WiFi to reach. Though the communication may not use the same physical infrastructure as what was discussed in the paper, it gives a good idea of what issues to look out for as we develop our infrastructure.

Pre-existing systems rely on a variety of network elements and organization strategies to succeed, mainly determined by environmental and resource constraints. Prabal Dutta breaks network elements down into three categories: root, mesh, and leaf nodes. Root nodes, which connect the sensornet to external networks, are considered to be resource-rich; they are typically wall powered always-on devices with 32-bit or even 64-bit processors and significant onboard storage. Mesh nodes deal with their own sensor data as well as data sent from other nodes and are typically resource constrained and operate on limited on-board energy sources, such as power from a battery or solar cell. Leaf nodes are similar to mesh nodes in their power limitations and computing power, but do not receive data from other nodes; instead, leaf nodes send collected data to nearby mesh or root nodes.

The location we are working with will most-likely require all three node types in order to create a reliable mesh system. There is reliable access to wall power and WiFi that would support a root node, but the specified area of study is well beyond the reach of

said WiFi network. Selectively placed mesh nodes could be spread out around the surrounding location to receive and transmit data collected by leaf nodes (which would house all the required sensors) back to the root node. Both the mesh and leaf nodes would require some form of battery or solar power (or a combination of the two) as there is no available power source within the data collection area.

Multi-hop wireless mesh topology brings several benefits over a more common star topology, including reliability, reach, and power efficiency. [2] Mesh networks are able to route around anomalies that cause shadowing and reduce multipath fading by offering multiple pathways through different peers (Dutta). This improves overall performance in challenging RF environments such as heavily wooded forests, buildings, and hilled areas.

Multi-hop mesh networks also bring benefits in power consumption, as multiple short transmissions each require less power than a longer transmission.

As we will be working in a heavily wooded environment with significant elevation changes and no line of sight between our prospective root node and the farthest mesh node, the possibility of transmitting data using mesh nodes over short distances to avoid environmental blockers is a solid prospect, considering constructing a large enough antenna to clear all obstacles would be difficult given the location and resources available, and the monthly costs associated with LTE and other similar communication systems would quickly add up given multiple devices.

Once the data is collected and aggregated at the root node, the final step that must be taken before giving access to the end user is the data processing level. Because the system would be out in uncontrolled environments there could be issues with the integrity of the data collected as a result of the weather or other natural phenomena. Because of this, some manner of error prevention and correction in software is necessary. While preventing data loss from errors in transmission can be done largely with methods such as Hamming codes and data redundancy, there must be a method in place to account for if a sensor stops reporting data correctly or ceases function entirely. A sensor that has become damaged may still be able to send correctly formatted data but that does not mean that the data itself is accurate to the real world input that is being measured. In a paper titled “The Seawater Quality Monitoring and Data Inconsistency Processing System Based on a Long-Range Sensor Network,” Hongji Xu et al. discuss software driven data analysis methods that are used to reduce error in collected data. By utilizing a method similar to those discussed in the paper we can make our system more robust and give end users an easy way to determine if physical issues are interfering with the data that is gathered. [1]

4. System Overview

4.1. Methodology

The devices have solved the proposed problem by implementing a network of nodes in a star topology around a gateway constructed out of a Raspberry Pi that is installed in the education center on the preserve. The nodes collect sensor data that is processed by an onboard microcontroller and then transmitted to the gateway using an onboard LoRa module. The gateway then uploads the data to the The Things Network server. The API on the server then allows us to download the data in a more usable form to the Pi which can apply post-processing to the data and upload it to the database on the website we have created.

We had originally planned to use a rechargeable battery and solar panels to power the nodes but because of the locations they were installed in this solution was not feasible. Instead, we used a single-use lithium thionyl chloride battery that was chosen to give our nodes a lifespan of almost three years. The gateway is powered through a wall adapter in the Education Center of the Fairfield Osborn Preserve and connected to the Internet through their LAN.

Our team's solution differs from the solutions listed above by using LoRa for short range communication instead of costly LTE or more power demanding solutions such as WiFi. The recurring cost for the end users will be minimal as the only expense is the web server and database hosting both of which could be replaced by software on the gateway Pi itself, though this option would limit the accessibility of the system to only the LAN of the preserve.

4.2. Requirements

Marketing Requirements

System Overview

MR1. Node Support: The system must support a minimum of six Nodes.

MR2. Range: Nodes must operate within a 0.75-mile radius of the Gateway.

MR3. Autonomy: Nodes must function autonomously for three years under normal operating conditions (sensor readings and broadcasts to the Gateway every 15 minutes).

MR4. Durability: The system must be resistant to water, dust, and potential damage from animals.

Data Collection and Management

MR5. Data Collection Frequency: The data collection rate must be adjustable via the user dashboard, up to a maximum frequency of once every 15 minutes.

MR6. Data Updates: The data upload rate must be adjustable via the user dashboard, up to a maximum frequency of once per data point collected.
MR7. Data Reliability: Sensor data must be accurate and reliable.
MR8. Data Storage:
- 8.1. Sensor data must be temporarily stored on each Node in the event of communication failure.
- 8.2. Aggregated data must be retained on the Gateway for as long as necessary.

User Interface and Dashboard

MR9. Accessibility: Users must have easy access to collected data through a visualization dashboard
MR10. Battery Monitoring: Node battery status must be viewable via the dashboard.
MR11. Connection Monitoring: Node connection status must be displayed on the dashboard.
MR12. Location Tracking: The location of each Node must be accessible through the dashboard and must be able to be changed via the dashboard.

Deployment and Maintenance

MR13. Relocatable Nodes: Nodes must be designed to allow for relocation after initial deployment.
MR14. Documentation: The system must include a user manual detailing the normal operations of the system as well as how to maintain it in case of issues.

Engineering Requirements

System Overview

ER1. The system must allow communication between a minimum of six Nodes and the Gateway and allow the addition of 4 more Nodes in the future. (MR1)
ER2. Communication must be reliable, with less than 10% packet loss when the Nodes are installed at locations not exceeding 0.75 miles away from the Gateway in hilly forested terrain such as that present on the Preserve. The LoRa signal must have a SNR of greater than -20dB as that is the lowest signal level that can be decoded by our Gateway. (MR2)
ER3. Nodes must be able to operate continuously for three years, given data collection once every 15 minutes, data transmission once every 15 minutes, and with an average battery consumption of no more than 2.5 mAh in a day of operation. (MR3)
ER4. The Nodes must meet IP67 rating to survive weather conditions, and external wires must be resistant to damage from animals through the use of shielding and conduit. (MR4)

Data Collection and Management

ER5. The Nodes must sample data from their sensors autonomously at a frequency that users can change remotely through the dashboard, up to a maximum frequency of once every 15 minutes with a margin of +/- 1 minute. (MR5)

ER6. The system must autonomously upload data collected from the Nodes to a publicly accessible dashboard website at a frequency (minimum once per day) that users can change remotely through the dashboard. (MR6)

ER7. The moisture sensors used must be calibrated before installation, with a calibrated range measuring water tension from 10 to 75 centibars. Error must not exceed +/- 6 CB, two times the max error of the reference device. (MR7)

ER8. If data sent to the Gateway is not acknowledged with a message back it must be stored in nonvolatile memory on the Node to ensure it is not lost. (MR 7)

ER9. The data stored on the Node's nonvolatile memory must be transmitted to the Gateway when the connection is restored between them. (MR 8)

ER10. The data is aggregated by the Gateway and must be uploaded to a website hosting the dashboard, in addition it must be saved locally to a removable storage medium connected to the Gateway. (MR8)

ER11. Data stored on the removable storage device must be updated with data gathered even if the connection with the website is lost. When the connection is reestablished, it must be uploaded to the server. (MR8)

ER12. The communication between the Nodes and Gateway must be bidirectional to allow for verification of data receipt and reconfiguration. (MR 6)

User Interface and Dashboard

ER13. The data collected from the sensors must be available to users as a graph on the dashboard. The graphs must be able to show individual data from single Nodes as well as aggregated data from all of them. (MR9)

ER14. The battery status of all of the Nodes must be available to users as graphs on the dashboard to allow for monitoring of the battery level remotely. (MR10)

ER15. Battery status must be recorded with data collection from the sensors and be transmitted to the Gateway at the scheduled times. (MR10)

ER16. The status of the connection between the Nodes and the Gateway must be available to users of the dashboard. If data is not transmitted from a Node when it was expected, the Node will be flagged as having been disconnected. (MR11)

ER17. The status of the connection between the Nodes and the Gateway must be available to be monitored by users of the Dashboard, recording the last time a message was received from each Node. (MR11)

ER18. The Nodes must be able to be shown at their location on a map to allow users to easily tell which Node's data corresponds to the location of the Nodes as they were installed. (MR12)

Deployment and Maintenance

ER19. The location of each Node must be stored in the database of connected Nodes and be able to be changed by users of the dashboard to allow for reuse of devices and moving them while preserving the data. (MR13)

ER20. The Nodes must be able to be installed by users with minimal tools and knowledge required to allow for easy reuse, movement, or expansion of the system. (MR13)

ER21. The Nodes must be able to indicate to users installing them that they are connected to the network without requiring users to look at the dashboard or any other devices. (MR13)

ER22. The system must have documentation made available to users including instructions on how to install Nodes, how to troubleshoot connection issues, and how to use the dashboard to access and manipulate data. (MR14)

4.3. Theory of Operation

LoRa transmitters have three operation modes divided into classes, Class A, Class B, and Class C. The three classes are separated by their signal receiving behavior, which directly impacts the power efficiency of each class. Class A is the most power efficient, as the node receiver only listens during two short windows after each of its own transmissions, seen in Figure 1. This means nodes are only able to receive configuration updates after a transmission. Class B has periodic RX windows that open up every 128 seconds; this requires synchronization between the gateway and the node in order to time the RX windows and the broadcasts from the gateway. In Class C, the LoRa receiver is constantly listening. This results in the most responsive communication with the node, but also the most power consumption as the radio module never actually goes to sleep.

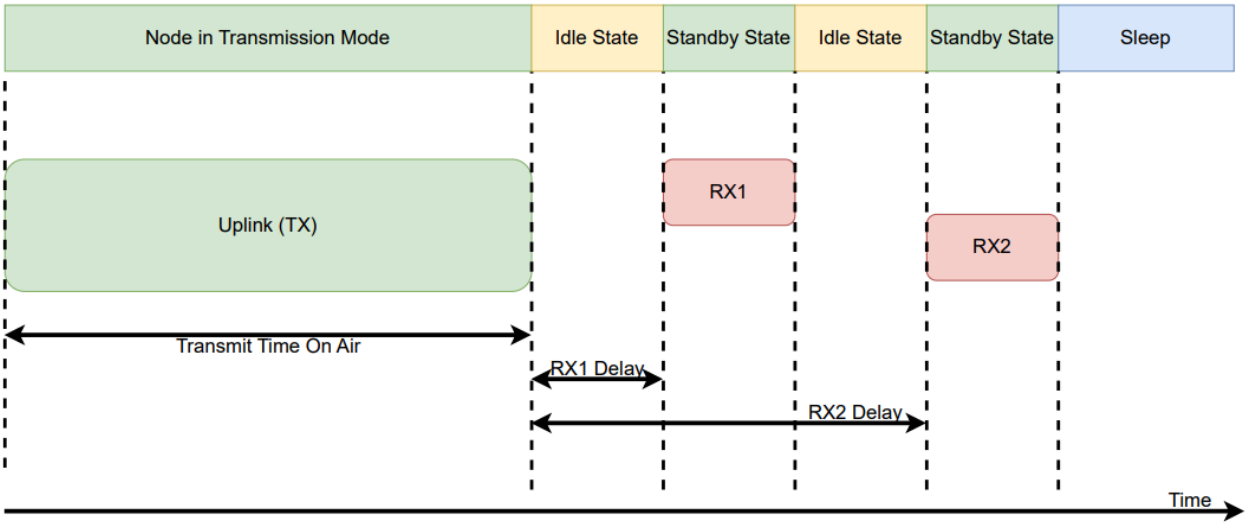


Figure 1: LoRa Class A visualization

The power consumed by the nodes varies greatly by what the node is doing at any given time. For instance, the power used while broadcasting is much greater than that used while receiving a signal. The total energy used in a given day can be determined by determining the amount of time the node will be doing each activity and multiplying by the power used for that activity. We determined the largest contributing factors to be the energy used in transmission, receiving signals, measuring the sensors, the microcontroller's normal running power consumption, and the energy used while the node is sleeping.

Using information from the datasheets of the various components, we were able to calculate that the average power consumption of the microcontroller to be 80 µW, the LoRa module to be 140 µW, the moisture sensors to be 37 µW, the temperature sensor to be 18 µW, and the SD card module to be 1.5 µW. In total the average power consumption

of a node is 275 µW. This number does not include losses, something which we account for by multiplying the final result by 1.5 to get an upper estimate of 413 µW average power consumption.

$$E_{Day} = E_{Micro} + E_{LoRa} + E_{Sensors} + E_{SD}$$

$$E_{Day} = 2.845\text{mWh} + 5.0358\text{mWh} + 1.980\text{mWh} + 0.0528\text{mWh} = 6.609\text{mWh}$$

$$6.609\frac{\text{mWh}}{\text{Day}} * \frac{1\ \text{Day}}{24\ \text{Hours}} = 0.2754\text{mW}$$

We selected a battery that is rated for 3.6 Amp-hours of current at a voltage 3.6 V. Based on this capacity and the power requirements of the node, this should allow for a total lifespan of three years when measuring data and transmitting it every fifteen minutes.

Though the batteries selected will allow for a lifespan of three years with the power consumption as we calculated, the batteries themselves are easy to replace for users of the system. The batteries will use a standard two pole connector to allow for quick replacement with another battery of the same type. The batteries are inside the sealed enclosures of the nodes to protect them from the elements.

