FRESH VIEW

Formal Report

By: Urban Fresh

Pawan Kumar
Solomon Lindsay
Varun Prabhakar

Recipient: Mel Dundas

ECET 290 - Applied Research Project

Dec 8, 2018

Memorandum

To: Stephen Tordoff

CC: Mel Dundas, Wayne Mayes, Kimberly Lemieux

From: Urban Fresh

Date: Nov 28, 2018

Subject: Fresh View Final Report

We, the Urban Fresh group, are pleased to present the second iteration Fresh View prototype. After three months of hard work we have created a product to satisfy Camosun College's capstone project requirements, and our own self defined requirements.

The process was challenging but all the more valuable for it. We advanced the genesis Fresh View prototype to integrate a custom PCB, a more compact and efficient microcontroller, a camera, and a custom app to access data from anywhere. Additionally we created an example control environment to show that our unit can be used to automate the tending of a plant from germination to vegetation.

We would like to thank our client, Energy Canvas, for the allowing us to advance the Urban Fresh technology and for his help along the process. We also thank Mel Dundas, Kimberly Lemieux, and the rest of the college instructors and staff for their support and guidance.

Executive Summary

Over the past 13 weeks our team, Urban Fresh, developed a unique product to assist farmers or gardeners in monitoring and growing their plants. The Fresh View system allows users to view their crop and analyze growing conditions anywhere they have internet. With this solution users can have the right information and the right time to prevent crop failure, increase yield, and in the grand scale help improve food security.

To develop this solution our team worked hard to upgrade the design from the genesis model provided by our sponsor, Energy Canvas. This initial model measured essential parameters of the growing environment and uploaded that data to a cloud storage system for the user. From this first model we:

- Introduced an improved processor
- Upgraded sensors
- Added a camera
- Added an on-device display
- Upgraded the cloud storage system
- Designed a custom app
- Started development of an automatic plant environment control unit
- Reduce the size and improve the resiliency

Although we encountered several challenges along the path, we managed to fulfill most of our requirements, remain below our $200 budget, and finish with an operating prototype.

Overall our team is satisfied with our performance in the project and the resulting outcome. We are especially satisfied with what we learned over the course of the project in areas of project and management, programming and app development, and design. We believe we have developed a valuable and unique product worthy of 13 weeks of hard work.

Table of Contents

Table of Figures

1.0 Introduction

The Fresh View system is a low cost flexible plant monitoring system. It shows the live conditions of a plant growing environment, including numerical and visual data, on a mobile app with internet connection. In addition to live data, Fresh View keeps a record of past data on a cloud based system where the history can be accessed to see the plant from germination to harvest. This way we can see what environment conditions work best for a plant and use that to improve plant growth in the future. This technology is designed to be flexible and can work in greenhouses and hydroponics systems where the environment conditions must be controlled.

The Urban Fresh team consists of Solomon Lindsay, Pawan Kumar, and Varun Prabhakar. With the support of our client, Energy Canvas, and Camosun College, we were able to complete our project on time and within the set requirements.

2.0 Purpose

As the world population grows, food demand grows with it. Food security is now a major world problem in need of more creative solutions. Our solution addresses this by removing the issue of unstable growing conditions resulting in failed crops. Fresh View will accomplish this by allowing the user to monitor plant conditions, identify and prevent failing crops, and automatically control the growing environment for optimal plant yield.

The system is designed to be low profile, low cost, and flexible to work in any growing environment from backyard gardens to hydroponic farms. Fresh View has the potential to be an invaluable tool in the belt of any farmer that will improve the resilience of today's farms to feed people of the future.

3.0 Our Process

Fresh View was based on work done by Solomon over a Summer work term at Energy Canvas in Vancouver. During this term Solomon worked with Energy Canvas to develop a renewably powered hydroponic system. One outcome of this work was the Fresh View genesis prototype. This initial model measured the essential parameters of the growing environment and uploaded that data to Google Spreadsheet accessible on the cloud. The mission of our project was to upgrade this genesis model with superior components, a more compact design, a camera, and a custom web application, as well as explore the possibly for automatic environment control.

The process to upgrade the genesis model began by upgrading most of the hardware with more convenient and reliable components. With the hardware selected we programmed the basic functions to read the sensors and upload data to the cloud. Once the core of the monitor unit was

working, we simultaneously began designing our custom printed circuit board (PCB) and programming the user interface to display data from the cloud to the user. Finally, we assembled the monitor, added the camera module, and created the enclosure. With the unit assembled we were free to run our testing and verify all functions worked properly.

