**AgriTrak - Irrigation System Metrics**

Adam Benjamin, Matt Boubin, Mary Floren, Ashley Youngwirth

Dr. Scott, Dr. Rajasekhar

2016-2017

**Table of Contents**

1. Problem Definition………………………………………………………….3
2. Project Background………………………………………………………....3
3. Project Research……………………………………………………………4
4. Solution Process……………………………………………………………7
5. Results……………………………………………………………………...13
6. Future Plans………………………………………………………………...17
7. Conclusion…………………………………………………………………17
8. References………………………………………………………………….19

**Problem Definition**

Our project is centered around collecting relevant farm metrics from the Institute of Food at Miami University (“The Farm”) and displaying and storing this data in a way that improves the farm manager’s, ability to efficiently and effectively grow crops.

The overarching goal of this project is to create a general solution that can be implemented beyond just the scope of the Institute of Food. Our solution needs to be applicable to many other use cases including urban and small-scale farming around the world.

**Project Background**

Our team consulted our a farm manager in order to gain a stronger understanding of what metrics are important to farmers and how we could create a product that would be most beneficial to anyone trying to grow food. We discovered two main metrics that we could display to help our potential customers.

1. It is important to farmers to know if the ground level temperature had dropped below freezing, and if so, for how long it had dropped below freezing. This information would help farmers make more proactive decisions about crop management and planting crops in the early spring.
2. It is important for farmers to have an understanding of the evapotranspiration of different plants on the farm in order to make more informed decisions about irrigation on a given day.

After obtaining this information, it became apparent to our team that it is important for us to display live data to farmers and also store past data in a database and give them the option to view it and use it to make decisions about their operations.

Once gained insight as to what data farmers might like us to detect, we began to discuss potential display solutions that would best fit the general farmer’s needs.. The farm manager of the Miami Institute for food relayed to us that he would like to be able to view data from his phone. In the next section, we will discuss the solutions we considered in order to meet his needs.

**Review of Previous Designs**

Parrot is a precision agriculture company that sells products similar to our team’s intended design solution. Parrot’s most popular sensors are targeted toward individual consumers who want to view metrics that provide them with information about individual plants, particularly flowers.

Our design is different from the most common sensors that Parrot provides to its customers because our intended user is growing crops that will be used as food.

**Project Research**

**Metrics to Measure**

After discussing possible measurements with experts from the Miami University Institute for Food we have learned that he would most greatly benefit from measuring the temperature and evapotranspiration levels of the soil. The temperature values can help him see patterns over the span of a typical day and trends over a long time period that may help them make decisions such as when to water the plants or what date to harvest. Evapotranspiration determines the amount of evaporation occurring in the soil, a sensor measuring this metric would compare the soil’s water levels over time. With information on the moisture of the soil farmers can better understand how much water the plants need and how often.

**Sensors Measuring Metrics**

In order to achieve the goals of this project, we will only need two sensors. One sensor to measure the evapotranspiration, and one to measure the temperature. For the Evapotranspiration sensor, our team plans to use the eTape Sensor, and for the temperature sensor our team plans on using the Adafruit Water Proof Temperature Sensor.

**XBee**

The XBee is a high level transceiver that interfaces well with Arduino Microcontrollers and USB ports. The XBee Pro Series 1 offers high level abstraction with the XCTU program that would make it simple for us to implement. However, the XBee transceivers would be expensive and also require shields to interface with ports and microcontrollers