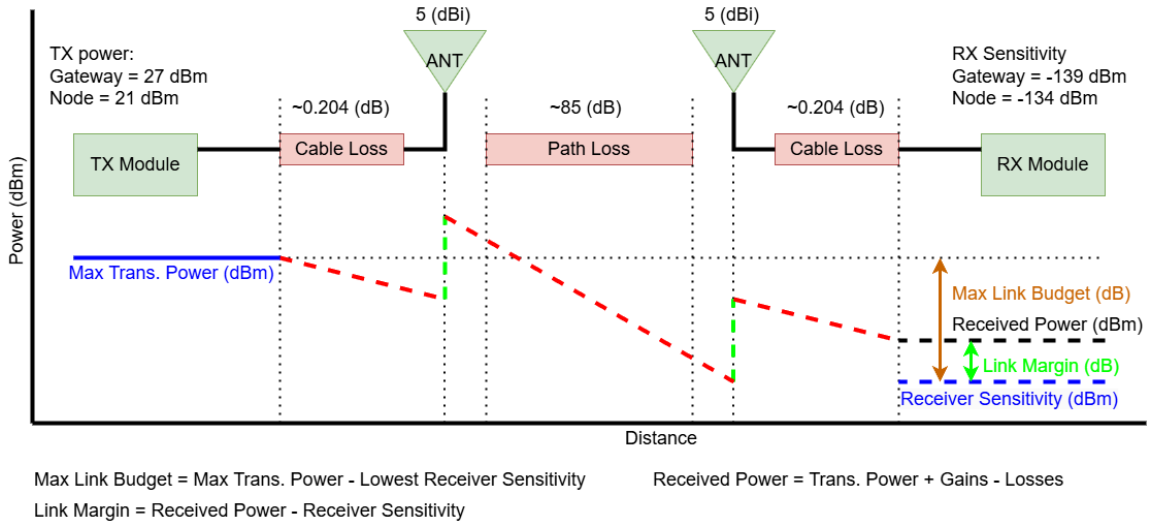


Figure 2: RF Link Power Budget

Data loss could be a prevalent factor in the unreliability of the system as we were unable to make the nodes store their own data locally. We had planned on using a nonvolatile storage medium such as a MicroSD card to keep data until the nodes received a confirmation message from the gateway though limitations with the microcontroller we chose prevented us from implementing this feature. Similarly we had planned to have the gateway store the data locally on an external storage device such as a USB drive but were

unable to accomplish this because of the encryption that TTN uses for data. Both of these challenges could be overcome by redesigning the relevant part of the system.

The scalability of this project has limits. If nodes are placed outside of the previously tested range then connection may not be possible with just a single gateway. It may require retransmission nodes or another gateway closer to the new nodes. If more devices are added the likelihood of device's transmission colliding will increase greatly. For the max amount of nodes you would take transmission times available divided by the amount airtime multiplied by the amount of messages per hour or $Node_{MAX} = T_{available}/(T_{airtime} * M)$. Additionally the addition of more sensors or more power intensive sensors would cause a necessity of a larger battery in order for the node to remain powered for the specified time.

5. Implementation

5.1. Overview

The system we are designing uses multiple modes of communication for multiple purposes. The nodes use LoRaWAN to communicate with the gateway and the gateway uses the Internet to communicate with the TTN server, a MQTT broker, and our server. This combination of technologies allows for power efficient data transmission and communication between all involved devices.

The LoRaWAN network is the heart of the system. The gateway is constructed using a Raspberry Pi and an attached header board that acts as a LoRa signal aggregator and transducer. The nodes use the LoRaWAN to communicate with the gateway and vice versa, transmitting measurements to the gateway and configuration data back to the nodes.

The gateway utilizes a Raspberry Pi image provided by TTN to take these LoRa packets and upload them to the TTN website automatically. This is very convenient though the TTN website is not suitable as a data storage service so further processing is required. The gateway therefore uses the outgoing API that TTN provides to download the data onto the Pi. The gateway then uploads the data to our private SQL database via a PHP based API on our website.

Communication from the website to the gateway is done periodically at a rate of once every five minutes. The gateway connects to the website as a way to let end users know that the device is still running and at the same time it checks to see if the configuration of the sleep duration for the nodes has changed. If the duration has changed then the gateway will send a MQTT message to the TTN server to schedule the downlink to the nodes to update this parameter. This is done in this manner because of limitations in how TTN schedules downlinks and how our server host handles running code we uploaded to it.

5.2. System Architecture

The system used in our project consists of three major components; the nodes, the gateway, and the dashboard. The nodes and gateway have varied hardware and software needs represented below in figure 3. The dashboard contains several parts, with each component represented below with software diagrams. The nodes communicate with the gateway, and the gateway communicates with the dashboard. The dashboard is also able to control nodes through the gateway.

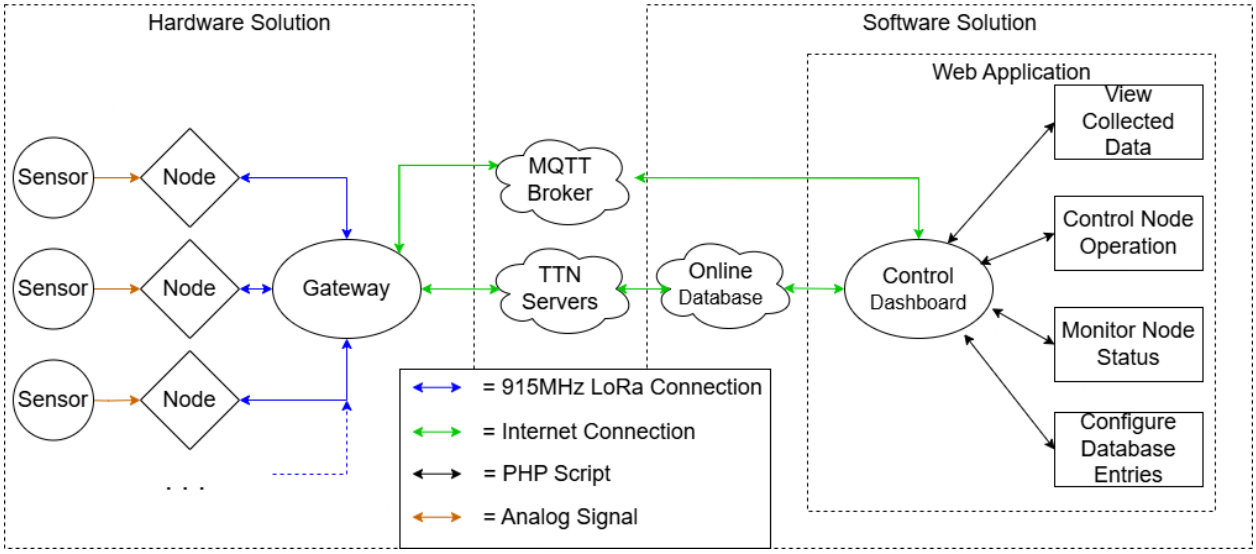


Figure 3: System Overview

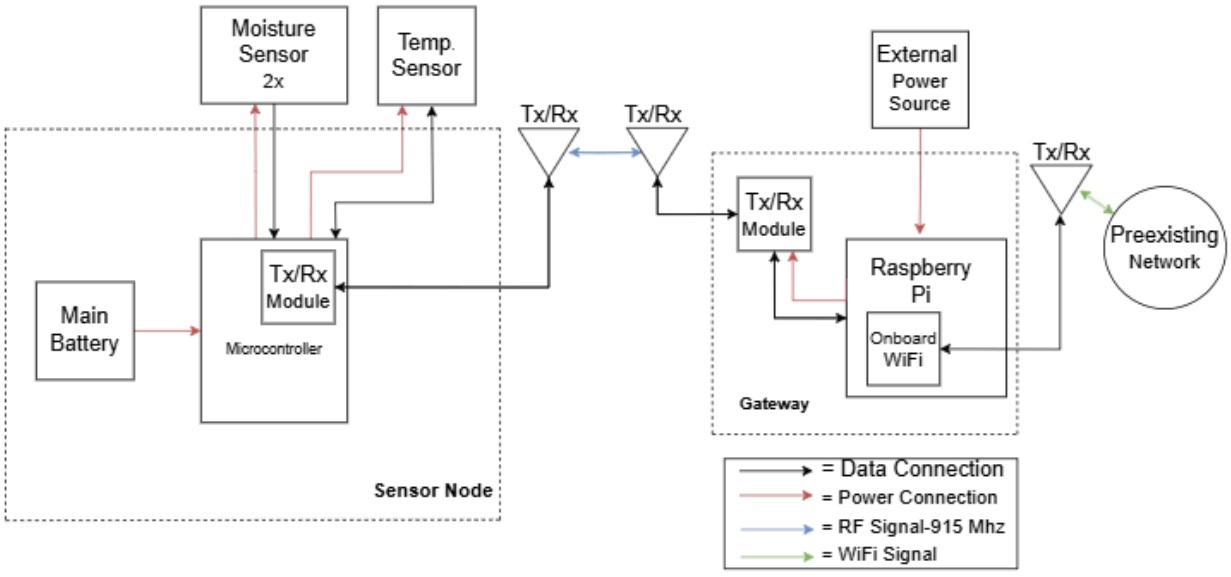


Figure 4: The communication between the sensor node and gateway

The system has been deployed with the nodes communicating to the gateway via LoRa, while the gateway uses WiFi to connect to the dashboard, shown in figure 4. The nodes record their battery's voltage, two measurements of the water tension of the soil and a measurement of the temperature of the soil before sending the data to the gateway.

The sensors used on each node are two Irrometer 200SS Soil Moisture Sensors and an Irrometer 200TS Soil Temperature Sensor. As reported by the datasheet, the moisture sensors measure the pressure required to draw water from the soil in kPa of vacuum and can be calibrated to ± 6kPa over the range from -10 to -75 kPa of vacuum. The temperature sensor has a maximum temperature it can operate at of 150 °C and has a maximum power rating of 30 mW at 25°C derated to 1 mW at 125°C. The accuracy of the sensor is ± 0.2 °C in the range from -55 °C to the maximum operating temperature.

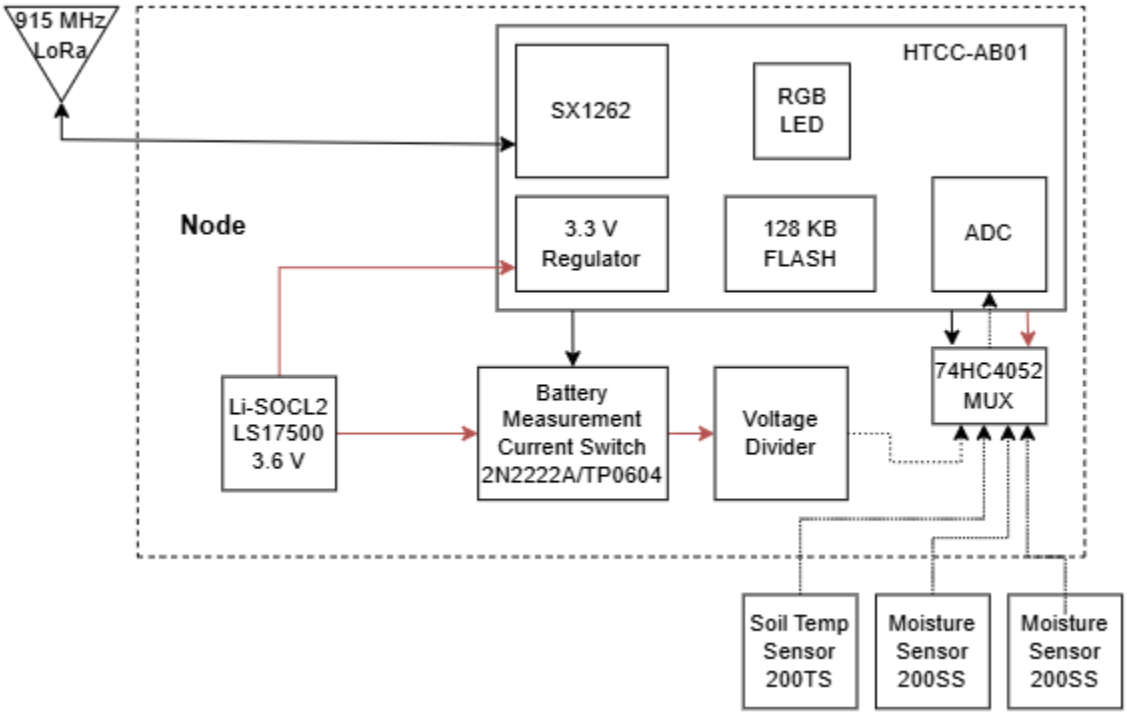


Figure 5: The hardware block diagram of the nodes

Using the LoRa RAK3172 module nodes communicate with the gateway through an antenna. The nodes are powered by a Lithium Thionyl Chloride battery that is regulated to the 3.3 V needed for the microcontroller. The moisture and temperature sensors will be fed through the microcontroller's 12-bit ADC, as well as the battery's voltage. Because the HTCC-AB01 microcontroller only has one analog input pin, we designed a multiplexor circuit that is able to connect each of the sensors to the ADC pin. All of the hardware components can be seen in figure 5.

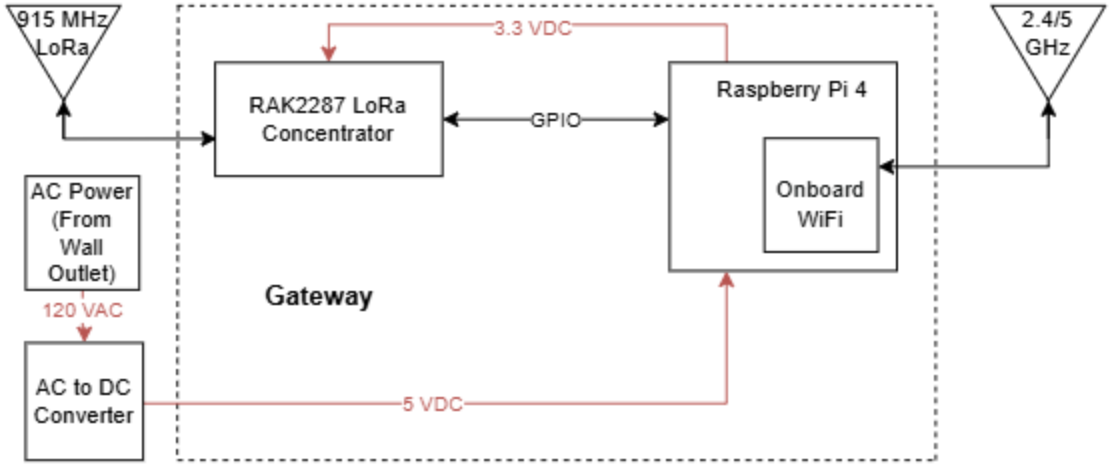


Figure 6: Hardware diagram of the Gateway

The gateway is powered by a wall adapter, and communicates two ways. The communication with the nodes is done through a LoRa Concentrator, which is interfaced with the Pi through the GPIO pins. The gateway communicates to the dashboard through the Raspberry Pi's WiFi module and the existing internet connection present on the preserve. The gateway's design can be seen in figure 6