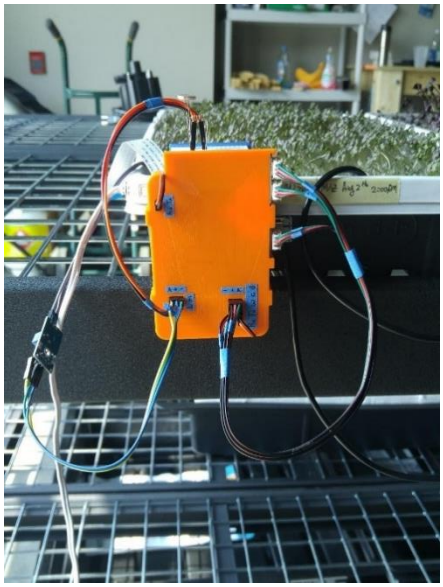


Figure 1: Genesis Model

Figure 2: Fresh View Prototype

4.0 Project components

4.1 Hardware

Our first step in upgrading the genesis unit was selecting the appropriate hardware to improve on the prototype components. We first replaced the Raspberry Pi computer and Arduino microcontroller used by the genesis unit to read sensors and connect to the internet. To save space, cost, and reduce complexity, we used one simple microcontroller. We also upgraded all the prototype sensors to be lower profile (non-prototyping components) and more resilient for long term operation in a plant environment. Lastly, we selected a camera that would be compatible with our selected board. Once an appropriate board was selected, the sensor had been chosen, and a camera module was identified, we were able to order all hardware and focus our programming to the specific devices.

The final Fresh View monitoring unit consists of a micro controller, four sensor devices, a camera, and a custom printed circuit board. The whole system is powered by 5V from a micro USB connector.

4.1.1 Microcontroller:

We first had to replace the Arduino and Raspberry Pi with a microcontroller that had sufficient GPIO to read all sensors and had Wi-Fi connectivity to upload data. The Heltec ESP32 Wi-Fi LoRa microcontroller (ESP32) filled all those requirements in a low-cost and low-profile unit. It also had the advantages of using the Arduino Integrated Development Environment (IDE) which we were already familiar programming in, and it came equipped with a built in OLED display to show our readings on-board the monitoring unit.

Figure 3: ESP32 heltec

4.1.2 Camera

Providing a visual link to the plant environment was an important feature of our project. To accomplish this we had to select and program a camera module that would work with the limited GPIO of the microcontroller and the C++ programming language accessible in the development environment. Initially we attempted to use the OV7670 FIFO camera module; however, this proved to be too complicated to work with the ESP32 microcontroller we were using (see Section 7.0 for more information). Instead we used the Arducam 2MP Mini camera module which is designed to work specifically with the limited GPIO of an Arduino microcontroller using simple SPI and I2C interfacing. With this camera module we are able to capture and upload pictures to a server at the user's demand.

Figure 4: Arducam

4.1.3 Sensors

To provide a detailed analysis of the plant environment we used 4 sensors to read 5 essential parameters: humidity, air temperature, water temperature, moisture, light intensity.

4.1.3.1 Temperature/Humidity

To measure the temperature and relative humidity of the growing atmosphere we used an HIH7120 Temp/Humidity Sensor. Measuring the ambient temperature and humidity of the growing atmosphere allows us to know when a crop is at risk of failure as a result of extreme hot or cold temperatures, disease or molding from high humidity, or dehydration and stress from low humidity. With this data we can determine if a crop will require ventilation or further intervention to prevent serious damage.

Figure 5: HIH7120

We selected the HIH7120 specifically because this device uses I2C interfacing to the GPIO of the microcontroller, allowing us to get high accuracy and reliability with a basic programming library. Additionally, it is very low profile while having the ease of solderability with through hole leads.

4.1.3.2 Water Temperature

Figure 6: DS18B20

The water/soil temperature of the growing environment is measured by the DS18B20 waterproof temperature sensor. This measurement allows us to track the state of the soil or water temperature and indicate when the plant might be at risk of failure from extreme temperatures. Additionally, the long lead and waterproofing provides greater flexibility than the HIH7120, so we can get a better picture of the growing environment as well as to help verify air temperature readings. The DS18B20 is also easy to program using 1-Wire interfacing to transmit data reliably.

4.1.3.3 Moisture

Figure 7: Capacitive Moisture Sensor

The soil moisture levels are measured by a capacitive moisture sensor. Measuring the moisture content in the growing environment allows us to determine when a crop may be lacking water or at risk of drowning. This is very useful to determine when a crop needs watering or is at risk of root rot which can cause crop failure. We used a capacitive moisture sensor over the original conductive moisture sensor of the genesis model because of the greater longevity. Due to the capacitive nature, no conductive material of the sensor contacts soil, so it will not oxidize quickly and degrade like the conductive sensor did. It provides an analog value to indicate the conductivity of the substance it is placed in. This reading requires calibration depending on the substance it is in; however, it provides very valuable information that no other sensor can provide.

4.1.3.4 Light Intensity

Figure 8: Photoresistor

The light intensity is measured using a simple photoresistor. Measuring the light intensity allows us to track how much light a plant is receiving and can indicate is light augmentation is needed. We used a photoresistor with a simple voltage divider circuit as the component is readily available at minimal cost and provides as good an estimate of light intensity as we can measure without investing in an expensive PAR meter. Using the voltage divider circuit, we read an analog signal and convert that into a light intensity reading after calibration.

4.2 Software

The greatest improvement on the genesis model was the backend programming that read the sensors, stored the data, and displayed it to the user. Programming the system required the development of three different parts: the micro controller of the monitoring unit that reads and uploads sensor data, the online database where information was stored, and finally the user interface of the Android application.

4.2.1 Microcontroller

The microcontroller is the most essential component of the Fresh View system. It is the device that reads all the sensors, the camera, and uploads it to the internet. Due to its importance the microcontroller was the first piece of the system we worked on.