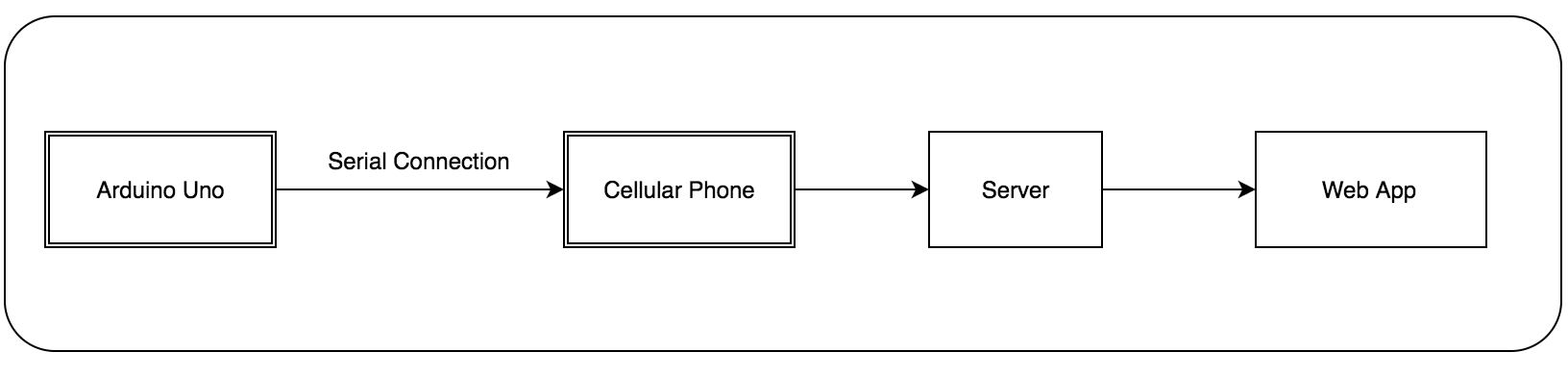
**RF Data Link**

The RF Transmitter and Receiver Pair by sparkfun are extremely inexpensive wireless transmission tools that offer basic functionality for a low cost. Two team members have also worked with these tools in the past and have the ability to integrate them into this design easily. However, these only offer a much shorter range than the XBee.

**Cell Phone**

Another option is to transmit the data through a telephone line, eliminating the need to extend a wifi signal to our hardware on the farm. There are two ways we could do this, both including an Arduino sending information to a cellular phone.

One way is to use a shield to connect a modern cell phone to an Arduino. Another, less expensive option is to could use a more dated cell phone and send information through the audio port. In the headphone jack in some older models of cell phones there is a serial port that uses TTL serial communication. A connection can be made by using the headphone jack as an interface for serial communication. The Arduino can transmit data to the cell phone which can send messages containing the tracked data to a server.



**Block Diagram of Cell Phone Serial Communication Process**

**How to Store Data**

We can run a program in python or C++ to access the ports on the raspberry pi to receive data from the XBee Shield, then post the data to our web server as a JSON message by using a SLIM framework .HTTP protocol.

We decided to choose a MySQL database to house long-term data. This because the information is very structured and will not take up vast amounts of space. We will have a small number of fields including the date and time as well as the taken measurements so this is easily fit in a table which is how a SQL database is structured.

**Displaying Data - Web Application**

After gathering data from our sources and assessing the possible solutions for displaying data, we decided that a live website/web application would be the best solution. This leads to the question of what programming languages to use to build the website.

For the front end, we decided to use HTML and CSS, as well as Bootstrap. Bootstrap was particularly important, as it allows websites to become responsive, which means the site will automatically resize depending on whether it is viewed on a cell phone or laptop. Bootstrap also includes many easy to use components, including progress bars that can be used to display metric data.

We wanted a clean, easy to understand way to display metric trends like temperature over the last 24 hours. In order to do this, we decided to use the Charts.js Javascript library. This library allows for the easy creation of charts to display these trends.

For the back end, we are planning on using PHP to connect to our database and query relevant data. We then will use Javascript functions to format this data in an easy to display way in our charts.

**Ways to Host Site**

The information will be presented to farmers on a live website. This poses the question of where the website will be hosted. One option is to host the website on a third party remote server. Examples include Amazon Web Services, Google Cloud Platform, or Microsoft Azure. To host with a third party ensures we have no physical hardware to maintain and the reliability is very high because it is a paid service. However, costs may compound in the future and maintenance may be difficult if the people maintaining the website in a few years are not well-versed in the specific third party environment.

An alternative way to host the site on a remote server is to use Miami’s Technology Services. We can have a server set up for us here at Miami and maintained by the Tech Services Dept. This would be very reliable, as the physical hardware maintaining the server is close to the farm and Miami Tech Services will always be there to maintain it. A con to using Miami’s services is that we are unsure if we will be charged in the future and if maintenance can be done in a timely manner.