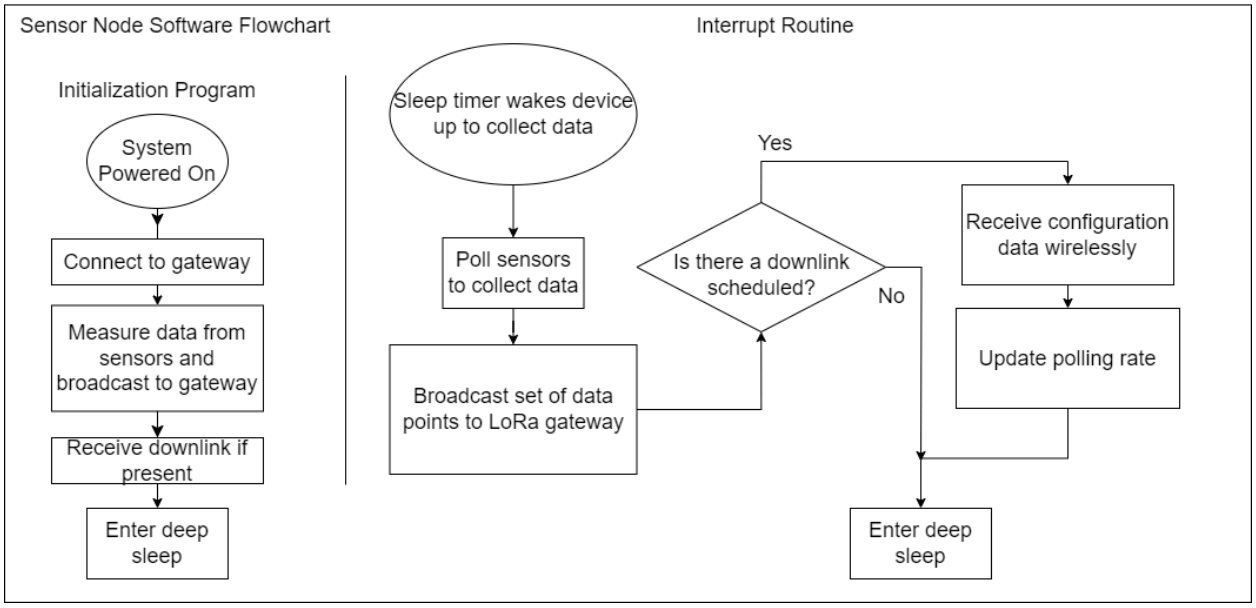


Figure 7: The software design of the sensor nodes

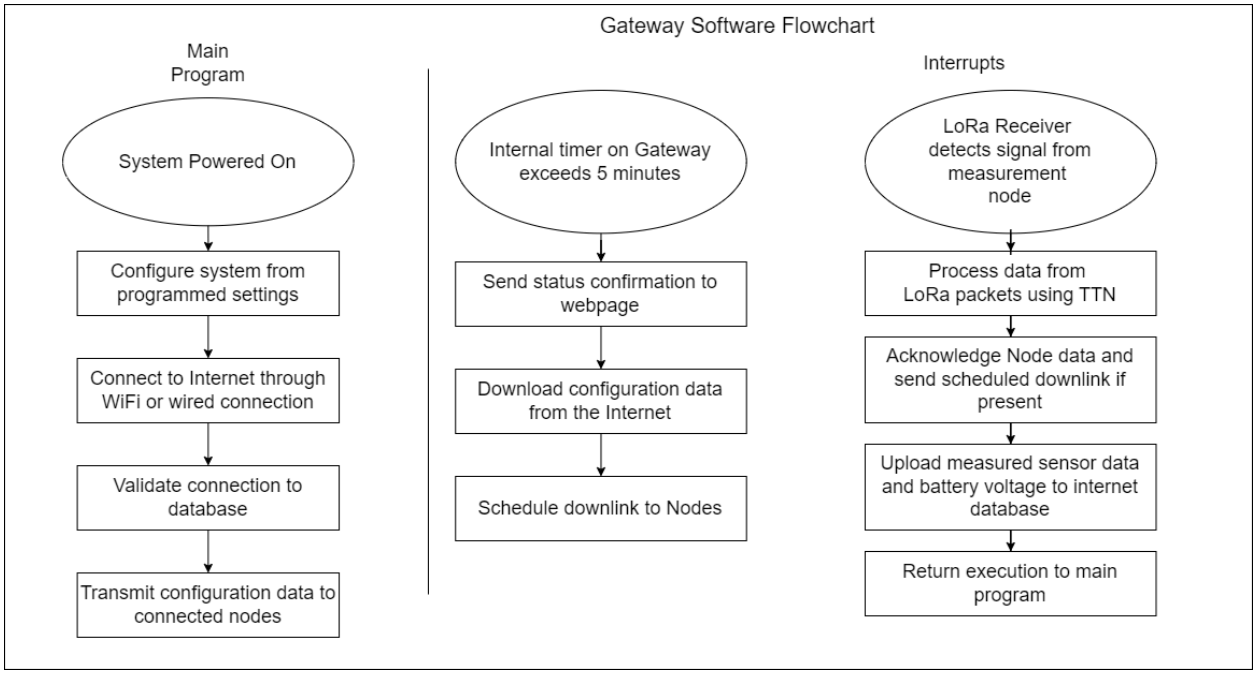


Figure 8: The software design of the gateway

The gateway operates on two main threads, one to listen for LoRa packets from the nodes and the other to interface with the dashboard website. Data from the nodes is processed through the TTN server and then uploaded to our website's database through an API. At the same time, the gateway periodically connects to the dashboard website to verify to users that the system is functioning and download the most recent sleep duration as set by the users. The flowchart can be seen in figure 8.

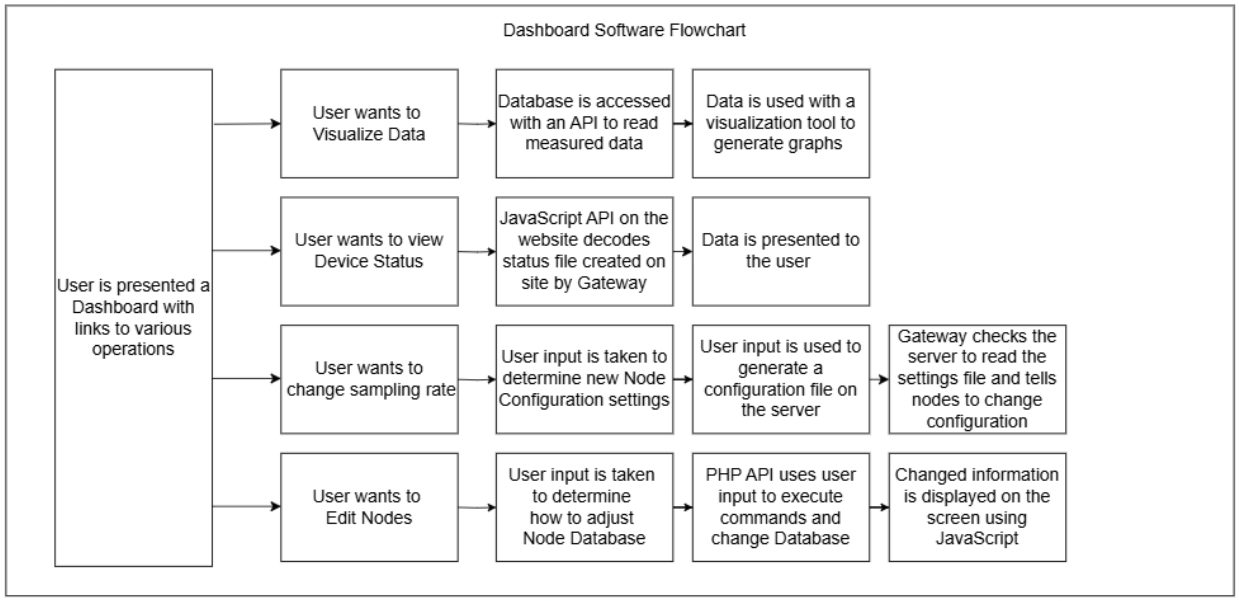


Figure 9: The software design of the Dashboard

The dashboard's design, seen in figure 9, allows for the user to visualize collected data, view the status of the system as a whole, update the duration that the nodes will sleep for, and to edit the stored information regarding nodes and their placement. The nodes are also plotted as points on a map of the preserve via the location that users input when configuring the nodes in the installation process.

The status of the system is determined by comparing the last received measurement from each node to what time the measurement should have come in at given the current sleep duration. If a node has not uploaded data in longer than the period between measurements should be, it is deemed as being disconnected from the system. The data in each measurement is also analyzed and if any of the measurements are outside the bounds of what the value should normally be it is marked as a potential error. Users of the site can see which nodes have had either errors or low battery voltage through the status page.

5.3. Alternative Design Matrix

Below are three Alternative Design Matrices explaining the selection criteria for three components vital to the success of our project. In Table 1, we address the selection criteria for the wireless communication technology to be used. We prioritized range, as the deployment area for this system lacks line of sight and has dense tree cover. Cost was another consideration, as our budget is limited and our clients requested that yearly subscriptions be kept to a minimum, which makes LTE a difficult choice. Ultimately, it was kept as a backup in the case our other options failed. Transfer speed and ease of use were relatively minor considerations, as we are not using sensors that require large data transfers and all three options pose their own unique challenges. Power consumption, while extremely important for a battery-powered device, had to be balanced with the total range of the technology and the required number of devices required to reach the network found at the preserve's Education Center. While BLE Nano had the lowest power requirements, it also had the lowest range, requiring multiple devices to forward packets 0.75 miles to a gateway; this ultimately made it a less appealing choice.

In Table 2, we cover the criteria for selecting the microcontroller to be used in each node. It should be noted that there were three design philosophies to choose from when selecting a microcontroller: select a microcontroller with an integrated LoRa radio module, select a microcontroller and pair it with a standalone LoRa radio module, such as the SX1276, or select a microcontroller and pair it with a SoC LoRa radio module that has its own built in microcontroller, such as the RAK3172 that is based on the STM32WLE5. As for why we decided to go with the third option, combining a STM32F401 microcontroller with a RAK3172, the decision goes beyond what is shown in the design matrix. First, the cost of a standalone LoRa radio module (such as the SX1276) combined with the need to create custom firmware to implement the LoRa stack makes it less appealing than the RAK3172, which is almost the exact same price as a stand alone radio module and has the capability of receiving AT and binary commands to implement LoRaWAN, requiring no custom firmware. Second, while the STM32WLE5 in the RAK3172 is an excellent choice as a standalone microcontroller with built in LoRa capabilities, its form factor is limited unless you design a full PCB around a bare STM32WLE5 chip. This is mandatory to gain access to all the necessary peripherals of

the device, but also because development boards for the STM32WLE5 are significantly more expensive than our other options. Even if we were intending on designing a PCB to support a bare microcontroller chip, which is not easy to due considering our limited time, the STM32WLE5 lacks the sheer number of peripherals and features supported by the STM32F401; one of the most important being SDIO, which will be used to interface an external SD card for long-term data storage. By combining the STM32F401 and the STM32WLE5 found inside the RAK3172, we are achieving several things. First, the process of collecting and processing data and sending/receiving LoRa signals is separated, allowing for isolation between both the devices and code. Not only does this allow for easier debugging during development and in the future, but it also allows the firmware (or even hardware) related to the STM32F401 to be changed or upgraded without having to worry about the LoRa component (as long as you can still send AT commands). However, it was soon discovered that the STM32F401 developer boards we purchased had significant parasitic current draw that would require significant board modifications to remove. As a solution, we pivoted to the HTCC-AB01 microcontroller, which offered low power consumption and an integrated radio module. As a tradeoff, it only had a single ADC pin with no internal multiplexing, which is something we had to deal with further down the line.

In Table 3, the criteria for selecting our LoRa gateway was much more cut and dry. For cost, considering we only require a single gateway for this project, and all entry level gateways cost a similar amount, the price of our selected gateway took minor precedence over its other features. Power, while not normally a concern for a device that's going to be plugged into a wall outlet, is a slight concern for us as the preserve's Education Center is completely off grid, relying on solar power and batteries. That being said, we needed a device that was easy to use, or at least familiar to us (which is the case with the Raspberry Pi). Now, even though off the shelf gateways like the Dragino LG308 are relatively plug and play, they lack any form of (customizable) onboard computing, which we intend on using to process and store sensor data, providing an additional layer of data storage and processing beyond the nodes and database.

Table 1: Comparison of Wireless Communication Technologies

Raw Scores				Weighted Scores		
Criteria	LoRaWAN	BLE Nano	LTE	LoRaWAN	BLE Nano	LTE
Cost	4	2	1	0.364	0.222	0.125
Range	4 (10 km)	3 (0.1 km)	5 (20 km)	0.250	0.250	0.250
Power Consumption	2 (100 mA)	3 (20 mA)	2 (100 mA)	0.222	0.300	0.222
Transfer Speed	2 (27 kbps)	3 (2 Mbps)	5 (200 Mbps)	0.167	0.231	0.333
Ease of Use	3	2	2	0.300	0.222	0.222
Score				1.303	1.225	1.153

Table 2: Comparison of Microcontrollers

Raw Scores				Weighted Scores		
Criteria	ESP32-C3	STM32F401	Cubecell AB01	ESP32-C3	STM32F401	Cubecell AB01
Cost	3 ($8)	4 ($6)	5 ($5)	0.250	0.333	0.417
Program Memory	4 (4000 KB)	3 (256 KB)	2 (128 KB)	0.444	0.333	0.222
Sleep Current	2 (5 uA)	5 (2.4 uA)	3 (2.5 uA)	0.200	0.500	0.300
ADC/IO	4	4	1	0.444	0.445	0.111
Ease of Use	4	3	3	0.400	0.300	0.300
Score				1.739	1.911	1.350

Table 3: Comparison of LoRa Gateways

Raw Scores			Weighted Scores	
Criteria	Dragino LG308	Raspberry Pi	Dragino LG308	Raspberry Pi
Cost	3 ($350)	3 ($200)	0.500	0.500
Power Usage	2 (12W)	3 (5W)	0.400	0.600
Computing	2	4	0.333	0.667
Ease of Use	5	2	0.714	0.286
Score			1.948	2.052

5.4. Budget

Our project was funded through a generous grant from the Norwick Memorial Fund, giving us a budget of $2000 for the full implementation. In order to stay within this budget, considerations were made when choosing sources for each component to develop a working system for the lowest cost possible. Though there were other goals we had wanted to reach beyond the requirements given to us by our client, the cost of the required sensors precluded us from going beyond the required specifications.

