SENIOR DESIGN PROJECT FINAL REPORT EE/CpE 4097 – Fall 2019

Project title: Automated Gardening System

Team members:
Alan Paaren (ampcn7@mst.edu, EE)
Nicole Voss (nlvdkc@mst.edu, EE)
Samuel Ogunmolawa (saopr4@mst.edu, EE)
David Szatkowski (ddsp86@mst.edu, CpE/CS)
Mohamed Mohamed (mamt6f@mst.edu, EE)

Customer(s):
Local Growers
Crop Manufacturers

Advisor(s):
Dr. Ian Ferguson (Dean of the Southern Polytechnic College of Engineering and Engineering Technology, Advisor)

Dr. Chuanle Zhou (Assistant Research Professor, Advisor)

Estimated cost: $ 500.00

Instructor: Dr. Robert Woodley

Presented to the
Electrical & Computer Engineering Faculty of
Missouri University of Science and Technology

In partial fulfillment of the requirements for
SENIOR DESIGN PROJECT II (EE/CpE 4097)
Fall 2019

Table of Contents

Contents

Table of Figures

1 Introduction

1.1 Executive Summary

The *Automated Gardening System* is a garden that uses sound frequency to stimulate plant growth while using feedback from sensors to control the amount of water, nutrients, air circulation, and lighting in the plant's environment. Different plants require different environments to grow. The *Automated Gardening System* is unique compared to other devices by using frequency stimulation and different color LED light wavelengths to decrease crop growth time. The project is broken down into five sections: Frequency Stimulation, Watering System, Nutrient Dosing System, Air Circulation System, and Lighting System. The garden is designed to be made of modular systems to be able upscaled or downscaled in size. Each team member oversees a system while multiple team members are a part of multiple sections. The team member in charge was responsible for providing research on their topic as to what is the specifications needed to fill their sections requirements. All final decisions were approved by the team leader and the other team members involved in the section before progressing further into the project.

Figure 1: Automated Garden System Logo

1.2 Background and Problem Statement

Dr. Ian Ferguson provided the inspiration to design an automated garden. Upon discussing the customers' needs of an automated garden, the team determined that the system must be able to function entirely autonomously after the garden is set up by the user, and that it should be able to function as an enclosed environment to ensure usability in locations with hostile outdoor growing conditions. These needs are accommodated by the lighting, nutrient-dosing, watering, and air circulation systems. During the brainstorming process, the team decided to distinguish itself from similar systems by implementing an easily customizable ideal growing environment, using sound frequency to stimulate plant growth, and implementing a companion application to improve usability.

1.2.1 Existing Works

There are numerous devices currently existing today relating to automated gardening. Below is a non-exhaustive list of these devices:

FARMBOT

FarmBot is an outside automated garden that does everything for the user except harvesting the crop itself. [1] FarmBot plants seeds, measures soil moisture, waters, and even detects and destroys weeds. The user uses the company's application to plot out their garden almost like a video game, and FarmBot plants the crops using a grid. Once the crops are ready to harvest, FarmBot will send a notification.

PLANTY
Planty uses sensors to monitor soil humidity, temperature, and light, while providing updates on the user's smartphone. If Planty thinks the temperature is too chilly for you plant, it'll let the user know. [2] Planty also has a water reservoir to automatically water your plants. However, Planty is not designed for crops.

NECTAR
Nectar is loaded has sensors to set specific temperature, humidity, and pH balance to recreate various environments for plants. Nectar learns the best conditions for the user's plant by using the app-controlled box bases on the user's settings for their plant.

GROBO
Grobo is made up of fifty-three LED lights where the user may control the wavelength and spectrum of the lights. [2] The user can set up watering schedule to douse up to eight times a day. Grobo also has an application where the user can use the function "recipe" to control the lights, water, and nutrients from a mobile device.

CITYCROP
CityCrop is an indoor self-maintained pesticide-free garden that notifies the user when it's time to harvest. [3] CityCrop comes with climate control, a hydroponic system, a nutrient dosing system, LED lights, individualized plant settings, and live monitoring.

AEROGARDEN
AeroGarden is a soil free indoor garden that regulates water and nutrient delivery with overhead of LED lights.

