ENG EC 463 – Team 27

Smart Agriculture Sensor Network

First Semester Report

Benjamin Livney, Maxine Loebs, Sergio Pareja, Emanuel Perez and Noah Spahn

**Executive Summary (M.L.)**—Farming is paramount to society. Throughout history, as global populations have increased, farming methods have had to improve to keep up with growing demand. Today, the situation is no different – with more and more mouths to feed, farmers are looking to modernize their practice and adopt technologies that can help them boost crop yields, reduce waste, save money, and stay afloat among industrial-scale agricultural corporations. To address this need, we will be developing a fully-functional agricultural sensor network, which reports data about specific locations on the farm back to the user in the comfort of their farmhouse. We have come up with a detailed plan to realize our design. The backbone of our final product is our sensor nodes. These will contain our selected sensors, a microcontroller equipped with a radio transceiver, and the required hardware to supply power to all of the electronics inside. All of the components will be put into a weatherproof enclosure. All of these nodes will be connected together in a scalable mesh network. The data from all of the nodes will be pooled into a single gateway node, which will then send compiled data to our web console. Our web console will present this live data in a neat, readable format. There are other agricultural sensor networks on the market, but our product will include a number of innovative features that make it stand out, such as energy harvesting, and the use of a wide variety of sensors to paint a full picture of many conditions out in the field.

—————————— ◆ ——————————

# 1 Introduction (M.L.)

In recent years, farmers have been embracing technology to supplement and/or replace more traditional farming methods in order to increase crop yields, lower the cost of production, reduce waste, and – in the case of smaller, local farmers – remain profitable and competitive in a market that is dominated by large-scale agricultural corporations.

As such, we are being tasked with creating an agricultural sensor network to help accomplish all of these things. The network will allow the farmer to be aware of the conditions at any location on his farm at which an individual sensor node is placed, all without having to actually step foot out into the field, as the data will be available to view on a web console. This saves time and money. The farmer doesn’t have to drive around the farm himself to check on things, or pay people to do the same. Some farms – even those owned by small farmers – are massive in size, so checking up on areas of the farm could become quite time-consuming, even when done only once a day. Having insight into the current conditions around the farm additionally allows the farmer to save money by being able to better distribute resources. For example, the device will have a soil moisture sensor. By monitoring soil moisture, the farmer can in turn adjust how much he is watering different areas of his fields, preventing overwatering and in hand saving money. With knowledge of field conditions, the farmer can make adjustments to ensure ideal conditions for achieving the most successful crop, and thus achieve higher profits.

We have come up with a detailed plan for the implementation of our design. Our network will consist of sensor “nodes” which will collect and relay data back to a single gateway node. These sensor nodes will, as the name would suggest, contain all of our sensors. We are planning to gather data pertaining to air temperature, humidity, barometric pressure, ambient light, soil temperature, and soil moisture. The data readings give the farmer a full picture of their crop’s current environment. Since the information needs to get back to the farmer, each node is also equipped with a transceiver. We will set up a mesh network protocol that will allow information to bounce from node to node until it gets back to the gateway. If one node in the chain is not able to relay data, the mesh networking protocol will allow for a new route to be discovered. Of course, both the sensors and the microcontroller need to be powered. We will use both a battery pack and a solar panel to provide power. The solar panel can also recharge the batteries. The electronics will be housed in a weatherproof container, which will protect our electronics from rainfall, windblown dust and dirt, and the formation of ice in colder months. The gateway node, which will be located in the farmhouse, is where all the data will be pooled from all of the sensor nodes out in the field. In addition to a transceiver, this node will also have a co-processor that will allow data transmission over WiFi to our web console. Our web console, upon receiving live data, will display the data in graphs that will allow the user to see changes in conditions over time. This is a powerful tool, since seeing trends of data, rather than just individual points in time, further allows the farmer to make predictions and adjustments in his plan to achieve the best crop.

There are other products on the market that seek to accomplish the same goal as our prototype: to provide farmers data about their farms via a network of sensors. We are including a number of features that make our product different. We will be using a number of different sensors, mentioned previously, that will provide the farmer with a multitude of data pertaining to both atmospheric and soil conditions. Many companies only produce nodes that report back one type of data reading, or require users to purchase several different models to get a wider variety of data. Additionally, with the use of energy harvesting, we are greatly able to extend the battery life of our nodes. In fact, our nodes will be able to last at least one year between charges. Many companies rely solely on battery power for their nodes, so they need to be recharged much more frequently, creating more work for the farmer.

# 2 Concept Development (M.L.)

To develop our concept for design, we identified key functions to be fulfilled by our prototype, and then came up with solutions to address each function.