As described in Section 4.1.1 we replaced the genesis model's Raspberry Pi and Arduino with an ESP32 microcontroller. The ESP32 was compatible with the Arduino IDE to program which allowed us to use a programming language and environment that we were familiar with; however, there were some unique challenges presented by replacing the Raspberry Pi with the ESP32. The largest challenge was the fact that the ESP32 could not be programmed in python unlike the Raspberry Pi. Although this was not an issue for reading sensors and doing most of the plant monitor's functions, this did make things more complicated for uploading data to the cloud, especially when programming the camera.

Despite the limitations of programming in C++, we were able to program the device to read all the sensors, the camera, and upload that information via the internet. The code begins by initializing all the necessary libraries (1-Wire for the DS18B20 temperature sensor and SPI and I2C for the camera and HIH7120 temp/humidity sensor) and establishing an internet connection. If the correct SSID and password was programmed into the code to connect to the internet then a connection will be established and it will enter the main function of the code. With the initialization done, the ESP32 will enter a loop to measure each of the four sensors, convert the data into strings, upload that data to the online database, test the server for a camera request, then repeat. If the ESP32 does detect a camera request from the user sent through the phone application then the ESP32 will temporarily break the loop, take a picture on the Arducam, and then post that picture to the dedicated camera server before returning to the regular loop. This whole process can be delayed to trigger at set intervals throughout the day; however, the time to communicate with the online database limits this to at least approximately 5 seconds.

4.2.2 Firebase

To store all our information from the plant monitoring unit we used Firebase as our online database. The genesis model used a basic Google Forms Spreadsheet to store and track data which was functional for the prototype; however, we wanted to upgrade this to a more flexible and customizable system. Firebase served as the ideal upgrade with its flexibility, greater storage, and the ease of connectivity being a Google product.

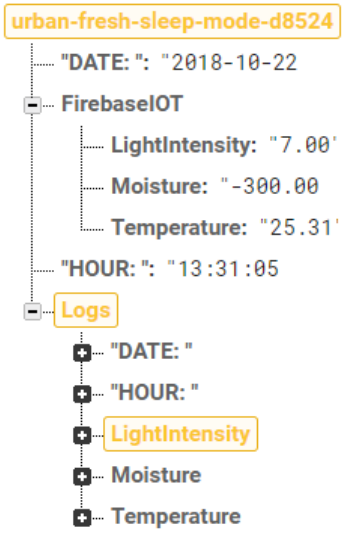


Figure 9: Firebase

To use and customize Firebase we simply needed a Google ID and access to the internet. Being a cloud based data storage system, Firebase is exists completely on the internet. Once we created the workspace for Firebase and the variables in which data would be stored, we could program the ESP32 to connect to Firebase. To accomplish this we simply used a C++ based Firebase programming library and in a number of lines our connection was established. To accommodate the datatypes supported by Firebase we programmed the ESP32 to convert all data to strings before uploading. Once this was accomplished and data was being stored in Firebase, we started work phone application through which users would access data.

4.2.3 Mobile Application

The mobile application was another important upgrade from the genesis model. Data from the genesis model was only accessible through a shared Google Spreadsheet which was clumsy and not very user friendly. We wanted to create a custom website or application that would be accessible to the user wherever they are, and display the necessary information in a convenient manner. After some research we determined that the limited time available to complete the project would make it difficult to learn the java script or python necessary to develop a custom website or app. As an alternative we found MIT App Inventor.

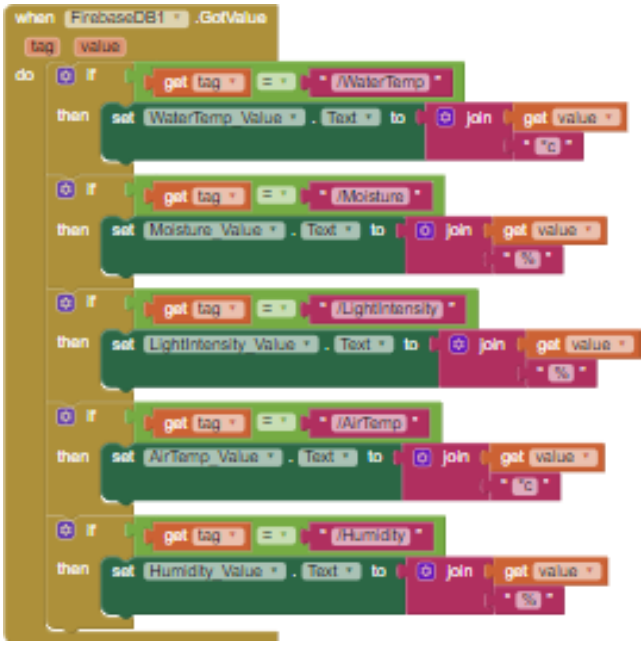


Figure 10: MIT App Inventor

MIT App Inventor is a user friendly Android app development environment. Using this allowed us to develop a customized user interface that connects to our database and serves all our needs without having to learn an entirely new language; however, the simplicity did come at the cost of some limited functionality. The limited flexibility of the MIT App Inventor environment meant we could only display the most recent environment data and display a full history as originally intended. The same issue applied to the plant photos where we could only provide the most recent photo and not a complete history as intended.