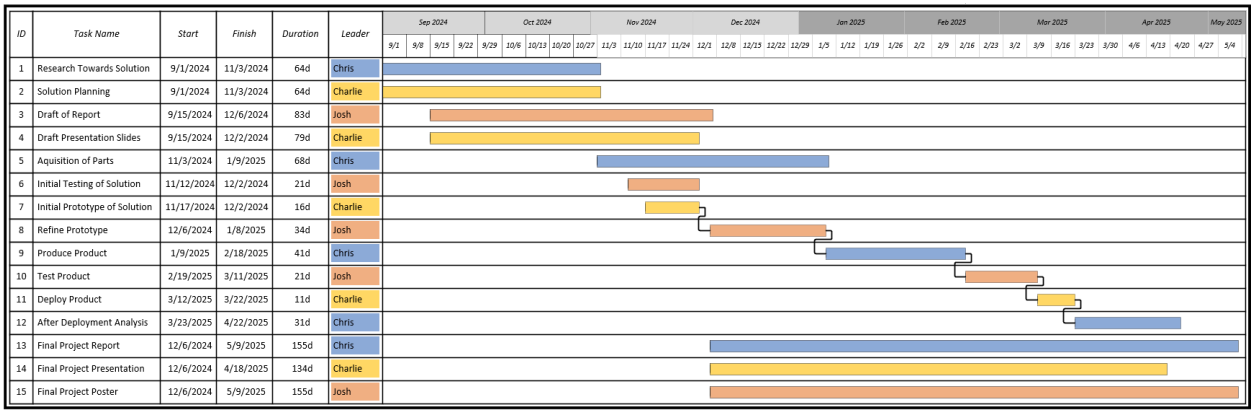
Table 4: Parts list with prices and a short description

Item	Part No.	Purpose	Manufacturer	Supplier	Price	Quantity	Ext. Price
Microcontroller	HTCC-AB01 V2	Controller	Heltech	Amazon	15	6	90
LoRa Concentrator	RAK2287	Gateway	RAKwireless	RAKwireless	155	1	155
Single Board Computer	SC0193(9)	Gateway	Raspberry Pi	RAKwireless	45	1	45
Various Parts	Various	Resistors, Connectors, Wires, etc.	Various	DigiKey	200	1	200
Web Hosting	N/A	Supporting Backend	Hostinger	Hostinger	100	1	100
Main Battery	LS17500	Main Power for Node	SAFT	Amazon	16.39	12	196.68
UV Resin	Fast	Battery Holder	Siraya Tech	Amazon	27.74	3	83.22
Soil Temperature Sensor	200TS	Temperature Compensation	Irrometer	Forestry Suppliers	40	6	240
PCB	Custom	Interfacing Node Components	PCBWay	PCBWay	6.35	10	63.5
IP67 Electrical Box	BG595935	Node Enclosure	Joinfworld	Amazon	15.1	6	90.6
Antenna	AOA-915-5A CM	Improve Signal Transmission	ALFA	Amazon	15	6	90
Soil Moisture Sensor	200SS	Measure Soil Moisture Content	Irrometer	Forestry Suppliers	44.5	12	534
						Final Total:	1887.72

5.5. Project Schedule

While the project does have several critical steps that must be completed in order and on time, there are several steps that can be completed in parallel. These steps include working on the project report, slides, and poster. Additionally, these steps can be worked on throughout the project, and are less impacted by other steps as they can be slowly built upon as the rest of the project progresses.

Table 5: Gantt Chart



ID	Task Name	Start	Finish	Duration	Leader
1	Research Towards Solution	9/1/2024	11/3/2024	64d	Chris
2	Solution Planning	9/1/2024	11/3/2024	64d	Charlie
3	Draft of Report	9/15/2024	12/6/2024	83d	Josh
4	Draft Presentation Slides	9/15/2024	12/2/2024	79d	Charlie
5	Aquisition of Parts	11/3/2024	1/9/2025	68d	Chris
6	Initial Testing of Solution	11/12/2024	12/2/2024	21d	Josh
7	Initial Prototype of Solution	11/17/2024	12/2/2024	16d	Charlie
8	Refine Prototype	12/6/2024	1/8/2025	34d	Josh
9	Produce Product	1/9/2025	2/18/2025	41d	Chris
10	Test Product	2/19/2025	3/11/2025	21d	Josh
11	Deploy Product	3/12/2025	3/22/2025	11d	Charlie
12	After Deployment Analysis	3/23/2025	4/22/2025	31d	Chris
13	Final Project Report	12/6/2024	5/9/2025	155d	Chris
14	Final Project Presentation	12/6/2024	4/18/2025	134d	Charlie
15	Final Project Poster	12/6/2024	5/9/2025	155d	Josh

Gantt Chart not including Slack or Highlighted Critical Path

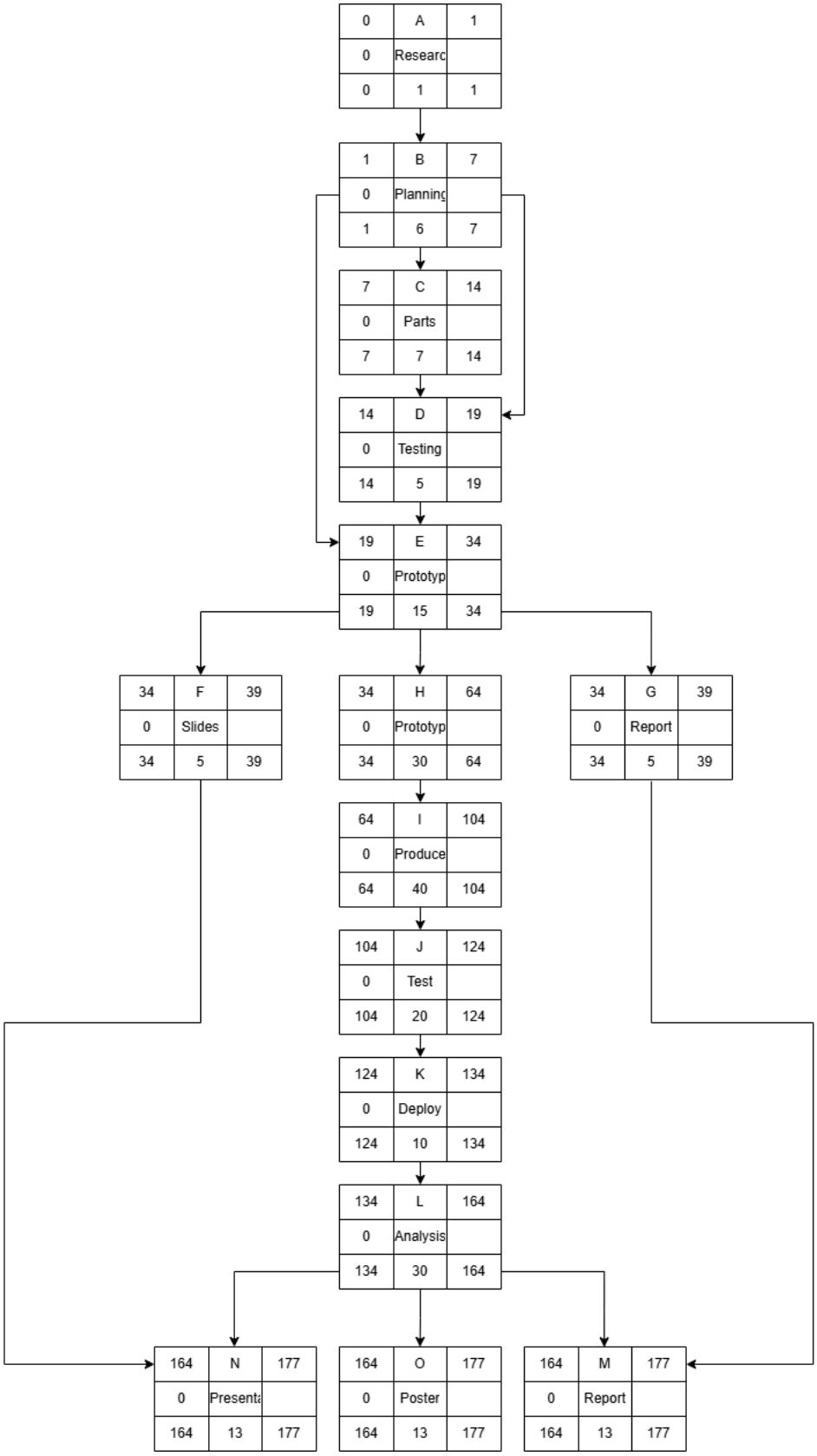


Figure 10: Critical Path Chart

6. Tests Conducted

6.1. Summary of Test

Below, we present a summary of tests that are conducted so far and those that are planned for the future

Table 6: Summary of conducted tests.

Test Number	Test Objective	Related ER	Status	Notes
FT-1	LoRa 915 MHz Range Test	ER2	Completed	Signal test under worst-case conditions completed
FT-2	Connect TTN server to our database and website	ER6	Completed	
FT-3	Create web application for user interface	ER6	Completed	Details will change as project progresses
FT-4	Test communication from Gateway to Nodes	ER4	Completed	Connect Node to serial monitor and print received data
FT-5	Testing Device Enclosure	ER2	Completed	Waterproof up to 40 inches of water for 30 minutes
FT-6	Moisture Sensor Calibration Test	ER7	Completed	Water tension & temperature compensation
FT-7	Testing interruptions of Gateway service and testing LED status indicator	ER21	Complete	

FT-8	Verifying reported ADC values aligns with measured	ER7	Complete	
ST-1	Test Device Comm. with Web Server	ER4,ER6	Completed	Must upload/ download with minimal loss
ST-2	Parameter Change for Nodes	ER4	Completed	Timestamps must be accurate
ST-3	Power Consumption and Battery Life	ER2	In-Progress	Dependent on frequency of data collection & transmission
ST-4	Full Autonomy Test	ALL ER	In-progress	The Nodes are installed and their autonomous behavior is being recorded

6.2. Description of Tests

Function Tests

FT1 - LoRa 915 MHz Range Test

The purpose of this test was to verify that the gateway is able to receive data from the nodes when installed in the client's requested locations on site with no packet loss and received signal to noise ratio of over -20 dB, which is the physical limit for LoRa. We were able to successfully connect a LoRa device to the gateway installed in the Education Center and send messages to it from various points along the trail running through the Preserve.

The setup, pictured in figure 11, involved a borrowed Dragino gateway, a cubcell test device with a 3dBi antenna, and a mapping software used to see if data was received. The test was conducted on the FOP at locations similar to our final locations where tree cover is present.

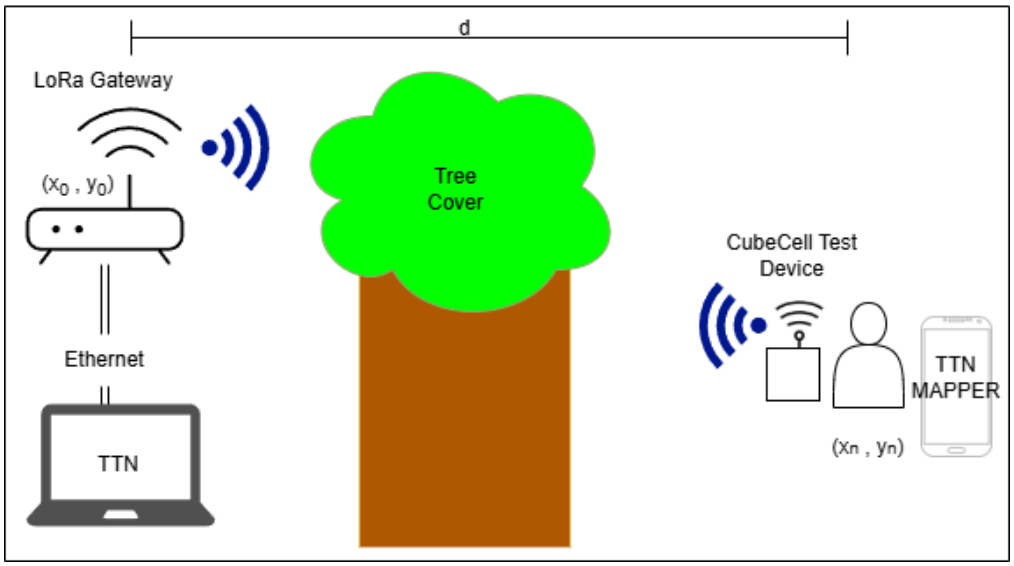


Figure 11: Setup For FT1

We tested 16 locations using this setup, and recorded the SNR and RSSI values for these locations, map seen in figure 12 and the data in table 7. In the appendix is a map of all locations and their corresponding values, and below is a smaller map and table with more information, including the elevation and distance.

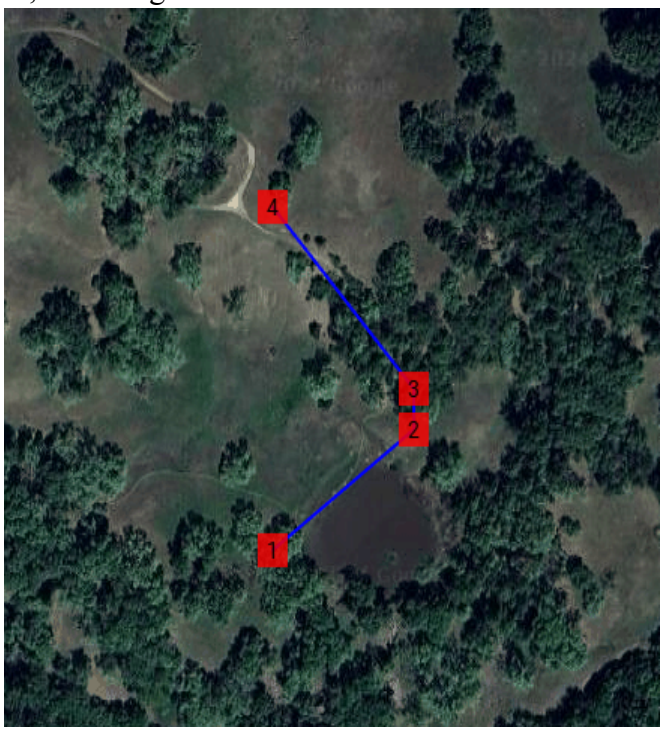


Figure 12: Map of 4 points

Table 7: Values associated with the points on the map

Node	1	2	3	4
Latitude	38.3506	38.3512	38.3514	38.3523
Longitude	-122.587	-122.586	-122.586	-122.587
Elevation (Feet)	2066	2076	2091	2151
Distance from Gateway (Miles)	0.651	0.725	0.727	0.745
RSSI (dB)	-134	-113	-112	-117
SNR (dB)	-15.5	-8	-7	0.5

The signal received was of higher SNR than the minimum required to pass the test and prove LoRa as a viable system however it was still very low. We plan on doing further testing in the future to ensure that this will not drop below the limit of what is detectable by using either better antennas that lose less energy vertically or by using more amplification.

FT2 - Connect TTN server to our database and website

The purpose of this test was to verify that the system can autonomously upload data from nodes to our dashboard website. This was done using the Raspberry Pi gateway, and a LoRa test device. The device communicated with TTN through the gateway, then the data was sent from the Gateway to our database as seen in figure 13.