Our sensors must operate as a network, and therefore communicate with each other, and not just with the gateway node. As such, we decided to use a mesh networking protocol. This has a number of benefits. Broken or unresponsive nodes can be identified, and since signals bounce from node to node in a mesh back to the gateway, a new route can be discovered to avoid the broken node until it is repaired, so that all other data from working nodes can still make its way back to the user. Since we are not sending data directly back to the gateway, but rather bouncing the signal back via other nodes, we also do not need a massive signal transmission range.

With the basis of our network established, we had to choose how our data would be transmitted. Eventually, we decided to use LoRa (Long Range) RF transmission-equipped microcontrollers. The microcontrollers, with the proper antennas, are able to transmit data across several kilometers while using very little power. We have set our node-to-node transmission range goal to be between 750 m and 5 km. We had considered some other methods of data transmission, such as equipping our nodes with SIM cards to connect to a cellular network, but we soon decided that this would become expensive, complicated, and use too much power.

Our nodes must be able to last at least one year between charges. This means that we had to carefully consider what devices we would use and how much power they would draw, and what we would use to provide power. At first, we considered just equipping the nodes with a massive battery pack, but we decided that this would become too bulky if they had to last an entire year, and that we were capable of something more innovative. It was suggested that we look into energy harvesting. We looked into both solar and thermoelectric energy harvesting. Ultimately, solar energy harvesting with a solar panel was the clear winner, as we were able to propose an implementation that would provide a much greater amount of power. So, our final powering solution consists of a combination of battery pack and a solar panel. The solar panel can recharge the batteries and supply power directly to the electronics inside the node as well.

The information that we are sending back to the user must be useful for farming. We carefully considered the sensors that we would be using in our prototype – air temperature, humidity, barometric pressure, ambient light, soil temperature, and soil moisture – making sure that the information they provided could be used to gain insight into how to achieve the best crop. We also considered including a sensor to measure soil pH, but these kinds of sensors, however, require relatively involved and individual calibration.

Lastly, we needed to make sure that our sensor nodes are weather-resistant. Water and dust should not be able to get inside of our node and damage our electronics. We researched different weathe-rproofing standards, and determined to aim for at least IP43 compliance, which would protect our electronics from spraying water.

We have tabulated a formal list of engineering requirements, that can be quantifiably measured, in Appendix 1.

# 3 System Description (B.L.)

As a solution, a mesh sensor network utilizing the LoRa communication protocol will be developed. The network will consist of many sensor nodes outfitted with relevant sensors, energy harvesting equipment and electronics, and a LoRa RF-equipped microcontroller. The network will also have a gateway node to collect data from the sensor nodes and push it to a web console to be viewed by the user.

Each sensor node will be outfitted with a solar energy harvesting system incorporating a small photovoltaic (PV) module and power electronics to maintain power for at least one year. Nodes will be equipped with a 3W 6.5V PV module capable of providing a maximum of 460mA in high light conditions. In order to not drive the module too hard, a maximum battery charging current of 400mA is set by the battery charge IC, the BQ24074. For low light conditions, a low-power boost converter, the MCP16414, will boost voltages above 1V up to 5V to operate the BQ24074 correctly and to power the microcontroller and sensors. The boost converter will have both a manual and an automatic bypass to ensure use only in low-light conditions to maximize efficiency, and to ensure the maximum input voltage to the boost converter of 5.25V is not exceeded. The battery pack will be between 12000mAh and 18000mAh, which will ensure sufficient power for three weeks with no sun, assuming an average current draw of 25mA from the microcontroller and sensors. This large backup battery will help meet the power budget requirements in conjunction with the solar energy harvesting system.

The sensor nodes will feature both digital and analog sensors capturing data to represent a variety of factors that affect growing conditions. The sensors will consist of: the BME280 to measure temperature, humidity, and barometric pressure, the VEML7700 to measure ambient light intensity, the VGH400 to measure soil moisture, and an NTC thermistor to measure soil temperature. Monitoring these quantities are basic necessities in order to create optimal growing conditions. For more specific applications, additional digital and analog sensors can easily be added to reflect the needs of a specific location. Additional sensors could include UV index sensors for UV-specific crops and motion sensors for wildlife tracking.

The LoRa communication protocol allows for wireless communication and mesh networking between LoRa-enabled devices using a tunable 900MHz carrier signal without the need for cell service, Wi-Fi, or Bluetooth capabilities. This protocol will be used to create the mesh sensor network by establishing a gateway node that sends poll requests and waits for responses. As shown in Figure 1, each sensor node will listen for incoming LoRa messages, either from the gateway node or from other sensor nodes. If the incoming message is not a poll request, meaning the message contains sensor readings, the sensor node will use the established mesh protocol using LoRa to repeat that message, sending it closer to the gateway node for analysis. If the incoming message is a poll request, the sensor node checks if it is polling itself, in which case it takes sensor readings and sends them to the mesh network. Otherwise, it repeats the poll request so that other nodes out of range of the gateway can receive it.