Figure 2: AeroGarden Bounty Elite Wi-Fi—Source: Adapted from [4]

Out of the six different styles of automated gardening listed, only FarmBot works with a garden plot, but FarmBot is for outdoor gardens. All the other technologies only grow a personal number of crops. Our design will have bigger indoor garden plots with the same features as most of these other technologies offer such as: climate control, a hydroponic system, nutrient dosing system, and LED lights. In our design, we are going to use sound waves to stimulate more nutritious and healthy plant growth. [5] When a sound wave is moving through a plant, particles of the plant will be displaced both right and left as the sound energy passes. The motion of particles is parallel to the energy transport direction. We are assuming the effects from the audible sound field on plant growth because the energy transport is dependent on the sound frequency and intensity.

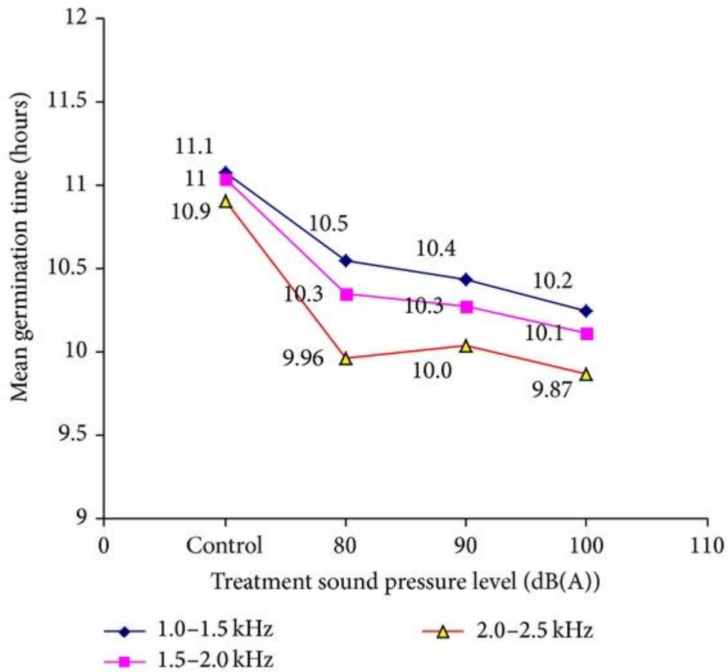

Figure 3: The Effect of Sound Pressure on Germination Time—Source: Adapted from [5]

There have also been studies on the vegetable and fruit plant growth with plant acoustic frequency technology (PAFT). [6] Using variable frequency of 0.08 kHz to 2 kHz and sound pressure of 50dB to 120dB increased the numbers of flowers and fruits, as well as the content of chlorophyll, the net photosynthetic rate, the photochemical efficiency of PSII (Fv/Fm) and non-photochemical quenching after 42 days of treatment in greenhouses.

1.2.2 Global and Societal Context and Motivation

After the industrial revolution and urbanization, there has been far-reaching effects on the environment. The clearing of large forests and farmlands to make room for cities along with the smog and air pollution that came from burning fossil fuels, all had a lasting impact on the environment. It’s estimated that in North America alone, 82% of people live in cities. A CNBC article projects that two-thirds of the world population is expected to live in cities by the year 2050. The global and societal impacts of gardening and green spaces is far reaching and becoming ever more central in today's politics. In the past few decades, society has been pushing to become ever “greener” and environmentally friendly. From more energy efficient appliances to even cutting back on our reliance to fossil fuels in exchange for renewable energy, there has been changes in all facets of society.

Despite the appeal of becoming more environmentally friendly, most people aren’t interested in investing the time and energy necessary to care for plants or gardens. Furthermore, they might lack the space or resources necessary for a functional garden. This is where our automated gardening system comes in. It allows people to be environmentally friendly without making significant sacrifices to other parts of their lives. Not only does our system save our user’s time, but it also improves their homes’ air quality by filtering the air around them. This can be extremely therapeutic and help reduce stress. A long-term study that followed nearly 3,000 older adults found that gardening is the single biggest risk reduction for dementia. For aspiring cooks, our automated system allows them to grow fresh vegetables and herbs that they can pick straight from the vine. Additionally, indoor gardening can be a great learning tool for people of all ages on the proper care of plants.