To avoid the fees and abstraction associated with a remote server we can host the website on our own hardware, for example we can use an ESP 8266 Wifi microcontroller. This way we have full control over the environment that supports our data and site. However, this requires more time and effort setting up the server and that may take away from work being done on the intended project. If we have a remote server set up for us it eliminates likelihood of error in the implementation process and increases our time available to work on the website itself.

**Solution Process**

**Hardware Details**

In order to collect raw data from the Nutrition Center, our team created a sensor node that consists of three sensors, a microcontroller, and a wireless transmitter. In order to post data to a web application our team plans to use a raspberry Pi 2b to interface with a wireless receiver and connect to the internet to post the data to a web application using a slim framework.

**Sensors**

Sensor Name: Sparkfun - Temperature Sensor - Waterproof

Sensor ID: DS18B20

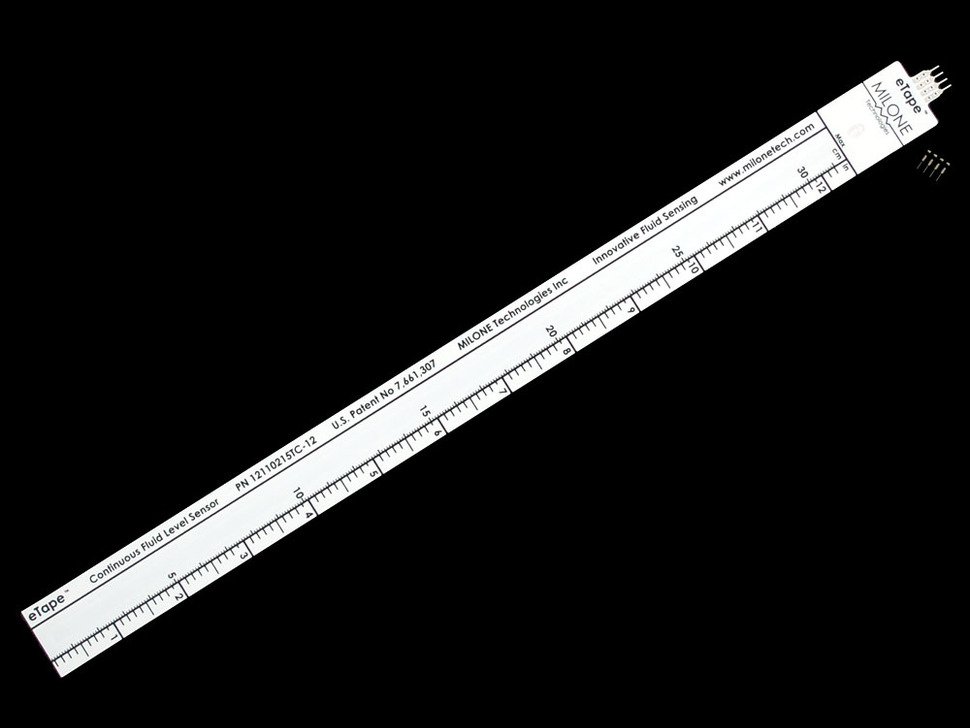
The water temperature sensor uses the oneWire library to interface with the Arduino Uno. The temperature data is transmitted to the receiver node.



**Sensor 1: Waterproof Temperature sensor**

Sensor Name: eTape Sensor

This sensor acts as a variable resistor and outputs voltage to the analog input pin of the microcontroller to indicate the level of water from 1-8 inches.