The gateway node will, at regular intervals, send poll requests to the mesh network for data from specific sensor nodes. As described above, the sensor nodes will echo the poll request until it reaches its destination, at which point the data will be sent and echoed back. If the gateway node receives a response before a timeout, it will push it to the web console for processing using an ESP8266 Wi-Fi enabled module. If no response is received before the timeout, the gateway node will push a notification to check the status of the requested node. Since the network should be installed such that every node is in range of multiple other nodes, there will be many paths from the gateway to any requested node. Thus, if a timeout occurs, the gateway will know that the requested node is at fault and not some other sensor node.

Once sensor readings from sensor nodes have reached the gateway node, the data will be processed by ThingSpeak IoT, a free IoT data visualization tool running off of MATLAB. This tool can create live-updating graphs using MATLAB plotting functions that run as embedded HTML divs. The gateway node, running inside the farmhouse, will have access to Wi-Fi, and thus will send sensor readings to ThingSpeak IoT, which will store, plot, and update the data live in an HTML webpage for the user to see. Data will not be pushed and stored in a JSON object, which can be read by ThingSpeak IoT.

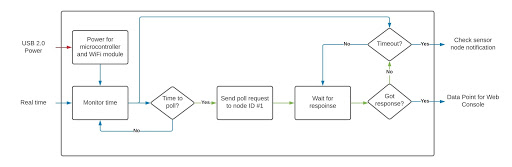


Fig. 2. A functional analysis of the gateway node describing poll requests, responses, and interaction with the web console.

# 4 First Semester Progress

## 4.1 Current Network Protocol (E.P.)

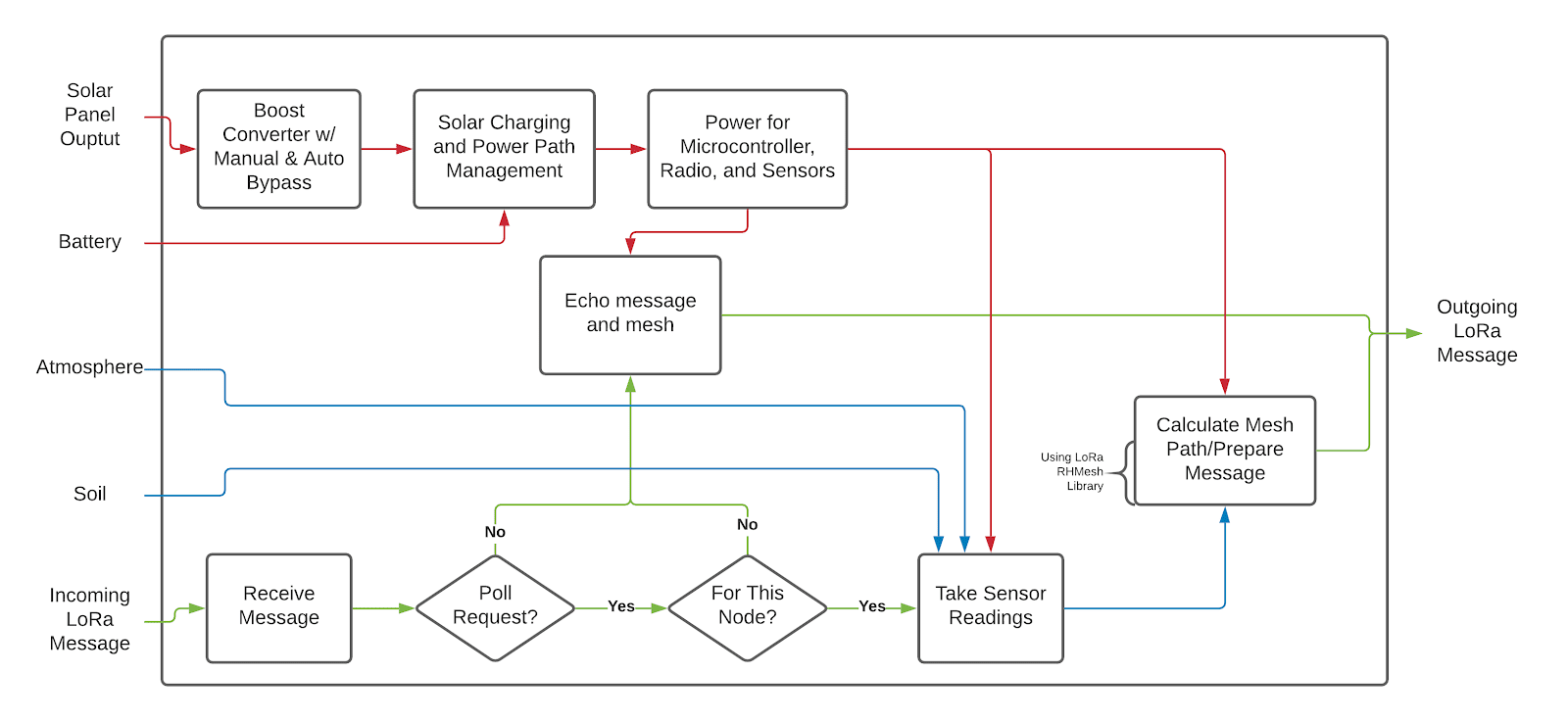


Fig. 1. A functional analysis of a sensor node describing power flow, sensor operation, and LoRa mesh networking.