2 Design and Methods

The goals, tasks, and requirements mostly remained the same. We replaced the phone application with an LCD screen for user interface and integrated the nutrient dosing system within the water supply. Having the nutrient dosing being its own system seemed pointless since the nutrients need to be mixed with water anyways. Our two overarching goals are to *improve upon existing indoor growing systems* and to *maintain a user-oriented design process*. These goals have a myriad of tasks associated with them as detailed in the Goals and Tasks section below.

2.1 Goals

2.1.1 Goal 1: Improve Upon Existing Systems

1) **Task 1.1: Lighting System**

   Description: The lighting system used in the automated gardening system utilizes full-spectrum LED's that can provide optimum lighting at various stages of plant growth. The lighting system provides users with control of the wavelength, intensity, and duration of the LED's. The user sets the lighting schedule, wavelength, and intensity. Then the system operates independently using the values provided. In the event of power-loss, the lighting system has a built-in battery that keeps track of time and set-values.

   Challenges: The plant implemented in our senior design to record data was Mung Beans. One of the issues we ran into was that the room the project was held in had temperatures that occasionally dropped to 65 - 58 degrees. Since Mung Beans are warm climate plants, frost can have significant effects on their growth. One of the drawbacks of LED's is that they don't produce much heat compared to traditional grow lights.

2) **Task 1.2: Automated Nutrient Dosing**

   Description: Through user inputs and a database of optimal NPK ratios for a variety of different plants. We initially planned on using real time sensors to measure the soils nutritional content. Then using a mixing reservoir with a motor controlled stirring device to dispense the correct ratio of NPK that the plants needed. As the team examined the design more carefully and reviewed its viability. We came to realize that many of these features ran beyond our scope. The final design used the optimal NPK ratio for mung beans (1 part Nitrogen, ½ part Phosphorus, and 1.7 parts Potassium) that was recommended from previous research done on the plants and manually mixed this into the reservoir.

   Challenges: During Senior design I, while the team was doing research on the design, we realized that real-time nutrient dosing was a lot more complicated than we originally expected. The first challenge was measuring the soil's nutrient content. This is a very complicated and expensive task that is nearly impossible to automate. In order to overcome this obstacle, we opted for periodically dispensing the nutrients using the manufacturer's recommended dosage. During the prototype stage, the team was implementing this manually. As the semester progressed, the team also came to realize that the mixing reservoir would have added another layer of complexity that we lacked the time and resources to complete. The team opted for mixing a diluted dosage that coincided with the manufacturers recommended dosage directly into the watering system.

3) **Task 1.3: Automated Watering System**

Description: The watering system is controlled by a moisture sensor which is placed into the soil. The moisture sensor is wired and programmed into the Arduino and reads the current moisture value. The watering system is on a timing mechanism. Depending on how low the value of moisture when the timer goes off, will determine when the water will be dispensed through the plants. The water is stored in a seven-gallon reservoir. The reservoir has tubing flowing in and outside connected to solenoid valve. The Arduino communicates with the solenoid valve when to be active.

Challenges: Being able to evenly distribute the water too all the plants without blocking the grow light. The pipe has one flow in, so the first box will get water first. To counter act the first box from receiving more water than the rest of the plants. The first box of plants only has two holes of water dispersing from the pipe rather than three like the other boxes.

4) **Task 1.4: Frequency Stimulation**

Description: In order to produce the plant frequency stimulator, we connected the Arduino Mega to a speaker to produce multiple variations of frequency sound waves. The Arduino Mega directly connects into an amplifier that alternates the sound intensity produced. The frequency is currently only able to be changed within the Arduino Mega code.

Challenges: One of the initial challenges we faced was determining if we wanted to produce the frequency through a circuit or through code in the Arduino Mega. Many of the studies we had retrieved described a desired noise distribution of ~100dB and the noise distribution perpetrating off the Arduino Mega was ~90dB. We needed to determine potential amplifiers to increase the noise distribution from the speaker. A risk with using frequency stimulation in the system is that the noise may be disturbing. We have used Styrofoam to help insulate the sound, but also have the frequency on when no one is around to hear it.