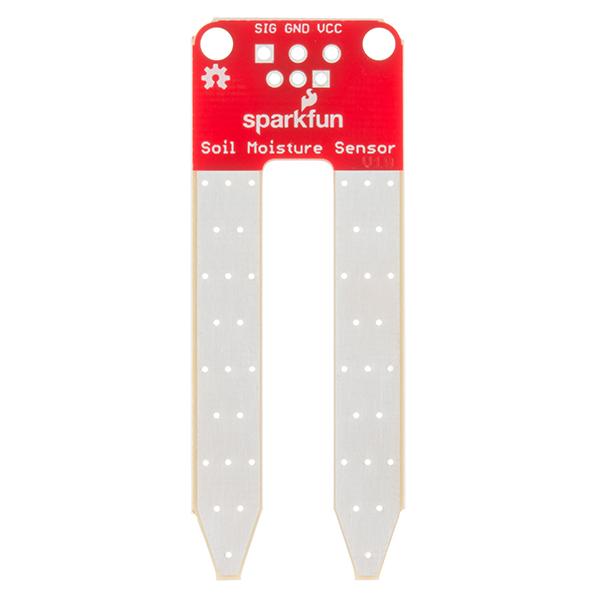


**Sensor 2: Water Level Sensor**

Sensor Name: Sparkfun Soil Moisture Sensor

Sensor ID: SEN - 13322

The soil moisture sensor acts as a variable resistor. As the soil moisture increases, the resistance of the sensor decreases. This causes the voltage to increase as soil moisture increases. The data from this sensor is used to control the irrigation system, and it is also transmitted to the receiver node.



**Sensor 3: Soil Moisture Sensor**

Sensor Name: Xbee Series 1 Pro

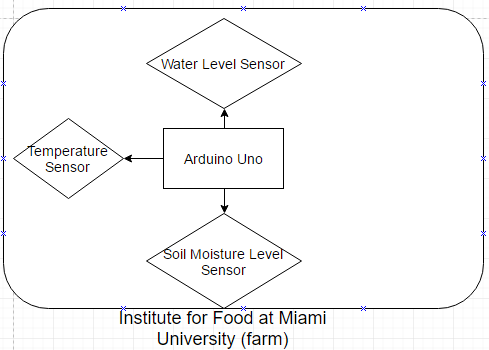
This is a wireless communication device that can be configured to be a transmitter or a receiver using the XCTU software.



**Wireless Transmitter and Receiver: Xbee**

**Sensor Configuration**

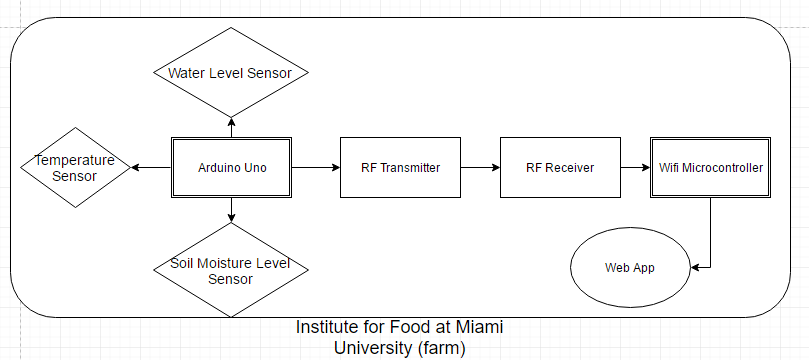
After doing our research on various components, we came up with three complete solutions to compare. All three solutions shared most functionality, including using an Arduino Uno microcontroller to interface with the temperature, water level, and soil moisture level sensors.



Block Diagram of Sensors and Arduino Uno

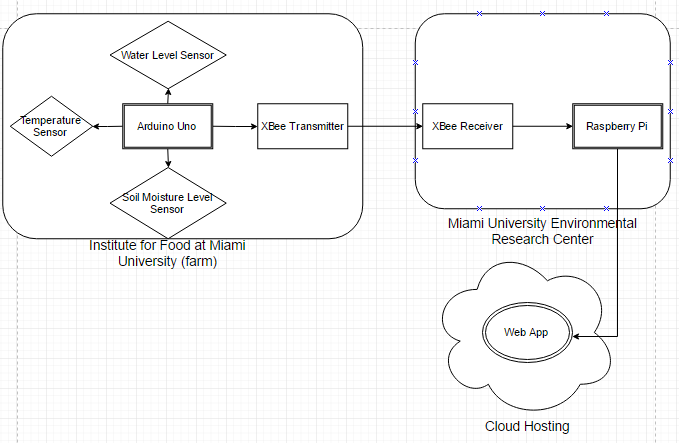
Where these solutions differed was how to store, transmit, and display the relevant data. Below, each solution is explained in detail:

1. **Locally Host Web App with Wifi Microcontroller**



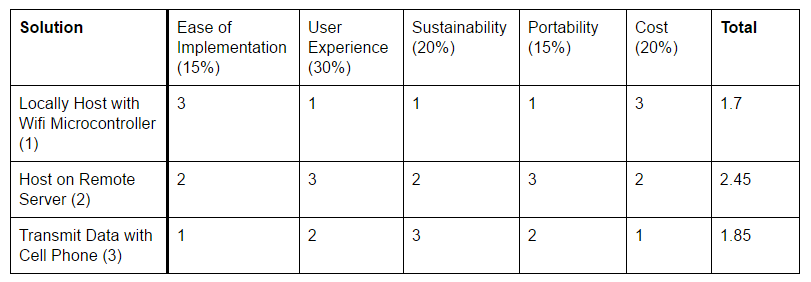
This solution consisted of using the Sparkfun RF Data-Link Transmitter and Receiver to transmit the farm data to another module also located at the farm. This solution would be extremely easy to implement, since we have experience with all of these components already. However, this would mean that farmers could only access farm data when close to the wifi microcontroller, not from his home or even certain locations at the farm. Additionally, it would not be feasible to store lots of data on the wifi microcontroller, which was important to the manager of the Miami University Institute for Food. If anything malfunctioned with the wifi microcontroller, it would be very difficult for someone to identify the issues and correct them. This would be a low cost solution, since we already own all of the components involved.

1. **Host Web App on Remote Server**



This solution consists of using an XBee transmitter and receiver to transmit farm data to a remote location, Miami University Environmental Research Center (ERC). The ERC was chosen because it has MU-Wireless internet access. From the ERC, a Raspberry Pi is used to store the data in an SQL database. From there, the relevant data is posted to our web application which is hosted in the cloud by a third party service. This solution requires a lot of on-the-project learning, as we don’t have experience with XBee technology or with storing data received from XBees in a database. However, this solution provides a great user experience for users, as it will allow users to view all relevant metrics on any internet-enabled device from wherever they are. It will also be possible to access past data via the database created. This solution would require a low level of maintenance in the future as long as someone was maintaining the server the web app was hosted on, as well as the web app itself. This solution would could easily be applied to other similar problems. It would be as easy as implementing another sensor module, transmit the data, and store and POST the data using a Raspberry Pi. This solution would require the additional purchase of XBee components as well as potential cost associated with hosting a web app.

**Choosing a Solution**



After devising the above three solutions, we wanted to create a way to quantitatively rank each solution to pick the best for the requirements of the project. We first decided on the five most important factors in choosing a solution:

* Ease of Implementation - how easy it will be on our end to implement solution.
* User Experience - how good is the overall experience for the user. This includes where the data is accessible from, how easy it is to view relevant metrics, etc.
* Sustainability - how sustainable will our solution be in the future, and will this be a solution that can be easily maintained by future senior design groups or Miami University so it can continue to benefit users.
* Portability - how can we take our solution and apply it to other problems.
* Cost - how much does our solution cost to build and maintain.

The solutions were compared head to head with these factors and ranked from best (3) to worst (1). Additionally, each factor was weighted based on its importance. The most important factor to us was user experience, which carried a weight of 30%.

Based on these rankings, “Host on a Remote Server" won by a significant margin. Thus, we decided to implement this solution.

**Challenges Encountered**

One of the biggest challenges we faced was being able to transmit and receive data via XBee in real time. We were able to successfully configure the XBees so that they could communicate with each other by using XCTU. We had one XBee connected to a laptop and the other one connected to the arduino uno via shield. We were able to send data by using the serial monitor on the arduino and view what was received in XCTU. However, we had to physically type out the data that we wanted to send. The data would not send automatically in real time, which is what we needed it to do. After spending several weeks of trying to solve this problem, we were at a loss. We had looked at every resource online we could find and tried changing up the arduino code, but nothing worked. Finally we decided to connect the XBee to the arduino directly and forgo the shield we had been using. This in fact turned out to be the problem all along. As soon as we removed the shield from the equation we were able to send and receive data in real time.