During the first prototype testing, we used a different network protocol than the mesh network to test the communication between our nodes. Similar to the mesh network, the network consists of a mother node (gateway node) and outer nodes (sensor nodes). The mother node sends poll requests for a specific node to the network. With the use of the LoRa microcontroller, the poll is sent via radio waves to nearby listening nodes. The receiving nodes will send an acknowledgement depending on whether or not it’s the node the mother node is looking for. If it is, the node will send the current sensor data back. If it isn't, it will send a simple acknowledgement and send the poll to other nearby nodes.

## 4.2 Power Management (E.P.)

To meet the client’s battery life expectancy of one year, the device has the ability to sleep/wake for an X amount of time. During the device’s sleep cycle, all electronics and internal functions are stopped. Once X amount of time passes, the device is woken up via a watchdog timer. If the desired amount of sleep is greater than the watchdog timer’s limit, then the device wakes up momentarily to sleep again till it reaches the desired amount. The current sleeping protocol can save up 40% of energy consumption than just the device being idle.

## 4.3 Power Electronics (B.L.)

For the first prototype test, we demonstrated the power path management abilities of the proposed system. We were able to charge a Li-Ion battery through the power path management IC, deliver power to a load from an external power supply through the IC, and power the same load from the battery through the same IC. From previous testing, it was shown that the battery could be charged from the solar panel with up to 160mA on a partly-cloudy New England winter day, and with up to 50mA while obstructing the solar panel, simulating shadows. This IC was then integrated into a PCB, also including low-light power adjustment with a boost converter, as well as the microcontroller and sensors. Schematics for the power electronics and sensor electronics are shown in Appendix 3.

## 4.4 Power Electronics (B.L.)

Using the I2C communication protocol, we were able to demonstrate data collection from multiple digital and analog sensors at once, and the formatting of data points into a string to be sent by the LoRa module. This was accomplished with the use of sensor breakout modules, inexpensive PCB sensor products meant for modular testing and projects that do not feature best practice layout techniques: ground between the data lines to reduce crosstalk and star grounding with separate ground planes for analog and digital sensors. The PCB we designed does feature those best practices, which will serve to improve reliability and precision in data collection. Schematics for all the electronics, as well as a preliminary PCB layout, are shown in appendix 3.

## 4.5 Enclosure (M.L.)

Our chosen enclosure, the Hammond 1554WA2GYCL, goes beyond the weather protection of our initially-set IP43 requirement, which provides protection against spraying water. It is a 7.1” x 7.1” x 3.54” polycarbonate box with a removable clear lid. It has a NEMA 4X rating, meaning that it is watertight, protects against dirt, dust, and the formation of ice, and is corrosion resistant. The size of this enclosure leaves us with ample room for our PCB, with all attached components, and our battery pack. The clear lid allows us to take readings of ambient light. To get accurate sensor readings of humidity and barometric pressure, two Stego 284 vent plugs were installed on opposite sides of the enclosure, 2” from the side and 0.75” from the top. These allow air into the enclosure and equalize the pressure inside and outside of the enclosure, while maintaining the NEMA 4X rating. Pressure equalization has the added benefit of protecting from condensation building up inside the enclosure with changes in temperature. Next steps will include making more through-holes for sensors and possibly the antenna, and then making sure they are sealed properly. A picture of the enclosure is shown in Appendix 3.

## 4.6 Current Web Application (N.S.)

In order to demonstrate the WebApp visualization portion of the client’s needs, we utilize a Matlab analysis script, and the ThingSpeak IoT tool to upload data and analyze the uploaded data into a visualization. We were able to upload data through a microcontroller WiFi shield simulator as well as in a local Matlab script. Testing determined that there is a minimum of 10 seconds between uploads. The frequency of data collection will not be near this fast. The visualization is done through ThingSpeak with a server-based Matlab script. During our test we demonstrated that the visualization, embedded into the console webpage, live updates to display exact data and timestamp the data pushed from the Matlab script.

## 4.7 Power Supply (N.S.)

The power supply was developed from four High Drain Samsung 30Q, 3000mAh 3.7V cells. This considered the power needs of the nodes as well as the available recapture rate observed in testing. For testing, our battery is a modular design connected with nickel plates and a PLA linked enclosure. This has the added advantage that additional cells can be added in later if needed. The battery was connected to the solar charger and demonstrated a successful charging process.