5) **Task 1.5: Air Circulation System**

Description: The air circulation system consists of a carbon air filter and pump located at the top of the tent and a 5-inch axial fan mounted at the base of the garden. The carbon pump operates independently from the other process and can be control via a switch, where the user can increase the power of the pump. The axial fan is controlled by the Arduino Mega and turned on based on the values from a DHT22 temperature and humidity sensor. The fan was programmed to turn on for 10 minutes every hour or when the temperature or humidity in the tent was too high. The system is designed to provide proper air ventilation, circulate air throughout the tent, and maintain an optimal temperature for the plants to thrive in. Additionally, the air circulation system prevents mold from accumulating, reduces heat impact, and helps strengthen the plants' stems by simulating the effects of wind.

Challenges: A challenge we faced while initially implementing our air circulation system was that our DHT22 sensor was not operational. As a result, we had to order additional sensors and waited about two weeks until the sensors came in. During this time, we were unable to use the circulation fan and only had the carbon air pump running in the tent.

6) **Task 1.6: LCD User Interface Screen**
Description: The user interface displays the clock and allows the user to adjust the clock using a 4-button array. The clock display is an important feature which allows the user to know more exactly when the alarms will occur. Allowing the user to adjust the time via the button array makes it very easy for the user to troubleshoot any timing related issues. For example, when the system loses power, from a power outage or other such issue, the clock resets, along with all the alarms. The clock display makes such an event easy to notice, and quick to fix.
Challenges: Learning how to read button inputs and print to the LCD screen was challenging. Additionally, reading button inputs required checking for button presses every fraction of a second. This caused some conflicts with several other systems already in the Arduino code and required those sections of code to be rewritten. Lastly, the team intended to attempt to control more parameters of the system through the LCD screen in the form of a menu but did not have time to finish developing this feature.

2.1.2 Goal 2: User Focused Design Process

1) **Task 2.1: Research Consumer Needs**
Description: While designing this product, the pains and gains of the consumer were placed at the forefront. Team members researched and brainstormed what features would benefit the consumer and interviewed potential customers throughout the design process to gain feedback and apply it to new prototypes.
Challenges: Research of consumer needs uncovered new pains and gains that team had not considered leading to large design changes in the automated garden. Throughout the semester our initial ambitions did not work as planned. We had to adapt to make things work effectively and easily for the consumer.

2) **Task 2.2: Iterative Testing**
Description: Testing was done constantly throughout the building process. Before assembling the prototype, we performed a cognitive walkthrough of features to evaluate key aspects such as usability, learnability, and effectiveness. We built modular prototypes, attempting to keep each feature of the prototype separate so that we can easily update and adapt each feature. Parts of the testing involved potential consumers so that we could maintain a focus on the needs and wants of the consumer.

Challenges: Numerous trial and errors occurred. The initial design for the watering system was a “T” shape configuration. However, the team’s advisor recommended creating independent sets of plants that we could remove from the tent in order to test the effects of frequency stimulation. Creating separate sets of plants introduced gaps in-between boxes that caused water to leak outside the system. The watering system was eventually designed to run between the plants rather than alongside them.

3) **Task 2.3: Track Changes**

Description: When creating our prototypes, we made note of each new iteration. This let us note in detail the problems we encountered, so we could fix and improve upon. The goal of tracking the changes was eventually to resolve all the issues we encountered to create a product that was effective and efficient.

Challenges: A lot of the improvements upon a prototype were recycled from the previous prototype due to working with a budget. As much as it would’ve been beneficial to create a new prototype from scratch, this wasn’t feasible. Before deconstruction of the previous prototype, it was documented to be reviewed later if needed.

4) **Task 2.4: Cost**

Description: The team kept the budget as low as possible by simplifying the construction or using relatively affordable items that could be purchased at your local hardware store. The team also did not want to go over the top and include things that would add unnecessary features and increase the cost of the system. We focused on the primary consumer needs and focused on those. The main things plants need to survive are light, water, and a suitable climate. Our build focused on achieving those factors at minimum cost to ensure plants could grow.