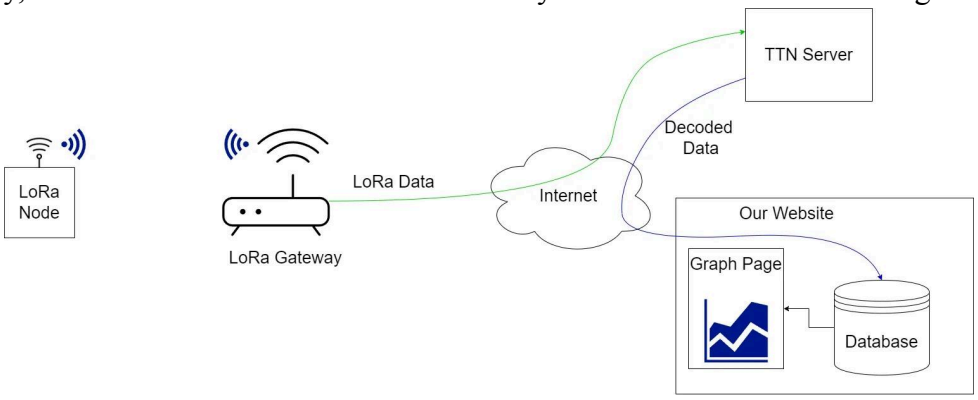


Figure 13: Setup for FT2

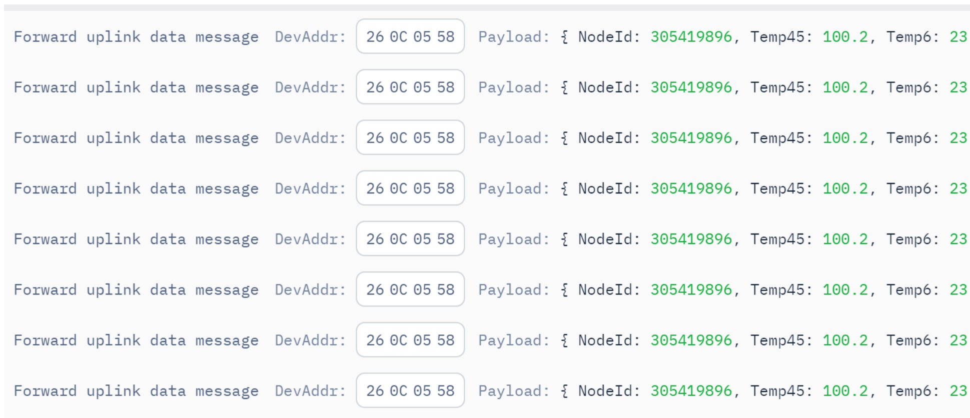


Figure 14: Received dummy data from the test device

The node was connected to the gateway as in the previous test and the gateway was connected to the TTN servers. We were able to successfully connect our Raspberry Pi and other devices to the TTN server with our API key and download the data to store it locally. The received data picture in figure 14. This data was then uploaded to the database we have created using the API we made for that purpose.

FT3 - Test web application for user interface

The purpose of this test was to verify that an end user of our control dashboard can control system parameters through our website.

We created a dashboard website with access to APIs and user input scripts that allowed users to input settings to have the server push to the gateway and then the nodes from there. The script on the page creates a locally hosted JSON formatted file that would then be downloaded by the gateway. The setup is visualized in figure 15.

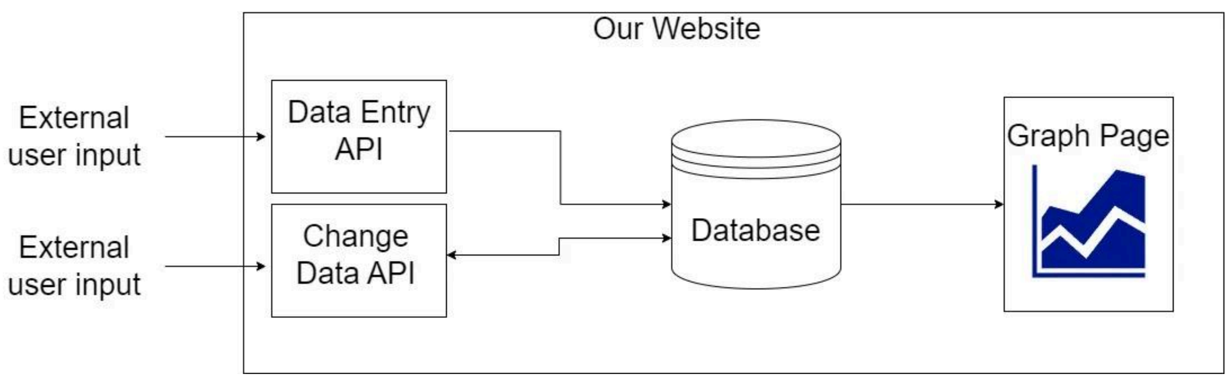


Figure 15: Setup for FT3

The creation of the file from user inputs was a success. When this test was run, our next steps moving forward from here were to attempt to make the MQTT communication channel between the devices and to make the gateway device download the data. We were unable to implement the MQTT protocol on the server as we had wanted to but were able to get the gateway to download data from the server.

FT4 - Test communication from Gateway to Node

The purpose of this test is to ensure that the configuration changes made by the dashboard are seen at the nodes. We determined this by using a serial monitor attached to a test device using the same HTCC-AB01 microcontroller we used in the nodes and scheduling a downlink message through TTN directly via their downlink scheduler, shown in figure 16.

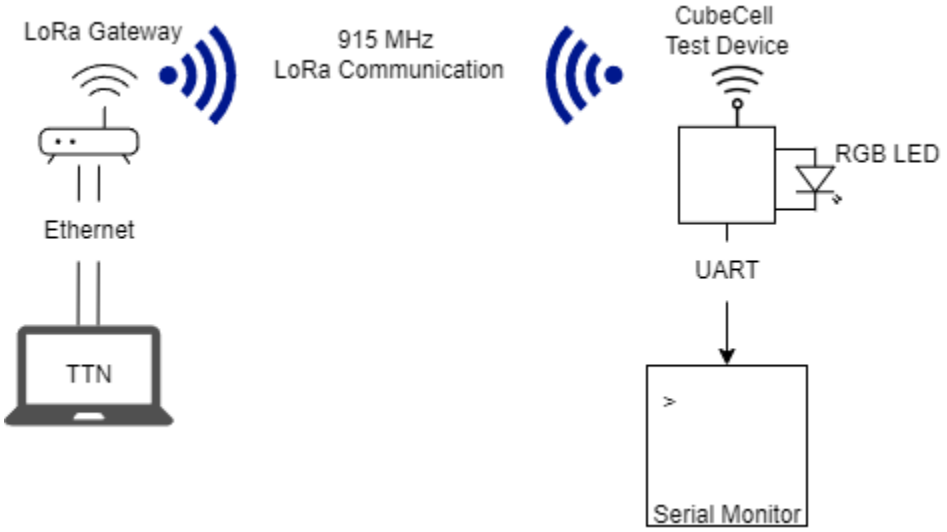


Figure 16: Setup for FT4

FT5 - Testing Device Enclosure

The purpose of this test is to ensure the enclosures are capable of withstanding being outside in rainy and dusty conditions.

The Node enclosures were sealed with absorbent material and weights inside. Once sealed as they would be under normal operating conditions, the enclosures were submerged under 40 inches of water for 30 minutes. Upon reaching 30 minutes of submersion, the enclosures were removed from the water and inspected for any water intrusion. Inspection was visual only; the absorbent material as well as the inside walls and connectors of the Node were inspected for any presence of water.

This test was a success; none of the nodes experienced any form of water intrusion through either the main lid seal or the two bulkhead connectors. Each bulkhead connector was sealed with an o-ring and a rubberized gasket maker on all of the exterior threaded connections; disassembly of the node will require the cleaning and reapplication of sealant on all threaded connections.

FT6 - Sensor Calibration

The purpose of this test is to ensure the soil moisture sensor is within +/- ^ kPa within a range of -10 to -75 kPa. The test also ensures that the temperature sensor is within +/- 0.4℃ of calibrated thermometers. This test was conducted using the buried sensors and known calibrated equipment. Using an IRROMETER Tensiometer, calibrated for a range of 0 to -100 kPa, we tested our own IRROMETER 200SS sensor by having both sensors placed in the same soil, seen in figure 17, with the same amount of water within the soil. An image of the setup is below.

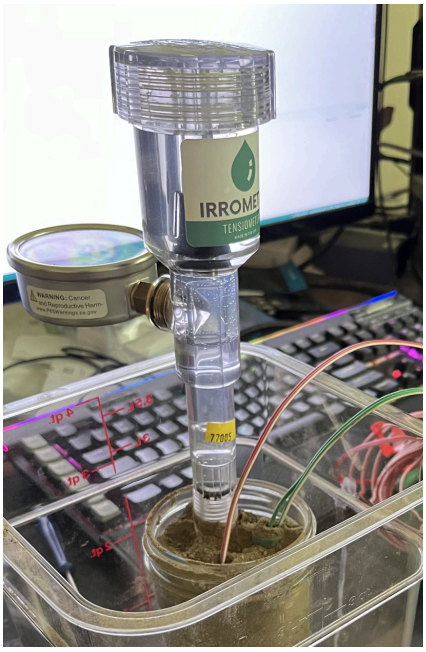


Figure 17: The calibrated sensor and our sensors in the soil for testing

Using this method we calibrated both the temperature and the soil moisture sensor, these sensors communicated via LoRa to send the output data for later graphing. As seen by the plot below the sensors were within our target calibration range.

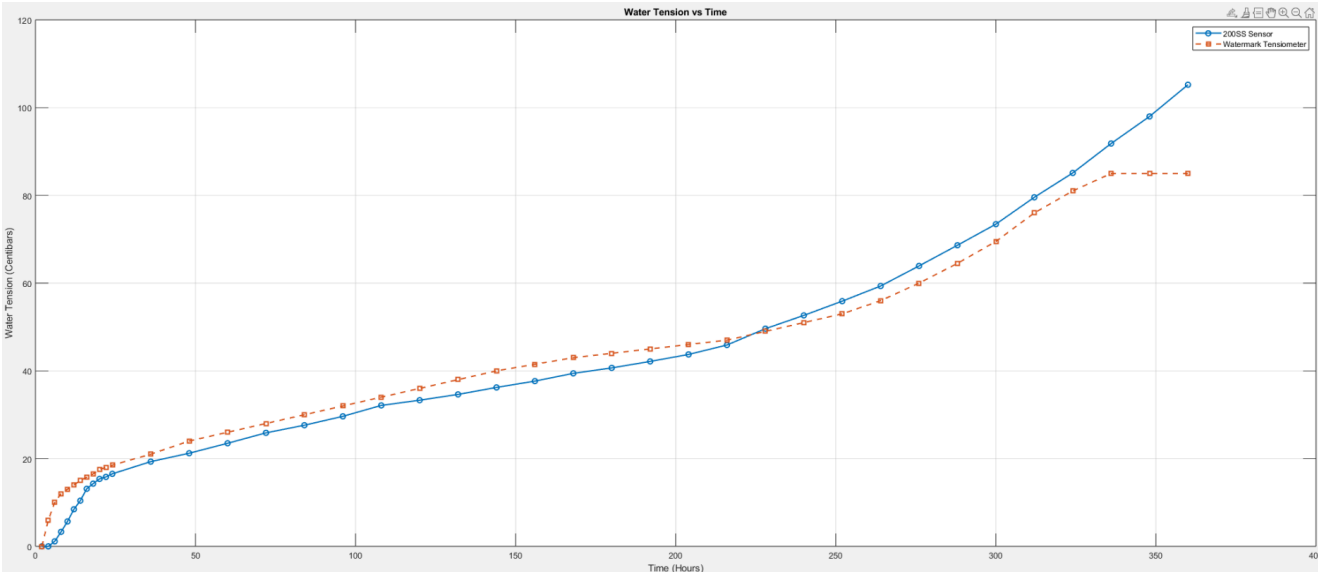


Figure 18: The graph of the reported values of the Tensiometer and the 200SS sensor

Using the data from figure 18, we were able to confirm that the sensor was functioning according to the specification on the data sheet, and that the values were within our target range of values. More graphs and a larger version of the above can be seen in the appendix.

FT7 - Testing interruptions of Gateway service and testing LED status indicator

The purpose of this test is to ensure that the node will continue functioning correctly even if its connection to the gateway is temporarily lost, or if the connection to TTN is interrupted that the nodes will function when connection is restored. This is indicated by the status LED on the microcontroller. This test was completed by taking a microcontroller with the correct node code, and seeing if an LED flash occurs at the correct events or by seeing the serial output of the device.

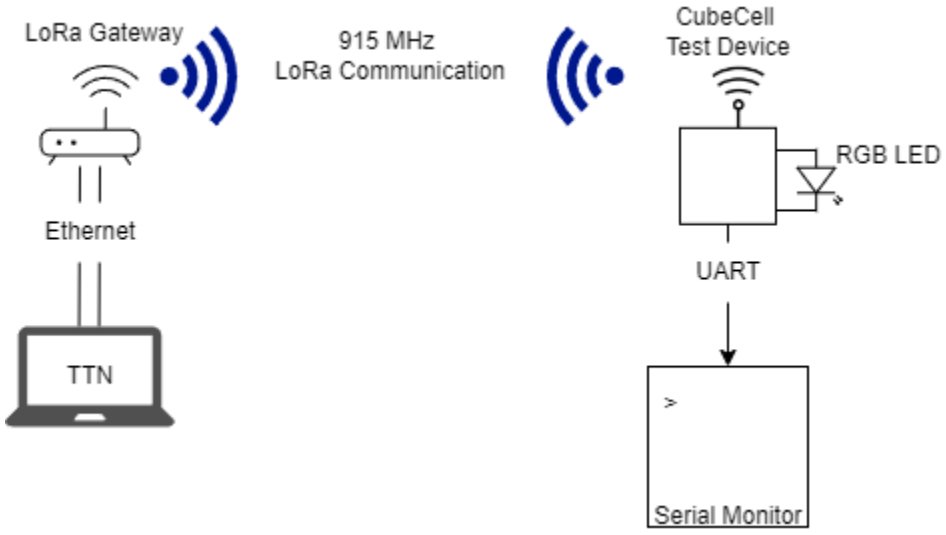


Figure 19: The setup for FT7

The node operated as expected, each event triggered the corresponding LED flash. Each event was repeated, and every repetition repeated the expected results. When the connection to TTN or the gateway itself was interrupted the nodes functioned correctly, continuing to keep their sleep duration constant and continued making measurements.

FT8 - Verifying reported ADC values align with measured

The purpose of this test is to ensure that the 12-bit ADC on the AB-01 microcontroller is reading consistent values. In order to complete this test we used a benchtop digital multimeter (DMM) to ensure that the value reported by the microcontroller was accurate. This test was completed by using the benchtop DMM to read the voltage across a voltage divider that was feeding into the ADC on the microcontroller, while the ADC output was written to the serial monitor.

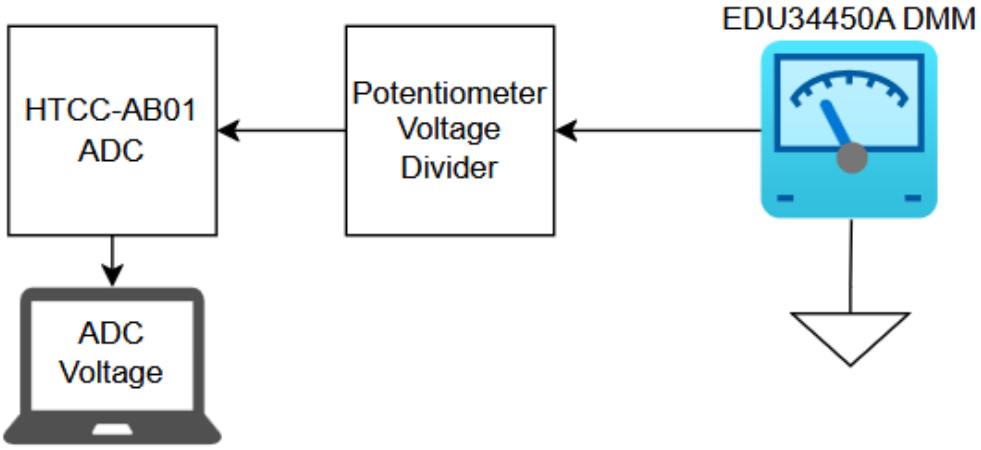


Figure 20: Setup for FT8

Conducting this test we found that the claimed reference voltage was actually 15 mV lower than what was stated on the data sheet. We then recompleted this test with a calibrated reference voltage to see how the difference between reported and measured changed, seen in figure 22. Both tables are below, and tables with all data points are in the appendix.