# 5 Technical Plan (N.S.)

The spring semester will mostly involve finalizing the hardware, the mesh network, and the gateway and console for the project. The fall semester has produced a variety of concepts and preliminary designs, so the spring will be focused on implementing them. The electronics need to be manufactured, assembled, and tested, the mesh network needs to be created and implemented, and the gateway node and console need to be able to communicate with each other and with sensor nodes. See the Gantt chart in Appendix 2 for more details about the scheduling of the remaining tasks.

## 5.1 Task I: PCB

**Task I.I: Finalize PCB design**.

The current PCB design will be refined and reworked to meet the capabilities of the manufacturer, JLCPCB. Their capabilities are listed on their website and should be implemented as design rules in Altium. The Altium design rule check tool will evaluate the design until there are no design rule violations, at which point the PCB will be ready to be manufactured. Lead: Ben; Assisting: Maxine.

**Task I.II: Assemble PCBs.**

Once PCBs are ordered and delivered, they will be tested for continuity and shorts. Once the boards’ manufacturing has been verified, they will be assembled according to the schematic, and will have their individual subsystems tested. This will be accomplished by first powering the Featherboard through USB to test the sensors, and then by connecting the solar panel to measure the current draw and charging current while operating from solar and battery power. Lead: Ben; Assisting: Maxine.

**Task I.III: Integrate software and hardware.**

The sensor systems and the power systems will be integrated with the microcontroller for data collection and power management. Mesh networking is not part of this task. Code will be written to operate the automatic boost converter bypass, and the function will be tested by using a variable lab-bench power supply to verify the over-voltage detection and protection. Code that reads data from sensors will be modified to accommodate new sensors and to improve formatting for increased RF bandwidth efficiency. The sensors will be tested to meet the specifications for precision. Lead: Ben; Assisting: Emanuel.

## 5.2 Task II: Console

**Task II.I: Uploading from Console**

An Arduino scripted console node will be made from a duplicate Featherboard and a WiFi shield enabling network connection. Code will be written to estable a private connection between the data channels on the server and the data queued in the console. The test shall be conducted from the device without user input as the console will not have user input to update the data sets. The design should be tested with data gathered from testing sensor nodes. Lead: Noah; Assisting: Maxine

**Task II.II: Generate Node Comparison Controls**

A Matlab visualization will be created for each data point at each sensor node to track its change over time independently. Controls shall be scripted to select and use Javascript to compare data channels across multiple nodes by generating the requested combination of data channels into a Matlab visualisation. The test shall be deemed successful if the webpage can correctly show a dynamic range of comparisons among both the same node and multiples of nodes. Lead: Noah

**Task II.III: Integrate Console Pages into GitHub**

All current HTML pages will be integrated within the plants-are-neat.github.io page. This shall be tested by testing nodes are live updating and that controls across pages as well as across node data channels still provide identical output whether they are based on the server or on a local client. This will enable both the source files for the design and web application to be present and viewable in the same location. Lead: Noah; Assisting: Maxine

**Task II.IV: Alerts for Data Bounds**

Controls will be scripted to allow the user to set bounds for any visualization that alerts the user if there is a threshold break off the bound. This will enable data on the console to not only be utilizable when viewing the page but away. The alerts will be scripted in Javascript and communication to the user sent in the form of an email. They will display what bound has been crossed and at what time. The test shall be conducted by producing a bound as well as data that goes through the bound. If the test is successful we should get communication reflecting the bound crossed and time it was crossed via email. Lead: Noah

**Task II.V: Alerts for Data Breaks**

Alerts will be scripted to alert the user if there is a break in the network. This will enable data on the console to alert the user if a channel isn’t updating so they can inspect the node. The alerts shall be scripted in Javascript and communication to the user sent in the form of an email. They will display what node has been lost and at what time it was last heard from. The test shall be conducted by producing a time signature break in the mesh and allowing the console to receive enough data that this node must no longer be communicating. Lead: Noah; Assisting: Sergio, Maxine

## 5.2 Task III: Mesh Network

**Task III.I: Unique identification**

For every single node in the field, there should be a single ID associated with that particular node. Upon failure from a node and having to restart, it is expected that it will have a protocol in place to reacquire its ID. To be done using past routing tables of nearby nodes and ISSR values to determine. No ID can be shared. Leads: Sergio and Maxine

**Task III.II: Self Healing**

The network should be able to function in the case of an error of a single node. Meaning that while the node is down, messages will be routed from a different route to reach its destination. Once the node comes back online, the node is seamlessly reconnected to the network and proceeds with its tasks. Leads: Sergio and Maxine