In the end our Android application served as a valuable tool for the user to access the most recent data and photos. The application displays as a single page when opened (see Figure 11).

At the top of the application page each of the five environmental parameters are displayed and updated every second. The user can evaluate these readings to determine what the plant environment needs for optimal yield, or determine if a crop might be at risk of failure.

Below the environment readings are the controls for the Fan, Pump, and Lights. The Fan and Pump can be enabled for automatic control. If enabled, the current temperature and/or moisture value will be tested against the set value entered by the user into the corresponding text box. If the control unit detects one of these values going beyond the user set value, the fan or pump will turn on to correct the value.

The Light control will turn the lights on in the control unit when enabled and turn them off when disabled.

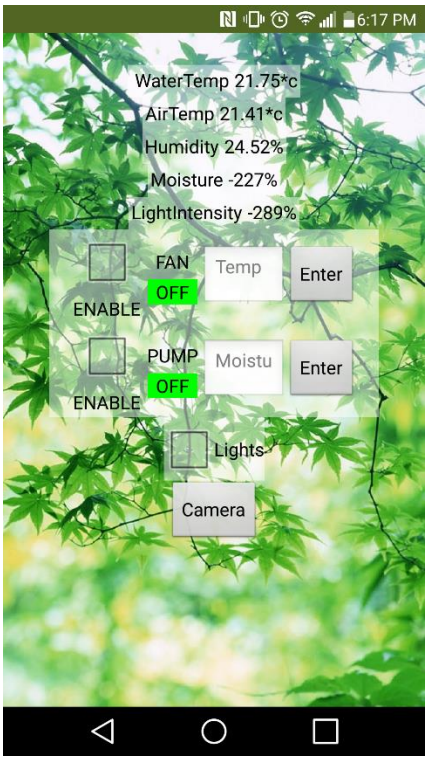


Figure 11: User Interface App

The last feature on this screen is the Camera button. When pressed this button will direct the user to a web browser where the most recent picture will be loaded from the ESP32. Due to the limits of data transfer from the ESP32, this feature only works when the monitor unit and camera are connected to the same Wi-Fi network.

5.0 Design & Manufacturing

5.1 Schematic Design

Creating a custom Printed Circuit Board (PCB) for the monitoring unit allowed us to greatly reduce the profile of the hardware as all the components could be directly connected without the need of a bulky prototyping breadboard. Once all the components had been ordered and their proper operation was verified on a prototyping board, we proceeded to design the custom PCB.

To design the PCB we had to meet specifications of our own design, the school, and the manufacturer. The manufacturer we were using was JLC PCB. Using the manufacturer fulfilled the school's PCB requirements and guaranteed a level of reliability; however, because they were based in China we had to be conscious of the long lead time to receive PCBs, especially in the context of our limited timeline. Once we had confirmed all of our components worked

properly we completed our PCB design and submitted them to JLC PCB. Six business days later our PCBs arrived in working order.

Our own requirements for the PCB design was to minimize the footprint while connecting all of our six active components. As described in Section 4.1, we had an ESP32 microcontroller, Arducam camera module, HIH7120 temperature and humidity sensor, DS18B20 waterproof temperature sensor, capacitive moisture sensor, and photoresitor light intensity sensor. Because this unit was still a prototype device, we used standard 2.24mm connectors for each active device so they could be removed and replaced easily. In the end our PCB fit within our specs by connecting all our components using removable connectors, and maintaining a minimal profile at 40mm wide and 70mm long.

5.2 Enclosure Design

Once our PCB was delivered and our components attached, we had the final dimensions to design and manufacture the Fresh View monitor enclosure. To manufacture this we used the school's Ultimaker 3 3D printers. Using the 3D printers gave us the flexibility to design a custom fit enclosure quickly, and at no cost.

To design the enclosure we used Autodesk Fusion 360. We used this modelling environment throughout the course for our Design for Manufacturing course, and Solomon had become experienced working in Fusion 360 over his summer working at Energy Canvas on the genesis model. When designing the enclosure we were prioritized maintaining a low profile, simplicity, and ease of access.

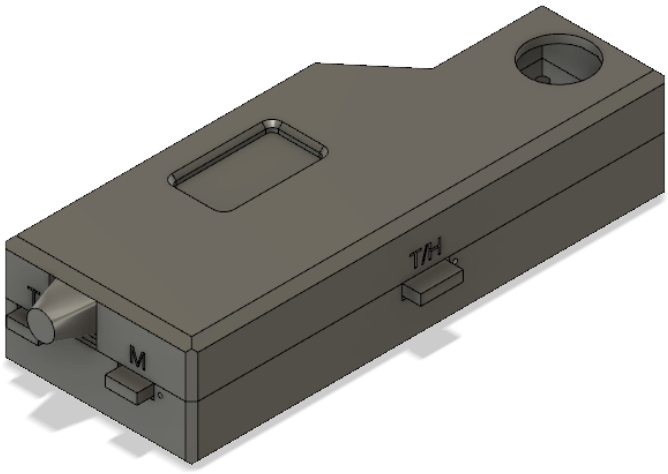


Figure 12: Fusion Fresh View Model

We printed the initial design and after a series of iterations ended with the design shown in Figure 12. This enclosure was printed on the school's 3D printers using Polylactic Acid filament. As mentioned before 3D printing the design gave us the advantages of speed and flexibility; however, this came at the cost of resiliency to the harsh conditions of a growing environment.