The next problem we faced was trying to read in and interpret the data that we were sending. The data was being transmitted as bytes in hexidecimal. We converted this back to decimal in order to figure out which bytes of data we would need to post to our database.

**Results**

**Sensor Node**

As addressed in our team’s design solutions, there are several different sensors that we are incorporating into this project. The most important one to our customer is the temperature sensor so that we can record soil temperature.

So far this semester, our team has obtained both successful data from the temperature sensor, the water level sensor, and the soil moisture sensor. In order to test the accuracy of these sensors, we viewed data using the serial output function in the arduino development environment.

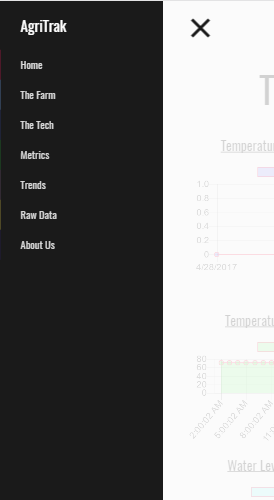
We will also be implementing an evapotranspiration sensor to measure the water level. This will allow our customer to see how much and how quickly water is being absorbed by different crops. After we get these sensors implementing and transmitting accurate data we can look to add more if the customer finds it beneficial to do so.

**Database**

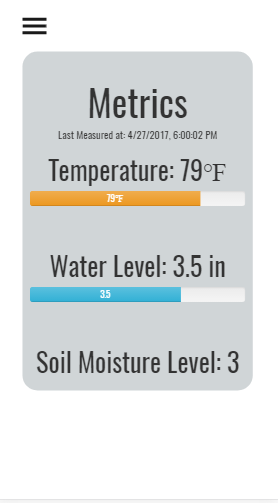
The type of database we used is NoSQL because it provides flexibility of data fields and allowed us to simplify our database design. Specifically, we used MongoDB because it has a wide range of tutorials and documentation online. To store our data, we used mLab.com which is a MongoDB hosting website. 500 MB of data is free, which allowed us to setup our database immediately. We stored the following metrics: date/time, temperature, water level, and soil moisture level.

The database was then queried by our web app using the mLab data API. We also used this API to POST data to our database via a Python script on the Raspberry Pi.

**Web Application**



The AgriTrak app (<http://benjamam.cse252.spikeshroud.com/AgriTrak>) is full-featured with many different pages. It includes a homepage with a brief introduction about AgriTrak, “The Farm” page that contains information about the Institute for Food, “Metrics” page that contains the current metrics, “Trends” page that contains metric trends, “Raw Data” page that contains the raw data from the past two days and the ability to export it to an Excel spreadsheet, and “About Us” page that contains information about our team.



The AgriTrak web application was designed to have a simple and clean interface that focuses only on displaying the most relevant data to the user.

The temperature bar changes depending on the current temperature. If the temperature is below freezing, the bar is blue, as seen below.



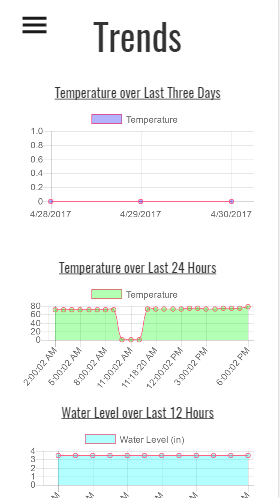
If the temperature is in optimal growing range, the bar is green, as seen below.



If the temperature is too hot for optimal growing, the bar is orange, as seen below:



These values are populated with real data from the MongoDB database being hosted on mLab.com.



Metric trends are displayed in graphical form. The most important trend for our users was being able to track when temperature was below freezing. The green graph displays temperature over the last 24 hours. By displaying this data to users in an easy to read format, it will allow him to make informed decisions about his crops.