**Task III.III: Message Forwarding**

It is impossible for most nodes to be able to reach the gateway node. To this end, messages need to be forwarded to nodes in between the destination and the source. Protocol dictates that the node receiving the message has a routing table with the destination of the message in order to send it to the most appropriate node. Leads: Sergio and Maxine

**Task III.IV: Routing Table**

A routing table is a table meant to store the most efficient routes between this node and others. This is done so that the nodes don't have the memory burden of having to have the entire network stored in memory. The routing table needs to update in order to keep an up-to-date list with best routes to take into consideration things like nodes failing and weather conditions. Leads: Sergio and Maxine

**Task III.V: Data Reporting**

This task is to ensure that the individual nodes are able to report their findings back to the gateway node. Leads: Sergio and Maxine

**Task III. VI: Sleep Scheduling**

Designing a schedule based on internal microcontroller clocks, signal sending, or job completion. Decides when the device will go to sleep, when it will wake up, and for how long it stays awake. The main test will be a self-sustaining schedule of sleeping, waking, sending information, receiving information, and sleeping again in sync with other nodes, especially the mother node. Completed once the schedule achieves three successful and consecutive sleep-wake cycles. Lead: Emanuel; Assisting: Sergio, Maxine

# 6 Budget (B.L.)

The budget for this project’s R&D phase is $1000, with a grant of $500 to be provided by the department. The estimated budget for the project comes in at just under the grant amount, at $450.50. A detailed budget is shown below.

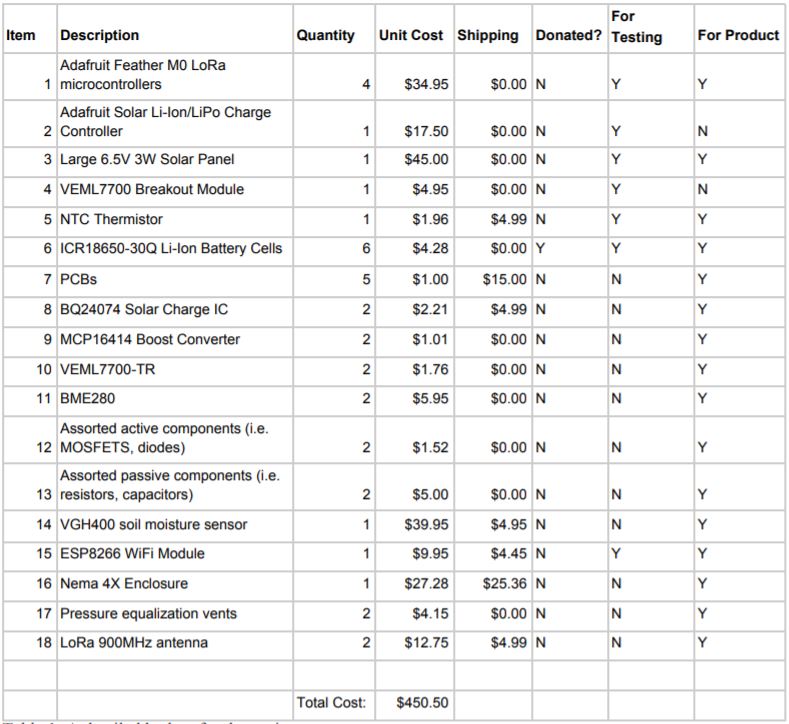


Fig. 3. A detailed budget for the project.

Many of the components used for testing can be repurposed for use in the final product to save money. However, there are many components needed for the PCB for example that have not been ordered yet. All of the components, already purchased or not, are shown in the table above. Items 5, 8-13 are tabulated more specifically in appendix 3.

# 7 Appendix I – Engineering Requirements (B.L.)

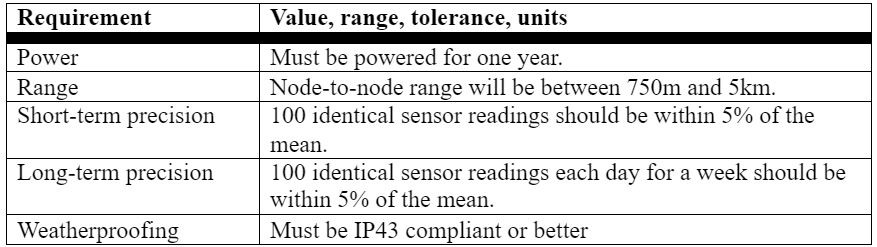


Fig. 4. A list of the engineering requirements for this project.