6.0 Challenges

6.1 Parts Ordering

As a result of working off the genesis model of the Fresh View model, we had to make sure our client, Energy Canvas, agreed with the direction of the project. Although our goals were aligned, there were complications with Intellectual Property protection when working with the school. With a satisfactory agreement Energy Canvas might have taken us on as an official project and funded our work. If no satisfactory agreement was made, we would continue as a normal project group under the school's budget. Due to this uncertainty in our budget we delayed ordering parts until we could know how much funding we would have access to. While waiting we focus on programming the devices we already had; however, it cost us time as we had to wait for all the part's arrival to test them and create the PCB. As a result of waiting to confirm parts we delayed every part of the project that depended on those parts. This delay was accommodated in our timeline, but it required other tasks to be rushed resulting in further issues.

In the future it would be important to stick more firmly to set timelines and identify delays early before they become serious. Due to our complacency in resolving this issue the entirety of our project was affected. Such delays could be avoided if a good plan is made from the beginning and more firmly communicated with all parties involved, including any clients or stakeholders when decisions must be made.

6.2 Camera Programming

The camera was one of the most crucial parts of our project but also involved the most complexity. When selecting our components, we tried to minimize costs and as such we used the readily available OV7670 camera module. This device had 20 pins which used an interfacing protocol specific to the device. This made both wiring and programming prohibitively complicated. When we recognized this, we decided to change to a more user-friendly device. After some research we identified the Arducam Mini as a good alternative with 8 pins and standard SPI and I2C communication. Although we did identify a viable alternative the delay to order the new component lead to more delays in the project. In the end this delay and continued complications in programming the new device required us to limit some of our requirements. The camera did end up functioning; however, not to the level we were expecting.

To avoid this sort of problem in the future it would be important to identify the priority of a projects requirements and schedule the higher priority first. If we had begun programming the camera as one of our first tasks we would have identified the issue early and had time to program it to full functionality.

6.3 Application Development

In the start of the project we wanted to create our own website to store the data collected by our sensors and then manipulate it any way we wanted but later in the research we realized that in order to make our website we have to learn HTML or Java which was not possible in 2-3 weeks. So, we had to use an easy to use App inventor from MIT which can act as a GUI. Using MIT app inventor instead of website, we had to make some changes in the requirements document as now the user will be able to monitor the plant growth using that app and there will be no need of logging into the website. We learned that in time sensitive projects, we should keep the requirements inside the boundaries of our knowledge.

6.4 PCB errors

As discussed in Section 5.1, we were careful not to make mistakes in the PCB as the lead time to get a replacement would be too long. Although we were careful and we made sure to test each component on a breadboard before finalizing our PCB design, we did encounter two minor errors with the PCB. The wiring for the HIH7120 Temp/Humidity sensor ground (Vdd) and power (Vcc) was switched on the footprint, resulting in the complete failure of that part until the device pins were re-soldered to match the PCB footprint (this was easily accomplished as we were using long leads to extend the sensor beyond the enclosure). The second error was with the wiring of the Arducam camera module. The Chip Select (CS) pin for the camera's SPI interfacing had to be changed to another pin on the GPIO so it would not interfere with the ESP32's built in LoRa function. To solve this we had to manually break the trace from the camera to the original CS pin on the ESP32 and solder a wire to connect it to the new pin.

Both of these errors came as a result of the rushed timeline. Because we were late to order our parts (as described in section 6.1), we were rushed to get our PCB order into the manufacturer. As a result we were not able to fully test the Arducam to identify the problem with the CS chip, and we were rushed when double checking the sensor connections. In the future it would be important to plan the time properly to fully test every component on a breadboard before manufacturing a PCB, and dedicate a full day after the PCB design is finished to cover every component's connection with everyone else involved in the design and programming.

6.5 Control Unit

Creating a control unit was one of our earlier objectives that we hoped we would be able to achieve. The premise was to create an enclosed environment that could be automatically controlled using the Fresh View monitoring unit and a paired control system. Although we did go through the process of assembling a test environment and acquiring the parts for a control system, it ended up being beyond the scope allowed by our timeline.

This issue was more a result of our limited time than it being a challenge itself. For the future it would be important to manage the time so that all of the scope can be accomplished, however it was good that we identified this as a lower priority early and as a result were able to focus on

more important things which allowed us to create a finished product. Developing a control unit is something we still wish to do and definitely something for future development.

7.0 Future Development

Over the course of the past 13 weeks we were able to accomplish a lot and fulfill many of the requirements we set; however, there is still a lot of potential for Fresh View to advance.

The genesis model was the result of a project to develop a renewably powered hydroponic system at Energy Canvas. Fresh View has since been greatly improved and could be easily integrated back into the same project as a useful piece in a larger system. Fresh View also has the potential to be a stand-alone product to help users grow more reliably and with greater yield. In either of these cases, the Fresh View monitoring system can be improved in a number of ways.

7.1 Picture Storage

One of the greatest challenges was programming the camera module (see Section 6.2). Our requirement was to allow plant photographs to be displayed alongside plant data and their time stamp. This was not possible with our limit in time and programming knowledge, but is something that could be done and could add great value to the product.