Claimed 2.4V ADC Reference					
	DMM_mV	ADC_Low_mV	ADC_High_mV	Low_Error_mV	High_Error_mV
1	52.62	52.15	52.73	-0.48	0.10
2	249.57	249.43	250.55	-0.14	0.98
3	508.21	510.29	511.41	2.08	3.20
4	750.95	754.75	755.80	3.80	4.85
5	1005.30	1010.98	1011.68	5.68	6.38
6	1500.30	1508.59	1510.14	8.29	9.84
7	2000.90	2012.34	2013.16	11.44	12.26
8	2381.10	2394.38	2395.66	13.28	14.56

Figure 21: Uncalibrated ADC measurements

Calibrated 2.385V ADC Reference					
	DMM_mV	ADC_Low_mV	ADC_High_mV	Low_Error_mV	High_Error_mV
1	52.48	51.94	52.46	-0.55	-0.02
2	247.38	246.29	246.99	-1.09	-0.39
3	504.15	503.79	504.19	-0.36	0.04
4	749.92	749.24	749.50	-0.68	-0.42
5	1001.10	1000.84	1001.24	-0.26	0.14
6	1502.70	1502.49	1502.99	-0.21	0.29
7	2004.80	2004.61	2005.27	-0.19	0.47
8	2380.10	2379.88	2380.19	-0.22	0.09

Figure 22: Calibrated ADC measurements

This error can be calibrated out, if this test was repeated for each node deployed, however we determined that this was not necessary as the difference in measured values equals to 2 CB at maximum, which is within our accuracy range for the sensors. Adjusting the reference voltage eliminates the error entirely, so if future students wished they could calibrate each node for higher accuracy.

System Tests

ST1 - Testing Device communication with the Dashboard

The purpose of this test is to ensure that the communication between deployed nodes and the dashboard is reliable and accurate.

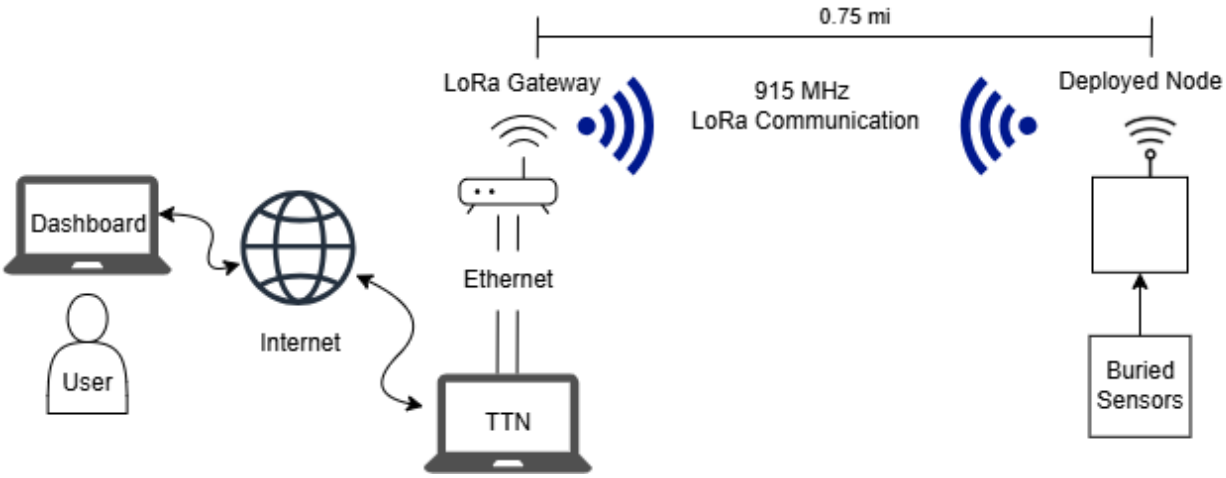


Figure 23: Setup for ST1

We installed the nodes in their final locations on the preserve and connected them to the gateway we had installed in the Education Center. From there we observed the dashboard website and verified that data was being uploaded successfully. Shown in figure 23. This test served as a full system test for uploading data and was a remarkable success.

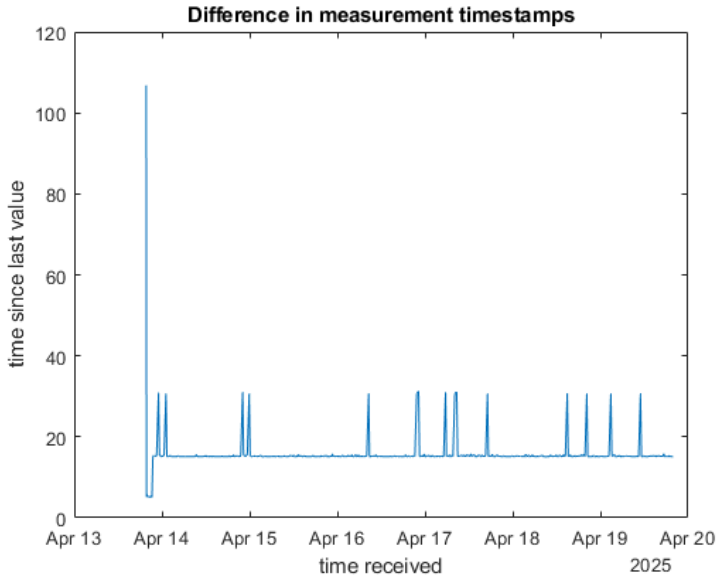


Figure 24: Analysis of missed communication

By analyzing the timestamps of the received data from the website, we were able to determine how many data points were missed compared to the number that we expected to receive from each node, seen in figure 24. Our requirement for the project was to drop no more than 10% of packets from any node and we satisfied the requirement. The node that has the worst signal quality was determined to lose only 2.74% of the attempted transmissions over the week-long observation period of this test.

ST2 - Parameter Changes for Nodes

The purpose of this test is to ensure that the updates from the dashboard are changing the desired parameters on the nodes and that the sleep routine functions as intended.

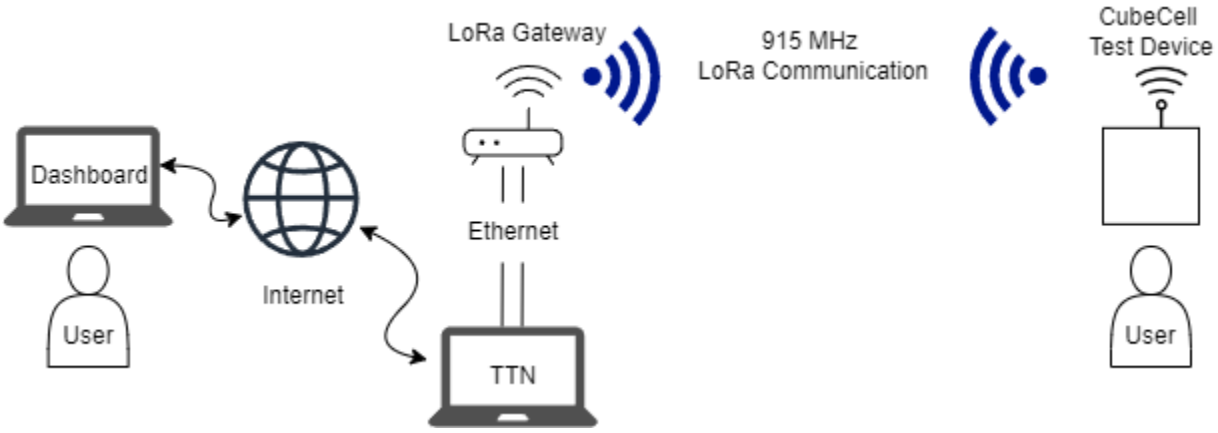


Figure 25: Setup for ST2

We verified that the nodes could have their sleep duration changed via downlink message with them installed on the fields in the process of our installation.

To verify the sleep duration is accurate to the value it should be, we downloaded the aggregated data from the dashboard and analyzed the timestamps of it using MATLAB. From this data we were able to determine that the sleep duration as set on the nodes is within the one minute criteria we set out to achieve.

Table 8: Analysis of sleep duration

Node	Time Delay (s)	Standard Deviation (s)
1	913.5051	11.353
2	911.62	10.0817
3	910.8443	9.2475
4	909.0372	3.5475
5	915.7446	9.9674
6	913.8262	9.2558

ST3 - Power consumption and Battery life

The purpose of this test is to ensure that the battery will last the estimated time. This will be accomplished by measuring the average power consumption used in a day and extrapolated from there.

ST4 - Full Autonomy Test

The purpose of this test is to put the whole system together and have it function while on the FOP. This test verifies that the nodes can function autonomously for an extended period of time when installed on the FOP, and verifies that the downlinks function while the nodes are at their locations on the FOP.

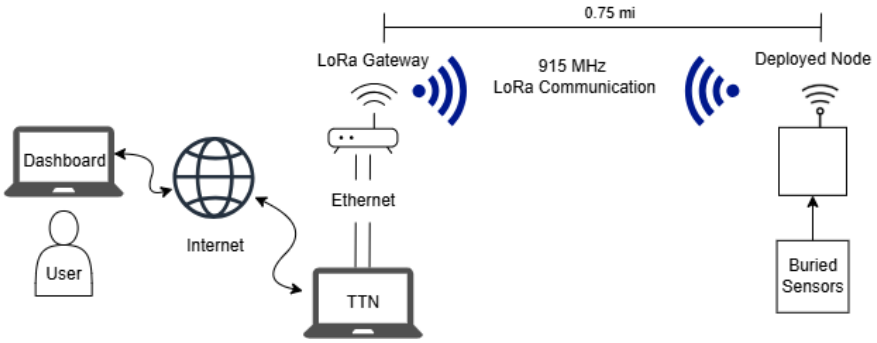


Figure 26: The Setup for ST4

The deployed nodes are monitored based upon the data being visualized on the website, and the status of nodes are monitored through its dedicated webpage. Below is graphed data from 3 days of full autonomy, but the current status of the test can be viewed from the link on the cover page.

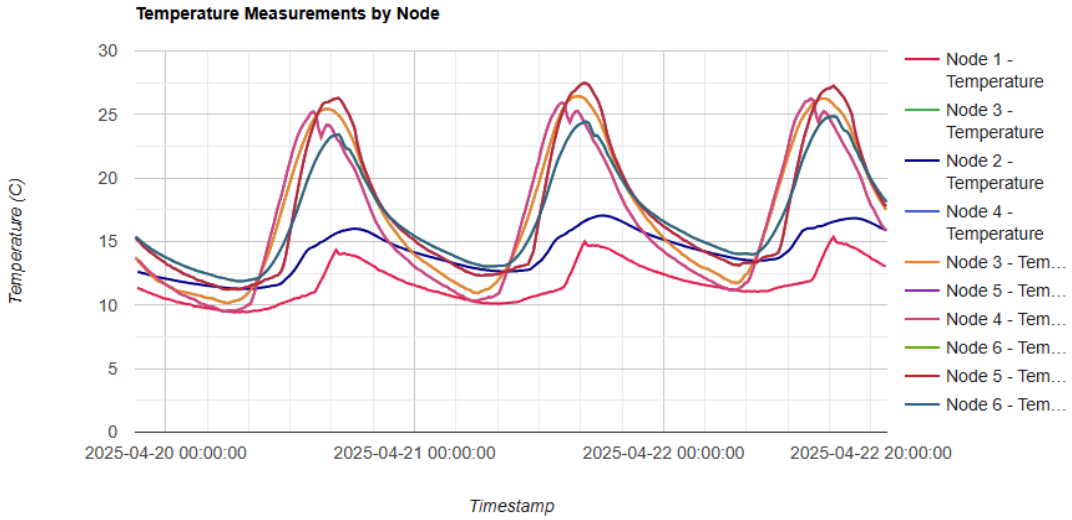


Figure 27: Graphed data from autonomy test

This test is still ongoing, however the nodes have functioned properly for the time they have been installed. The nodes were installed on the 19th of April, 2025. The downlink test was performed while the nodes were on the preserve, and they have been successfully received by the nodes.

The gateway, also deployed on the preserve, has had some issues since its deployment. Its connection to TTN has not been completely stable, likely due to the fact that the WiFi on the preserve is heavily managed with no provided workarounds or documentation as to the specific restrictions. The Raspberry Pi has temporarily lost the ability to complete DNS lookups, causing the Pi to lose connection to TTN's servers, which use dynamic IP addresses. Connection to our webpage has been constant, as it has a static IP address. This further confirms the

issue is DNS related. This issue is currently being addressed, and will require further work during the summer by another student or team.

7. Ethics

Ethical design requires taking several factors into consideration during design. The design of a system that does not exploit the people or the environment, as well as a system that does not use manipulative tactics, respects data privacy, and a system that allows for cooperative design.[4] Our system primarily must contend with the environmental impact of our physical system, as well as data-privacy as users access the online database.

The environmental concerns that are raised by this project are from the single-use battery, from the physical degradation of the devices, and from soil disturbance on the preserve. Single-use non-rechargeable batteries can raise environmental concerns, if not recycled properly as they contain hazardous materials.[5] The manufacturing process to make batteries can be highly toxic, and can leave behind a large environmental effect.

This project will allow for long term data collection of soil moisture on the Fairfield Osborn Preserve, which would allow for continued knowledge about the soil health. This data will help researchers understand the soil condition as a cattle grazing study is conducted. The labor impact of our system would lower the necessity for the researcher to hike out into the field in order to collect data.

When the battery used in our system is depleted it will require replacing with a battery of similar specifications. This will require the user to hike to the locations of the nodes and replace them, ensuring that the now dead battery is retrieved for proper disposal. The batteries used in this project are non-rechargeable, and should be recycled using a battery recycling program.

When the devices used in our project stop working the user will be able to know the status using the dashboard, allowing for the user to then go out and retrieve the device stopping it from continually degrading in the environment. The interface with the device will require only a basic internet connection, and would be highly accessible for a disabled person with a computer interface with accommodations for their disability. Potential harm our devices could cause come from the nodes prematurely degrading, releasing potentially toxic materials into the soil. The enclosure for the nodes will ensure that the node does degrade in the weather conditions that are expected in the environment.

8. Challenges

There were several challenges that had to be overcome for this project to succeed. First and foremost was finding a cost-effective wireless communication method that could transmit sensor data without transmission issues 0.75 miles with heavy tree cover, elevation change, rolling hills, and little to no line of sight. Since cellular services were not an option, per our client's request, our only option was to use a wireless communication method that is more cost effective than cellular transmitters, to relay sensor data back to a preexisting WiFi source. The LoRa 915 MHz band is perfect for this situation.