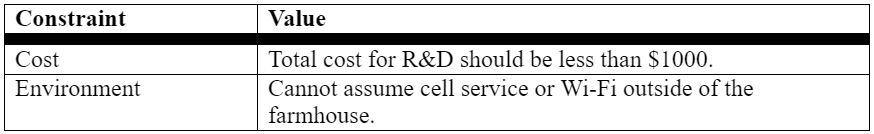
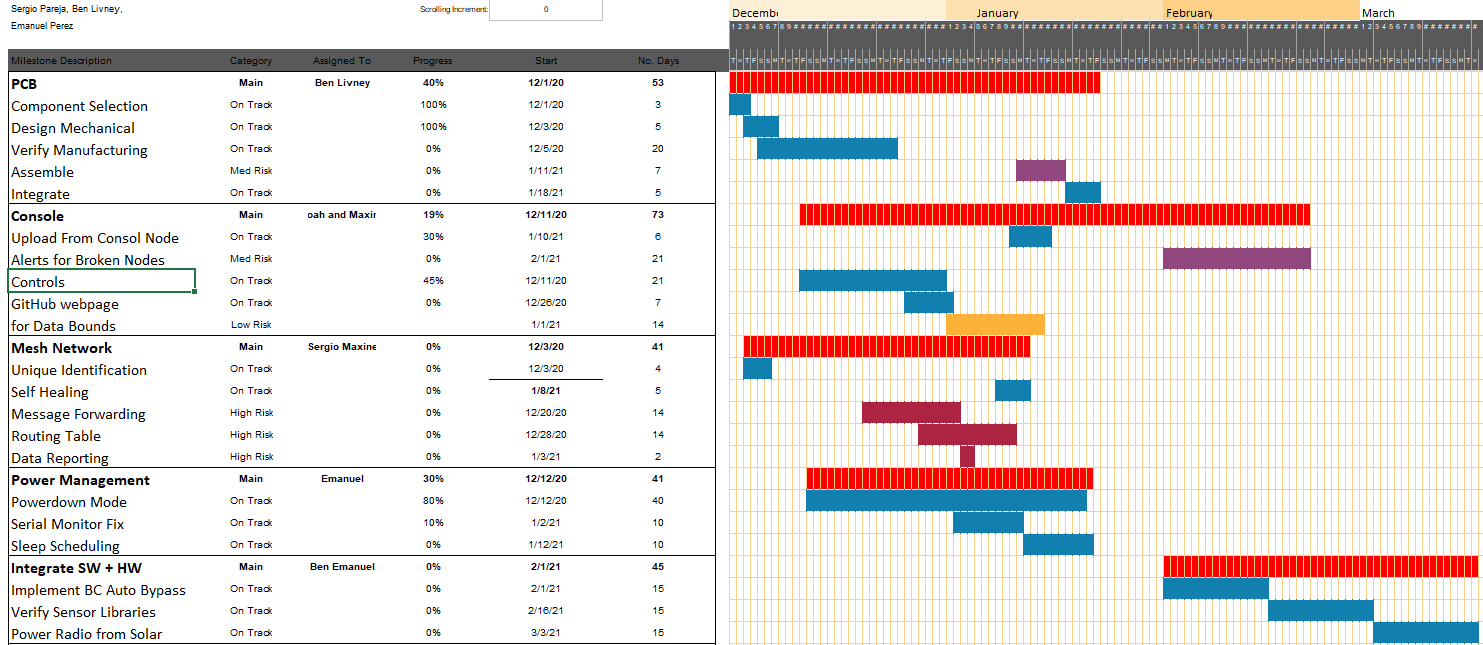


Fig. 5. A list of engineering constraints for this project.