With picture storage a more thorough history of the plant environment could be explored and more capabilities could be added such as machine learning. Machine learning could be used to analyze pictures of the growing environment and automatically identify different stages of growth and potential problems (such as disease or infection).

7.2 Upgraded User Interface

In our limited timeline the user friendly development environment of MIT App Inventor was the most practical tool; however, with more time a proper app or website could be developed to show live and recorded data and display it in more convenient ways. For instance, data could be graphed to show the trends of a plant environment, and data could be tagged and sorted so that multiple devices and multiple crops could be monitored with a single system.

7.3 Battery Power

The ESP32 comes equipped with a battery adapter to allow it to run completely wirelessly. We did not have the time to acquire an appropriate battery and program the power saving functions necessary, but it would increase the functionality of the device greatly if it were to run truly wirelessly.

7.4 Ingress Protection

Finally, the enclosure we designed and printed for the Fresh View monitoring unit does provide some modicum of protection; however, the PLA material and method of sensor connections do not provide the resilience necessary to operate in a harsh growing environment for a long period of time. With the use of proper IP rated connectors and a stronger print material, the enclosure could be made much more resilient and practical to the user's needs.

8.0 Financials

When designing the upgraded model of the Fresh View prototype we wanted to minimize the cost as much as possible but improve the resilience of the device at the same time. We were able to reduce some costs by using a single control board and not using "plug-and-play" prototyping components, but the low volume of sensors and issues with the camera increased costs.

Of our $200 budget provided by the school we spent $168 in total (see Appendix A for more details). Of that budget some money went towards the control unit, and some parts for the monitor were provided by the school. When accounting for all provided parts and discounting the control unit parts, the Fresh View monitor system costs $135.45, $88.62 of which goes towards micro controller and the camera. If buying in manufacturing volume the cost per unit would go down greatly; however, a lot more progress would have to be made before that was viable.

9.0 Conclusions

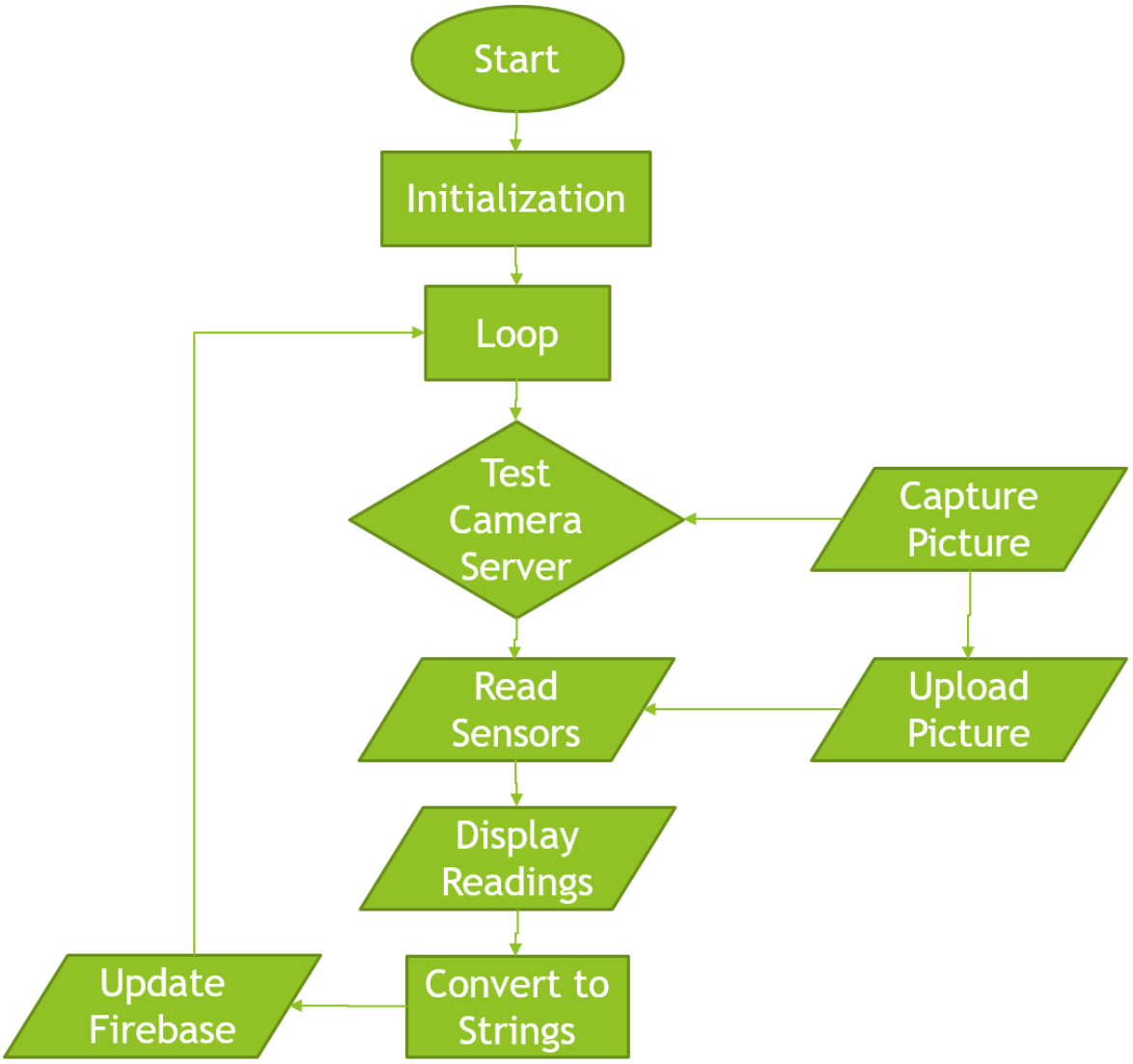
The Fresh View system has been an engaging and exciting challenge to work on over the past 13 weeks. Although we encountered several difficulties along the way and we did not fulfill every requirement we set, in the limited time we made great progress in advancing Fresh View to become an innovative product for the future. Over the course of this semester we learned many lessons that will be valuable far into our careers. Practicing this project as a real-world application of our knowledge is invaluable for building experience, and we hope future students will likewise appreciate it as an opportunity for professional experience. In taking this project semester seriously by properly planning time, requirements, and documentation future students can gain much more out of these 13 weeks. In the end we are satisfied to produce a cool project with some interesting capabilities. We believe it has a lot of potential to be something truly innovative in the future and we hope we can continue to develop it, and our own knowledge in the future.