Though the communication method is a solved problem, we still had to power the devices. Since wall power was not an option for any of the nodes, battery and solar power were our only options. However, to keep costs down, we could not simply install an oversized battery and solar panel for each device; this would increase costs significantly and make potential future scaling difficult due to the overall cost of each device, not to mention the increased installation difficulty. Additionally, some areas in the preserve have heavy tree cover, limiting the amount of sun a panel would be exposed to. As a result, we had to reduce the power consumption of each node through hibernation and other power-saving methods, reducing the need for large batteries and solar panels. Finally, the same environment that dictated the need for batteries and solar panels also dictated the need for resilient housings for each node. The Fairfield Osborn Preserve, while not the harshest environment, will still expose the sensor nodes to a multitude of environmental hazards. These include rain, dust, mud, wildlife, UV radiation, and varying temperatures. The housing containing all of the electronic devices that make up each node must protect said components from damage during their deployment.

When beginning this project we anticipated encountering risks in the design process that would require us to change our design. We were concerned that it was possible that our selected microprocessor either lacked the required peripherals or processing power to carry out its tasks, a concern that we mitigated by carefully selecting our microcontrollers to exceed the requirements to allow for changes that came. We were also very concerned about the ability of LoRa to transmit the data we required which led us to conduct several tests of the wireless capabilities on site at the Preserve. Had we encountered issues with the communication we could have mitigated them by using directional antennas or amplifier circuits.

Additionally, we faced challenges implementing features originally planned for the project; primarily local storage on the nodes and on the gateway. TTN was selected early on in the project, as it was what was familiar to our team and as far as we knew would be sufficient to complete the goals of our project. However, after time and development we found that TTN does not support gateway retransmission, as any packets TTN receive that are out of order or have any issue with their frame counter are simply dropped. This meant that retransmission of locally stored data would be impossible to implement without changing the LoRaWAN gateway software and management service we were using. This could have been alleviated if ChirpStack had been implemented in TTN's stead, this however would also have drawbacks. Using TTN gives additional benefits, such as our nodes being able to send data to TTN even when the nodes cannot connect to our gateway. As TTN boasts interconnectivity, as long as the node can connect

to any LoRaWAN gateway registered to TTN it can send data. During our implementation of this project, our own gateway was having difficulties connecting to TTN, meaning that several of our nodes could not send data, but because the other nodes could connect to another gateway we were still able to see data.

Furthermore, because retransmission was not possible on the gateway, it was not implemented on the node. Because of this local data storage was not implemented on the node. In order to implement node data storage a storage protocol, storage device, and retransmission protocol would have been needed, all of which would have added significant complexity to the nodes. To ensure that the data is not duplicated or lost, implementing storage on the nodes also requires implementing an additional RTC module to timestamp data as well as the physical interface for the MicroSD card. These additional devices would require a redesign of the PCB and add a significant amount of complexity.

Our team also had challenges with the server hosting on Hostinger, as we were not able to implement all of the features we wanted on the website itself. Hostinger was not able to accommodate our use of MQTT. Originally we had wanted the website to send the MQTT message itself, however because MQTT is not a standard communication method that is supported by browsers, we needed to use a python script to schedule and send the MQTT requests. Hostinger allows for JavaScript to be run, but severely limits what could be done with python, and we were unable to configure a JavaScript to complete this task effectively. We are still unsure of the cause of this issue and future work may be able to solve it. In order to circumvent this we used a MQTT broker to schedule the messages. This issue with Hostinger also meant that we had to put the ‘push’ program, the program to push data from TTN to our website, on the Pi gateway, when optimally it would be on a server. A future improvement of this project would be fixing these issues by hosting our website on our own server, allowing for these programs to be run and edited remotely, as well as, avoiding the need of more third party software.

9. Conclusion

The system we designed functions well enough to complete the goal we had set out to meet. The system is able to remotely monitor soil moisture and report the measurements to our website where it can be easily visualized. The nodes communicate across the Fairfield Osborn Preserve to our gateway using LoRaWAN, and our gateway is able to report that data to our website via WiFi. The implementation is not perfect, as there were several features we wanted to implement but were unable to, and there are several improvements that could be made to the project.

There were several areas in the project that could be improved upon in the future to make the system more accessible to users. As discussed in the challenges section, we were unable to meet some of the requirements we set out to achieve regarding the data storage and retransmission on the nodes. This feature would have been very useful to prevent data loss in the outages we have had from the gateway not connecting to the TTN servers correctly and if we had known the full limitations of using TTN for decoding the LoRaWAN packets we may have decided against using their service.

Similarly, we had trouble with the microcontrollers we used for the project with our original choice not being viable for our application because of significantly higher

power consumption than the datasheet would suggest. Our original solution would have had the available pins to simplify our PCB as well as interface with a microSD card for local storage but because of the higher power consumption we were forced to change to a different microcontroller.

Though we encountered these issues in our project, we were still able to deliver a system to the Fairfield Osborn Preserve that can be used for the study they are currently planning on conducting as well as expansion in the future for use in other ecological studies.

References

[1] H. Xu, J. Feng, M. Ji, S. Fan, and W. Ji, "The seawater quality monitoring and data inconsistency processing system based on a long-range sensor network," *Journal of Coastal Research*, vol. 105, no. sp1, Dec. 2020. doi:10.2112/jcr-si105-046.1

[2] P. Dutta and A. Dunkels, "Operating Systems and network protocols for Wireless Sensor Networks," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 370, no. 1958, pp. 68–84, Jan. 2012. doi:10.1098/rsta.2011.0330

[3] J. Porter et al., "Wireless Sensor Networks for Ecology," OUP Academic, https://academic.oup.com/bioscience/article-abstract/55/7/561/306750 (accessed Sep. 29, 2024).

[4] Sownie, C. *The principles of ethical design (and how to use them) - 99designs*. Available at: https://99designs.com/blog/tips/ethical-design/ (Accessed: 16 December 2024).

[5] Lizin, S., Van Dael, M. and Van Passel, S. (2017) 'Battery pack recycling: Behaviour change interventions derived from an integrative theory of Planned Behaviour Study', *Resources, Conservation and Recycling*, 122, pp. 66–82. doi:10.1016/j.resconrec.2017.02.003.

Appendix

FT-1 Additional data

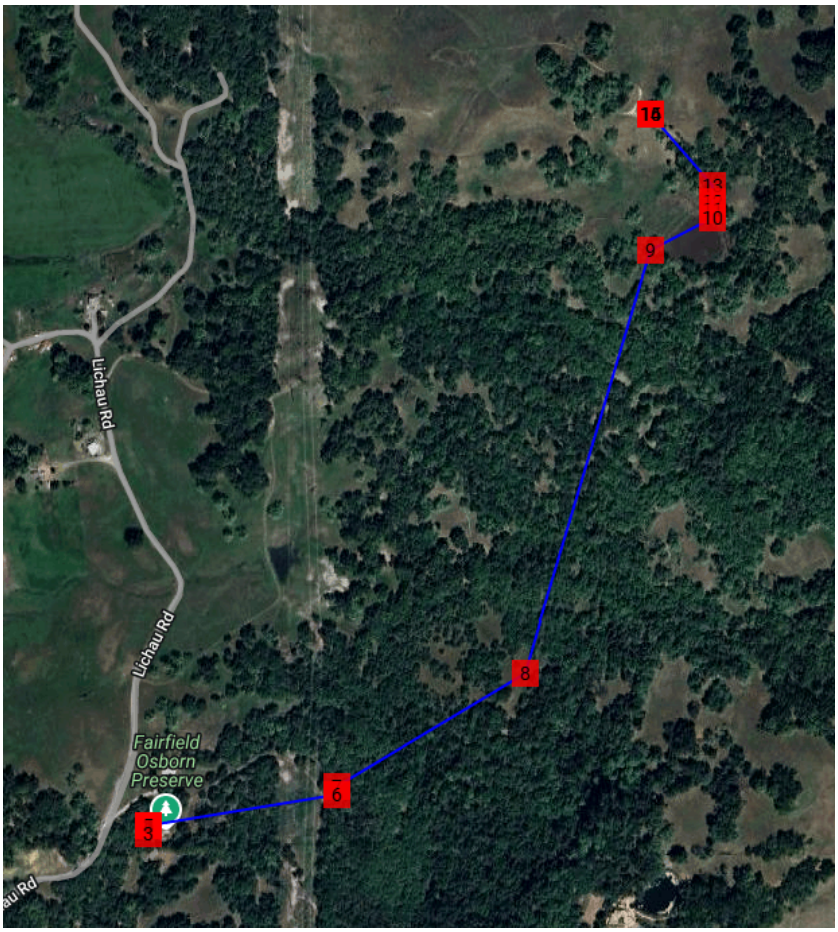


Figure 28: Large map with all data points

Table 9: Values for all map data points

Point	RSSI	SNR
1	-21	9.8
2	-48	9
3	-60	10.2
4	-12	9.5
5	-17	10.8
6	-115	-1.2
7	-101	8.5
8	-116	-4.5

9	**-134**	**-7.2**
10	**-124**	**-7.2**
11	**-120**	**-6.5**
12	**-113**	**-8**
13	**-112**	**-7**
14	**-113**	**-7.5**
15	**-110**	**2**
16	**-117**	**0.5**

FT-7 Additional Data:

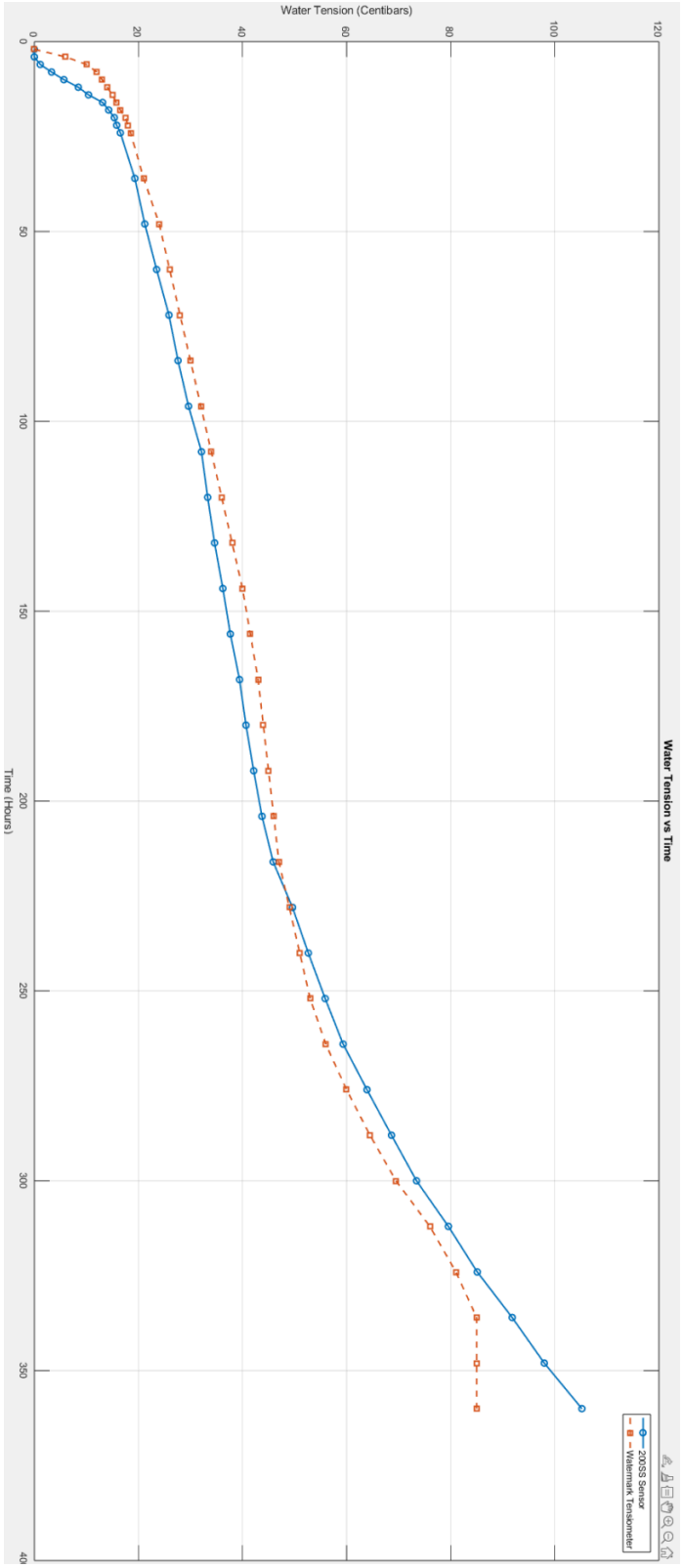


Figure 29: Larger Version of water tension graph

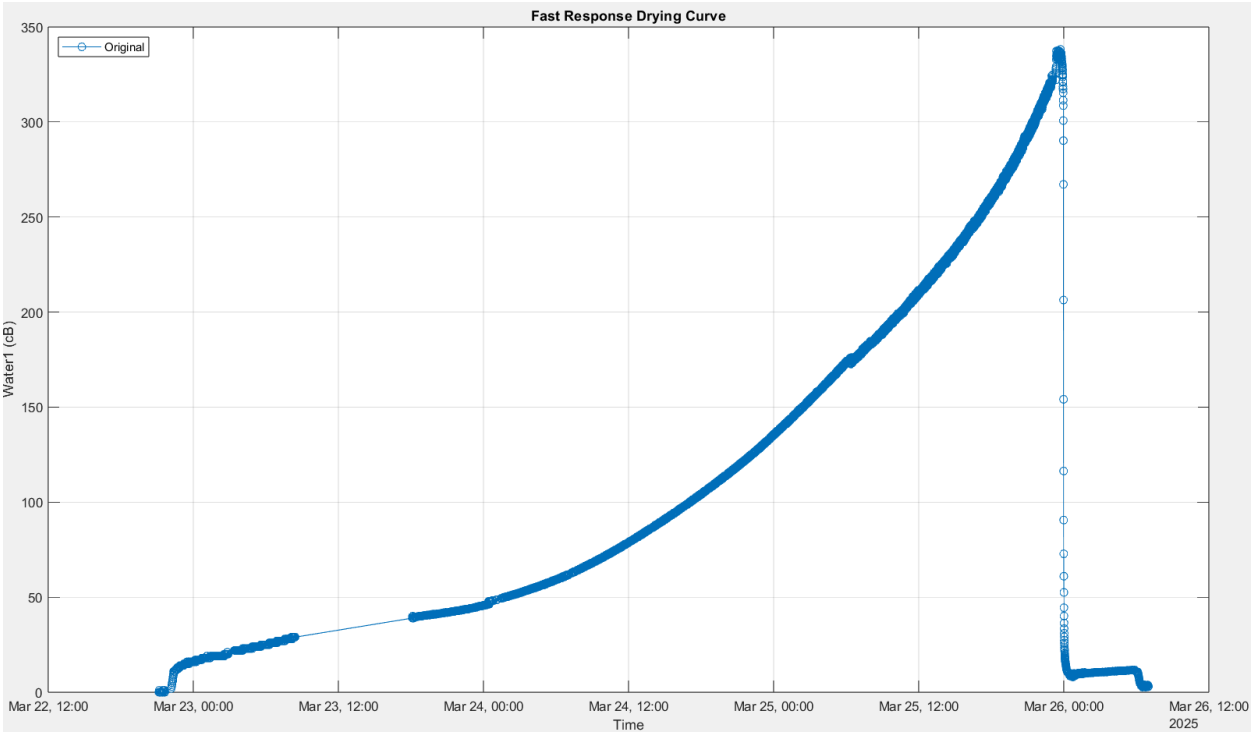


Figure 30: Fast Drying Curve

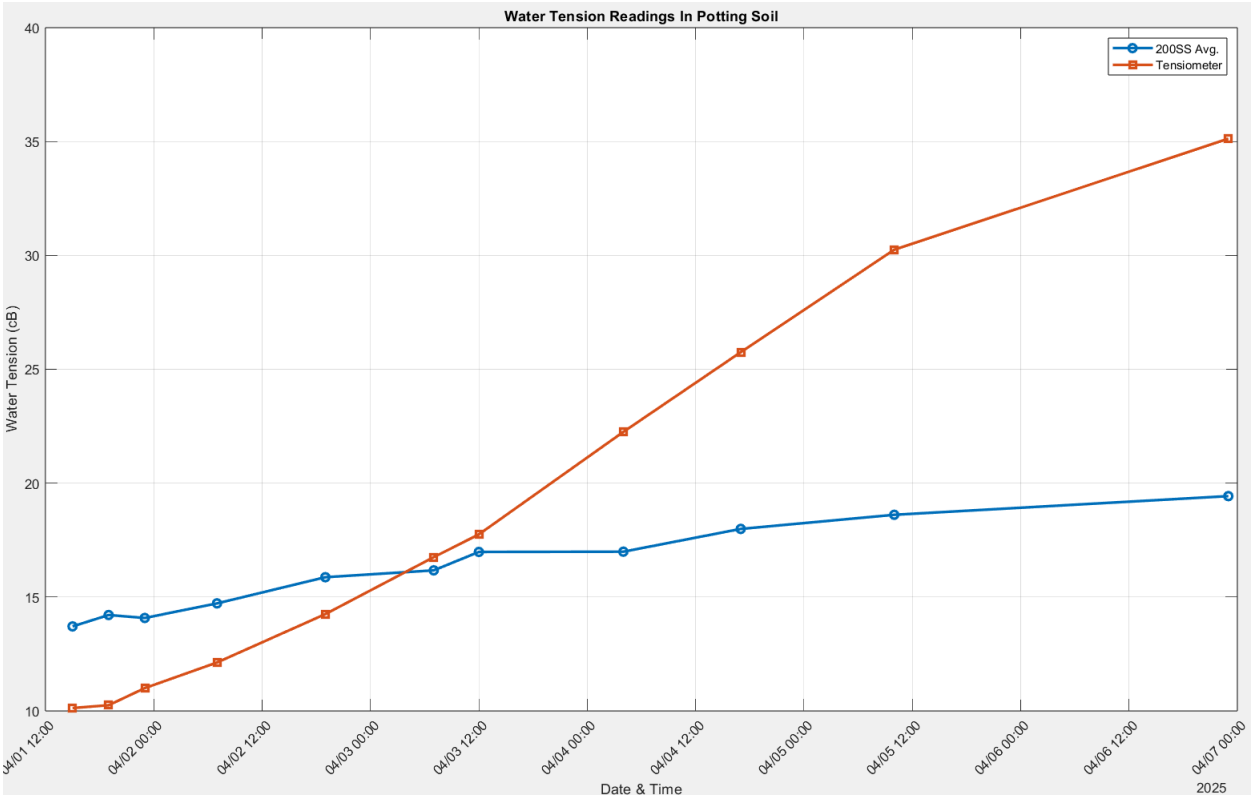


Figure 31: Readings of water tension across 5 days

FT-8 Additional data:

	DMM_mV	ADC_Low_mV	ADC_High_mV	Low_Error_mV	High_Error_mV
1	52.62	52.15	52.73	-0.48	0.10
2	72.63	72.25	73.13	-0.38	0.50
3	110.85	110.86	111.56	0.01	0.71
4	155.25	155.04	155.39	-0.21	0.14
5	204.00	204.32	205.20	0.32	1.20
6	249.57	249.43	250.55	-0.14	0.98
7	303.72	303.98	304.39	0.26	0.67
8	350.17	351.74	352.91	1.57	2.74
9	408.19	410.10	410.80	1.91	2.61
10	450.12	451.76	452.17	1.64	2.05
11	508.21	510.29	511.41	2.08	3.20
12	553.38	555.46	556.11	2.08	2.73
13	602.76	605.45	606.15	2.69	3.39
14	658.71	661.45	662.41	2.74	3.70
15	714.33	717.83	718.65	3.50	4.32
16	750.95	754.75	755.80	3.80	4.85
17	818.30	822.77	823.13	4.47	4.83
18	851.73	856.05	857.11	4.32	5.38
19	900.42	904.80	905.55	4.38	5.13
20	955.13	960.41	961.52	5.28	6.39
21	1005.30	1010.98	1011.68	5.68	6.38
22	1099.80	1105.96	1106.31	6.16	6.51
23	1200.30	1206.98	1207.56	6.68	7.26
24	1303.10	1310.51	1311.39	7.41	8.29
25	1404.40	1412.81	1413.53	8.41	9.13
26	1500.30	1508.59	1510.14	8.29	9.84
27	1605.50	1614.63	1615.61	9.13	10.11
28	1701.40	1691.95	1712.46	-9.45	11.06
29	1804.20	1816.41	1817.40	12.21	13.20
30	1909.10	1919.98	1920.88	10.88	11.78
31	2000.90	2012.34	2013.16	11.44	12.26
32	2101.60	2113.59	2114.47	11.99	12.87
33	2200.10	2212.79	2213.61	12.69	13.51
34	2302.60	2315.92	2317.08	13.32	14.48
35	2381.10	2394.38	2395.66	13.28	14.56

Uncalibrated

	DMM_mV	ADC_Low_mV	ADC_High_mV	Low_Error_mV	High_Error_mV
1	52.48	51.94	52.46	-0.55	-0.02
2	72.23	71.91	72.26	-0.32	0.03
3	109.53	108.54	109.20	-0.99	-0.33
4	151.13	150.26	150.48	-0.87	-0.65
5	202.84	202.00	202.22	-0.84	-0.62
6	247.38	246.29	246.99	-1.09	-0.39
7	298.00	297.14	297.32	-0.86	-0.68
8	350.93	350.10	350.30	-0.83	-0.63
9	397.81	397.16	397.32	-0.65	-0.49
10	463.67	462.88	463.07	-0.79	-0.60
11	504.15	503.79	504.19	-0.36	0.04
12	554.06	553.56	553.69	-0.50	-0.37
13	598.48	597.58	597.88	-0.90	-0.60
14	653.58	652.98	653.24	-0.60	-0.34
15	703.48	703.11	703.53	-0.37	0.05
16	749.92	749.24	749.50	-0.68	-0.42
17	802.25	801.83	802.14	-0.42	-0.11
18	855.45	854.74	855.54	-0.71	0.09
19	903.91	903.56	903.95	-0.35	0.04
20	957.95	957.75	958.23	-0.20	0.28
21	1001.10	1000.84	1001.24	-0.26	0.14
22	1101.20	1100.86	1101.54	-0.34	0.34
23	1207.50	1207.02	1207.59	-0.48	0.09
24	1310.10	1309.97	1310.42	-0.13	0.32
25	1410.40	1410.41	1410.97	0.01	0.57
26	1502.70	1502.49	1502.99	-0.21	0.29
27	1603.80	1603.90	1604.16	0.10	0.36
28	1700.50	1700.28	1701.09	-0.22	0.59
29	1803.30	1803.21	1803.83	-0.09	0.53
30	1902.90	1902.86	1903.21	-0.04	0.31
31	2004.80	2004.61	2005.27	-0.19	0.47
32	2099.60	2099.74	2100.16	0.14	0.56
33	2200.70	2199.99	2200.54	-0.71	-0.16
34	2311.90	2311.16	2311.84	-0.74	-0.06
35	2380.10	2379.88	2380.19	-0.22	0.09

Calibrated

MATLAB code used to analyze timestamp data

```
%% read data from file and select range
tableIn = readtable("data_export2.csv", 'DatetimeType', 'datetime');
data = tableIn(:,1:7);
data = sortrows(data, 'timestamp');

%cutoffTime = datetime('2025-04-13 12:00:00');
%data = data(data.timestamp >= cutoffTime, :);

subsetTables = cell(1,6);

for i = 1:6
    subsetTables{i} = data(data{:,2} == i, :);
end

Node1Data = subsetTables{1};
```

```
Node2Data = subsetTables{2};
Node3Data = subsetTables{3};
Node4Data = subsetTables{4};
Node5Data = subsetTables{5};
Node6Data = subsetTables{6};

Node1Delta = diff(Node1Data.timestamp);
Node2Delta = diff(Node2Data.timestamp);
Node3Delta = diff(Node3Data.timestamp);
Node4Delta = diff(Node4Data.timestamp);
Node5Delta = diff(Node5Data.timestamp);
Node6Delta = diff(Node6Data.timestamp);

%% calculate the percentage of missed values
missedData1 = sum(Node1Delta > minutes(16));
missedData2 = sum(Node2Delta > minutes(16));
missedData3 = sum(Node3Delta > minutes(16));
missedData4 = sum(Node4Delta > minutes(16));
missedData5 = sum(Node5Delta > minutes(16));
missedData6 = sum(Node6Delta > minutes(16));

%technically wrong because it undercounts by the number of misses.
%Ignoring this
loss1 = missedData1 / height(Node1Data);
loss2 = missedData2 / height(Node2Data);
loss3 = missedData3 / height(Node3Data);
loss4 = missedData4 / height(Node4Data);
loss5 = missedData5 / height(Node5Data);
loss6 = missedData6 / height(Node6Data);

%% Plot figures for timestamp differences

figure(1)
Node1Timestamp = Node1Data.timestamp;
Node1Delta = diff(Node1Timestamp);
plot(Node1Timestamp(2:end), minutes(Node1Delta))
title("Difference in measurement timestamps")
xlabel("time received")
ylabel("time since last value")

figure(2)
Node2Timestamp = Node2Data.timestamp;
Node2Delta = diff(Node2Timestamp);
plot(Node2Timestamp(2:end), minutes(Node2Delta))
title("Difference in measurement timestamps")
xlabel("time received")
ylabel("time since last value")

figure(3)
Node3Timestamp = Node3Data.timestamp;
Node3Delta = diff(Node3Timestamp);
plot(Node3Timestamp(2:end), minutes(Node3Delta))
title("Difference in measurement timestamps")
xlabel("time received")
ylabel("time since last value")
```

```
figure(4)
Node4Timestamp = Node4Data.timestamp;
Node4Delta = diff(Node4Timestamp);
plot(Node4Timestamp(2:end), minutes(Node4Delta))
title("Difference in measurement timestamps")
xlabel("time received")
ylabel("time since last value")

figure(5)
Node5Timestamp = Node5Data.timestamp;
Node5Delta = diff(Node5Timestamp);
plot(Node5Timestamp(2:end), minutes(Node5Delta))
title("Difference in measurement timestamps")
xlabel("time received")
ylabel("time since last value")

figure(6)
Node6Timestamp = Node6Data.timestamp;
Node6Delta = diff(Node6Timestamp);
plot(Node6Timestamp(2:end), minutes(Node6Delta))
title("Difference in measurement timestamps")
xlabel("time received")
ylabel("time since last value")

%% find mean and std dev of timestamp differences

% Calculate time differences in seconds
dtSeconds1 = seconds(Node1Delta);  % Convert to seconds
dtSeconds2 = seconds(Node2Delta);
dtSeconds3 = seconds(Node3Delta);
dtSeconds4 = seconds(Node4Delta);
dtSeconds5 = seconds(Node5Delta);
dtSeconds6 = seconds(Node6Delta);

% Filter for time differences between 14 and 16 minutes (now in seconds)
Node1TimestampsFiltered = dtSeconds1(dtSeconds1 >= 14 * 60 & dtSeconds1
< 16 * 60);
Node2TimestampsFiltered = dtSeconds2(dtSeconds2 >= 14 * 60 & dtSeconds2
< 16 * 60);
Node3TimestampsFiltered = dtSeconds3(dtSeconds3 >= 14 * 60 & dtSeconds3
< 16 * 60);
Node4TimestampsFiltered = dtSeconds4(dtSeconds4 >= 14 * 60 & dtSeconds4
< 16 * 60);
Node5TimestampsFiltered = dtSeconds5(dtSeconds5 >= 14 * 60 & dtSeconds5
< 16 * 60);
Node6TimestampsFiltered = dtSeconds6(dtSeconds6 >= 14 * 60 & dtSeconds6
< 16 * 60);

% Calculate average and standard deviation in seconds
avgDelta1 = mean(Node1TimestampsFiltered);
stdDev1 = std(Node1TimestampsFiltered);
avgDelta2 = mean(Node2TimestampsFiltered);
stdDev2 = std(Node2TimestampsFiltered);
avgDelta3 = mean(Node3TimestampsFiltered);
stdDev3 = std(Node3TimestampsFiltered);
avgDelta4 = mean(Node4TimestampsFiltered);
```

```
stdDev4 = std(Node4TimestampsFiltered);
avgDelta5 = mean(Node5TimestampsFiltered);
stdDev5 = std(Node5TimestampsFiltered);
avgDelta6 = mean(Node6TimestampsFiltered);
stdDev6 = std(Node6TimestampsFiltered);

% Optionally, display results in seconds
fprintf('Node 1: Avg = %.2f seconds, Std Dev = %.2f seconds\n',
avgDelta1, stdDev1);
fprintf('Node 2: Avg = %.2f seconds, Std Dev = %.2f seconds\n',
avgDelta2, stdDev2);
fprintf('Node 3: Avg = %.2f seconds, Std Dev = %.2f seconds\n',
avgDelta3, stdDev3);
fprintf('Node 4: Avg = %.2f seconds, Std Dev = %.2f seconds\n',
avgDelta4, stdDev4);
fprintf('Node 5: Avg = %.2f seconds, Std Dev = %.2f seconds\n',
avgDelta5, stdDev5);
fprintf('Node 6: Avg = %.2f seconds, Std Dev = %.2f seconds\n',
avgDelta6, stdDev6);

%% Make Table
% Create vectors for the table
Node = (1:6)';
AvgTime = [avgDelta1; avgDelta2; avgDelta3; avgDelta4; avgDelta5;
avgDelta6];
StdDevTime = [stdDev1; stdDev2; stdDev3; stdDev4; stdDev5; stdDev6];

% Create the table
timeStatsTable = table(Node, AvgTime, StdDevTime);

% Display the table
disp(timeStatsTable);

writetable(timeStatsTable, 'timeStats.xlsx');
fprintf('Table written to "timeStats.xlsx"\n');
```