# 8 Appendix II – Gantt Chart (S.P.)



# 9 Appendix III – Additional Figures, References, Team Info (E.P.)

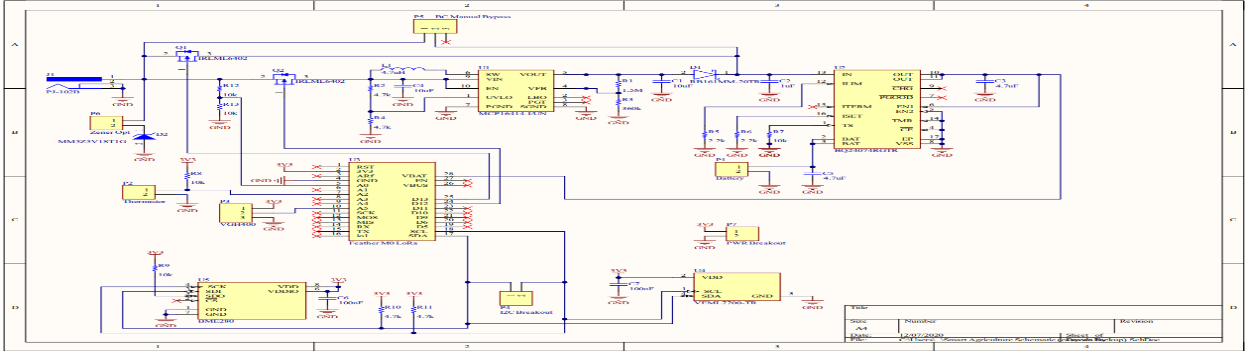


Fig. 6. Schematics for the power electronics and sensor electronics

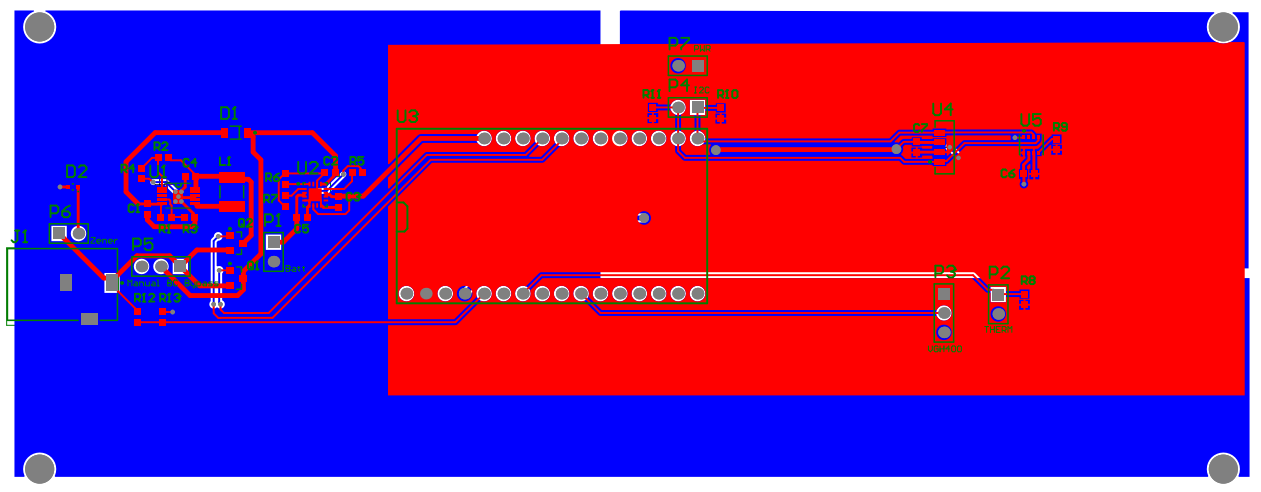


Fig. 7. preliminary PCB layout.

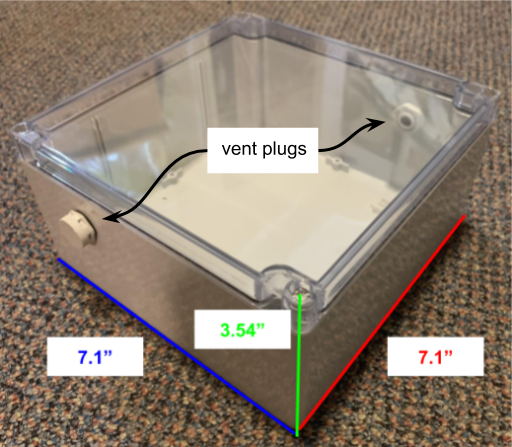


Fig. 8. enclosure design



Fig. 9. Budget Specifics

## 9.1 References

[1] “Almost 40 Percent of Worldwide Crops Lost to Diseases,” The Crop Site. [Online]. Available: https://thecropsite.com/articles/1202/almost-40-per-cent-of-worldwide-crops-lost-to-diseases/.

[2] "Sleeping Arduino - Part 1," Arduino, Zigbee and Embedded Development. [Online]. Available: http://donalmorrissey.blogspot.com/2010/04/putting-arduino-diecimila-to-sleep-part.html

[3] “Greenhouse Guardian: Environmental monitoring of greenhouse crops,” Bosch.IO, 17-Feb-2020. [Online]. Available: https://bosch.io/products/greenhouse-guardian/.

[4] “Homepage,” Sensoterra, 12-Oct-1970. [Online]. Available: https://www.sensoterra.com/.

[5] H. Solidpixels., “We bring innovative solutions to agriculture,” CleverFarm. [Online]. Available: https://www.cleverfarm.ag/.

## 9.2 Team Info

Ben Livney

845-642-0837

blivney@bu.edu

Maxine Loebs

631-702-5813

mloebs@bu.edu

Sergio Pareja

787-587-7420

sergio98@bu.edu

Emanuel Perez

832-267-8474

emperez@bu.edu

Noah Spahn

617-970-8255

nspahn@bu.edu

**History of Team and Company:**

The team started in mid September. We wanted to help one of society’s most important sectors, agriculture. A testament of our core values is in our name, Plants are Neat. Our lead market strategist, Ben, wanted our company’s name to follow our values of down to earth customer service while using simple solutions to solve modern problems. The goal of our company is to give farmers a reliable, economical way to survey their farmers without needing to traverse a huge landmass.