The other trends deemed as important by the experts at the Miami University Institute for Food were average temperature over the past three days and water level over the past 12 hours.

These charts were rendered using the easy to use chart.js Javascript library. Chart objects were created using predefined parameters and labels. These parameters were populated with the farm data and the charts were automatically created via the library.

All the mLab database was queried via mLab API to obtain all of the data for the web app. The query simply returns all of the data in the database. This was done via Javascript on the front-end, which is not the preferred web programmer’s approach. As the data continues to be collected, this query will take longer and longer. In turn, this will cause the web pages to load more slowly. However, for the scope our project this was an effective approach.

The styling of the web app was done with HTML, CSS, and Bootstrap 3. Javascript was used for load animations as well as the data handling and display. PHP was used very minorly to create “views” for the slide out menu bar, which allowed the code for the menu bar to be reused for each page instead of rewritten.

If we had more time, all of the database querying and data manipulation would have been done with PHP on the backend.

**Future Plans**

We accomplished all of our goals for this project, so the next phase would be implementation. We would need to build some sort of enclosure for the sensor node to protect the arduino and XBee from the elements as well as give it a power source--battery pack would suffice. We would also need to develop additional sensor nodes that could be dispersed across the farm so that the farmer can have access to see if there are different trends on different areas of the farm. As we add additional sensor nodes, we would likely need to create a control node for communication between the sensor nodes and the receiver node.

**Conclusion**

This semester we managed to complete all of the goals we initially started with for this project from last semester. We were able to get the XBees to send and receive data in real time, sort through this data, and post it to our database by using a Raspberry Pi. On the software side, we developed a website where this data can be viewed and have helpful trends generated and displayed in graphical format. If any more work is going to be done on this project, the next step would be to physically implement it on the farm.

The solution we created has the potential to be implemented beyond just the Institute for Food as well. By working with a farm manager, we were able to learn about the relevant metrics for farming. By measuring and displaying these metrics, we have a general solution that can be implemented in the future on other small farms or in urban farming solutions.

This solution allows for farmers to better allocate their resources and make informed decisions about their crops. Ground temperature alerts farmers about about frost. If plants endure frost for too long, farmers can make the decision to replant instead of wasting resources on lost crops. Soil moisture and evapotranspiration levels ensure that farmers are always allocating the optimal amount of water to crops. This limits over watering and saves this valuable resource.

**Team Reflection**

This project taught us that we have a professional and ethical responsibility as engineers to our “customer” -- in this case Charles. Our partner for this project was the Miami University Institute for Food. We had a responsibility to develop a system that would give them accurate data so that they can do their jobs.

Another realization that we have come to from working on this project is the need for and an ability to engage in life-long learning. This is especially true when it comes to working in STEM fields because technology is always changing and breakthroughs are being made. For example, with our project we need to be able to transmit data signals across a couple hundred yards. We have not had experience working across long distances like this, but after doing some research, we found that an option was to use the XBee Pro as a transmitter/receiver. We then had to do further research on how to interface the XBees with each other and with the Arduino. It is good experience looking into new technologies and figuring out how to work with them. A skill that will come into play and affect one’s success in industry.

This project helped our team better understand the impact of engineering on society. In our time here at Miami University, we have spent so much time learning niche skills and working on technical projects. In order to become the best engineers we can be, we must spend a good amount of time doing both of these things, but it is often hard to become lost in the technicalities of the task at hand while forgetting the importance of the impact that a project might have. Interdisciplinary projects like this project, have really help prepare our team to strive to gain a better understanding of the roles we will have in our careers in the future so that we can have as great an impact as possible on the world.

**References**

<http://global.parrot.com/au/products/flower-power/>

<http://www.chartjs.org/>

<http://getbootstrap.com/components/>

<https://www.adafruit.com/products/464?gclid=CjwKEAiAyanCBRDkiO6M_rDroH0SJAAfZ4KLYheZn95ieP5fpB2-oHozEkjId1Lu0ydU0ruTOi-O5xoClDvw_wcB>

<https://www.adafruit.com/product/381>

<https://www.sparkfun.com/products/13322>

<https://www.adafruit.com/products/128>