Appendices

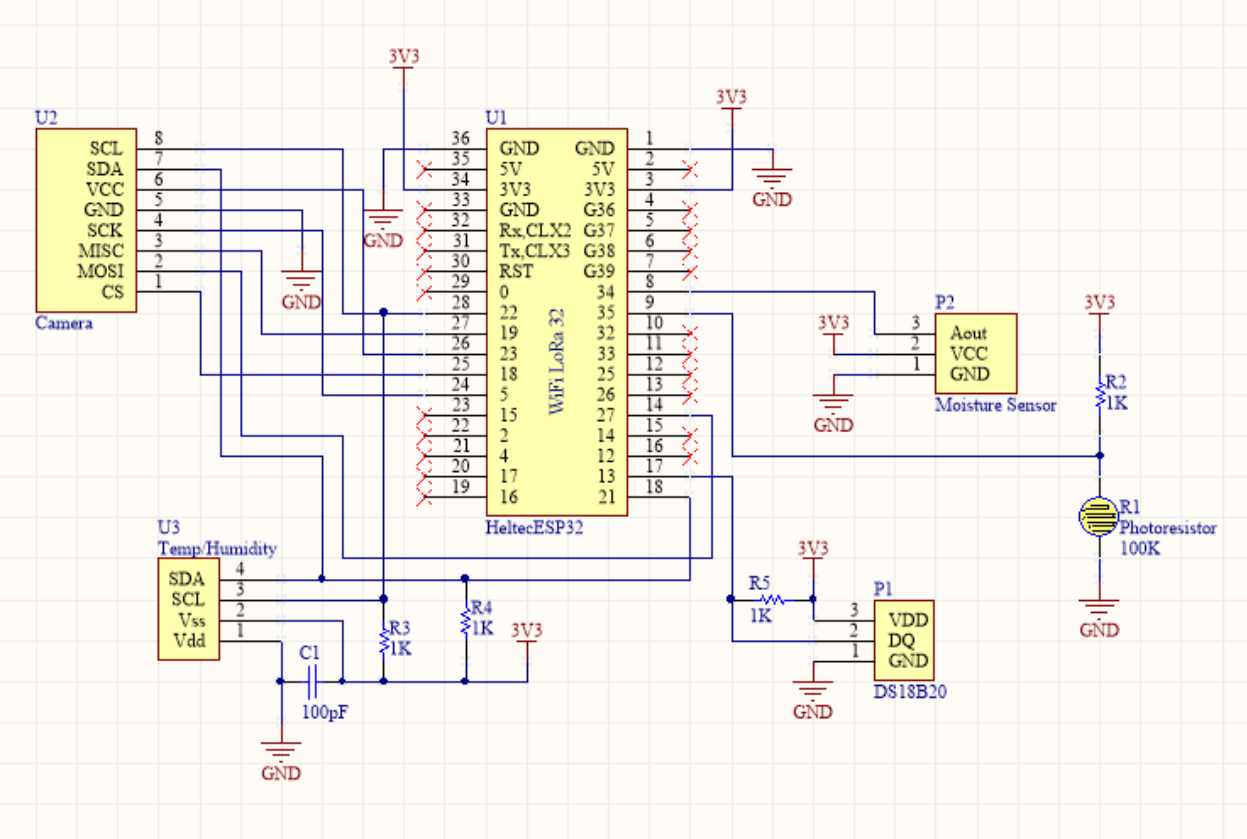
Appendix A: Budget

Item	Model	Supplier	Manufacturer	Price	Qty.	Total
Temp/Humidity	HIH7120-021-001CT-ND	Digi-Key	Honeywell	$15.70	2	$31.40
ESP32 controller	ESP32 OLED	Supplied	Heltec	$0.00	1	$0.00
Camera	2 Mpixel SPI Arducam	Amazon	Arducam	$56.59	1	$56.59
Relay Module	Kuman 4 channel DC 5V Relay	Amazon	Kuman	$10.99	1	$10.99
Moisture Sensor	Gikfun 2pcs Capacitive Soil Sensor	Amazon	Gikfun	$16.98	1	$16.98
Water Pump	Amarine-made Water Pump	Amazon	Amarine-made	$23.99	1	$23.99
Plant container	80 Liter Tote	Canadian Tire		$9.50	1	$9.50
Soil	18 Liter GV Potting Soil	Walmart	Great Value	$3.47	2	$6.94
Water Basin	10 Liter Can	Walmart	-	$5.97	1	$5.97
Lights	12V LED grow strip	Amazon	Ocamo	$12.96	1	$12.96
Fan	Repurposed power supply fan	Supplied	-	$0.00	1	$0.00
Photocel	-	Supplied	-	$0.00	3	$0.00
Water Temperature	DS18B20	Supplied	-	$0.00	2	$0.00
PCB	-	JLCPCB	JLCPCB	$2.00	1	$2.00
				Total:		**$168.35**

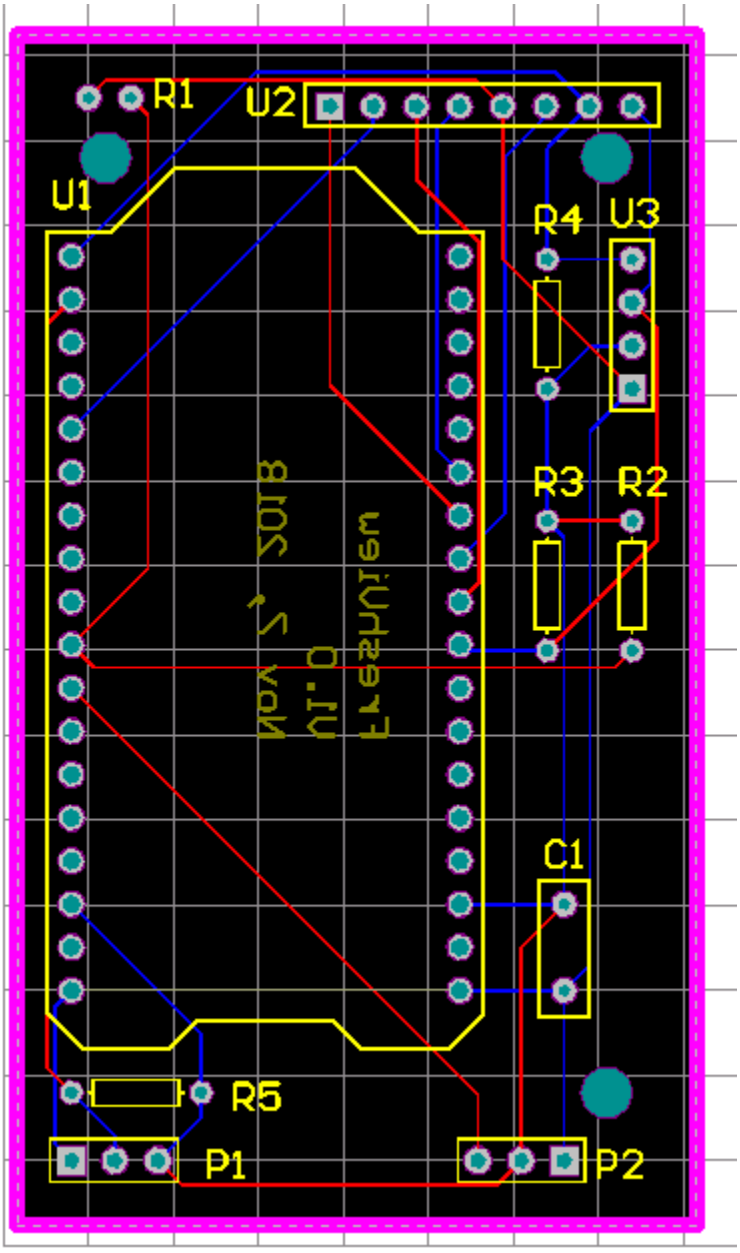
Appendix B: Programming Flow Chart



Appendix C: Circuit Schematic



Appendix D: PCB Layout



Appendix E: Gantt Chart

	Task Name	Durati
1	**Update Sensor**	**52 days**
2	Research sensor components	5 days
3	Confirm and order parts	10 days
4	**Program Control Board**	**50 days**
5	Control board testing	4 days
6	Camera	15 days
7	Data upload	5 days
8	Create Circuit Schematic	5 days
9	Order Circuit	6 days
10	Design Enclosure	4 days
11	Assemble Sensor	2 days
12	**Designing the web interface**	**26 days**
13	Program Database	10 days
14	Program Interface	16 days
15	**Control system**	**28 days**
16	Research control components	3 days
17	Confirm and order parts	5 days
18	Assemble Enclosure	5 days
19	System programming	10 days
20	System test	15 days
21	**Documentation**	**69 days**
22	Project Proposal	10 days
23	JLC PCB Reference Manual	5 days
24	Progress Report	5 days
25	User Manual/Datasheet	10 days
26	Final Report	10 days
27	Final Presentation	5 days
28	**End of Semester**	**0 days**