Challenges: Keeping the cost down did not guarantee the quality of components were satisfactorily. The problem with cheap electronic components is they wither fry easily or do not last long. We only had one component fry during assembly, but that was most likely human error. Another problem with cheap electronics is accuracy. Our soil moisture sensors fluctuate in value. Knowing the precision of the soil moisture is inconsistent as the number could read 600 or 150.

5) **Task 2.5: Reliability**

Description: For an automated garden, it is important that upkeep of the garden is entirely autonomous, and the autonomous functions must be entirely reliable. If any part of the automated garden fails to work with 100% reliability, it defeats the purpose of owning a garden that can take care of itself. The automated garden produced turned out to be very reliable. Except of a few component issues and bugs in our code, every system ran consistently since the integration phase of the design.

Challenges: Having a high standard on reliability requires thorough testing of the final product to ensure that every aspect of the automated garden functions correctly. Extensive testing within the scope of this course carries the risk of not having enough time to test and implement solutions. To combat this risk, the team allocated a significant chunk of time for testing in the design schedule.

2.2 Deliverables

The Automated Gardening System will come as a kit varying in size based on the customers purchase. Personal gardens will come as kits for the user to assemble themselves; an additional fee will have the kit assembled on site which is suggested if application is for industrial size. The initial system package will include an Arduino Mega as the operating system, LED lights, fans, carbon air filter pump, water repository, PVC piping, speaker, and materials for the garden frame. Number of components may vary depending on size of garden plot and overhead space.

2.3 Specifications and Requirements

ID	Title	Note	Verification
Requirement 1: Frame construction	The frame provides structure for the entire system. The depth can be changed easily, and the frame's planting area can be scaled up to accommodate larger gardens.	The frame must be scalable from a volume of a cubic meter to as large as 10 cubic meters. The depth should range from at least 10 cm to 50 cm.	Demonstration of a scaled-up version of the frame that can support a scaled-up version of the system.
Requirement 2: Water Transportation	A water reservoir dispenses a regulated amount of water through piping based on the soil sensors. The piping is easily scaled for larger gardens.	The piping must be scalable for large-scale systems. It must be capable of evenly dispersing adjustable amounts of water and nutrients. The reservoir must be capable of storing enough water to sustain the system for a minimum of a month.	Demonstration of Water Transportation System within a large-scale system.
Requirement 3: Nutrient Dispenser	Nutrients will be dispensed into the water reservoir based on feedback from the soil sensors.	The nutrient system must be capable of storing enough nutrients to sustain the system for a minimum of a month.	Demonstration of Nutrient Dispenser System within a large-scale system.
Requirement 4: Lighting	LEDs supporting a wide range of frequencies will allow for varied lighting conditions.	The lighting must be capable of 380nm - 780nm wavelength on the spectrum at ~1000 watts.	Use lighting that fits the desired specifications.

Requirement 5: Air Circulation	Fans provide light air circulation to promote growth and prevent stagnation.	The system must be able to provide consistent air circulation and be scalable with the overall system.	Demonstration of Air Circulation System within a large-scale system.
Requirement 6: Frequency Stimulation	Speakers will play varying frequencies to stimulate growth and hasten germination times.	Speakers must be water resistant and capable of producing up to 80 - 120 dB and frequencies ranging from 1,000 to 20,000 Hz.	Use speakers that can handle the desired specifications.
Requirement 7: Arduino Mega Configuration	The Arduino Mega serves as the logic unit for the system, gaining information from sensors and controlling the water reservoir and speakers.	The Arduino Uno must be capable of handling the logic for the entire system and communicate with the application via Bluetooth.	Iterative testing to confirm the system is functioning as intended.
Requirement 8: User Interface	Displays current status/values of lighting system, water system, frequency device, and temperature.	Easy-to-use interface for the user to remotely control growing conditions.	Usability and Accessibility Testing will be performed, and the application will be updated accordingly.

Table 1: Automated Gardening System Requirements Matrix

3 Technical Approach and Results

3.1 Final Design

The final design of the garden included the sound frequency system, watering system, air circulation system, lighting system, and a user interface. All the systems, except the lighting system, communicated and were controlled by the Arduino Mega. The code of the systems are broken down into each individual system with a description, but an overall code print out can be found in Appendix A.

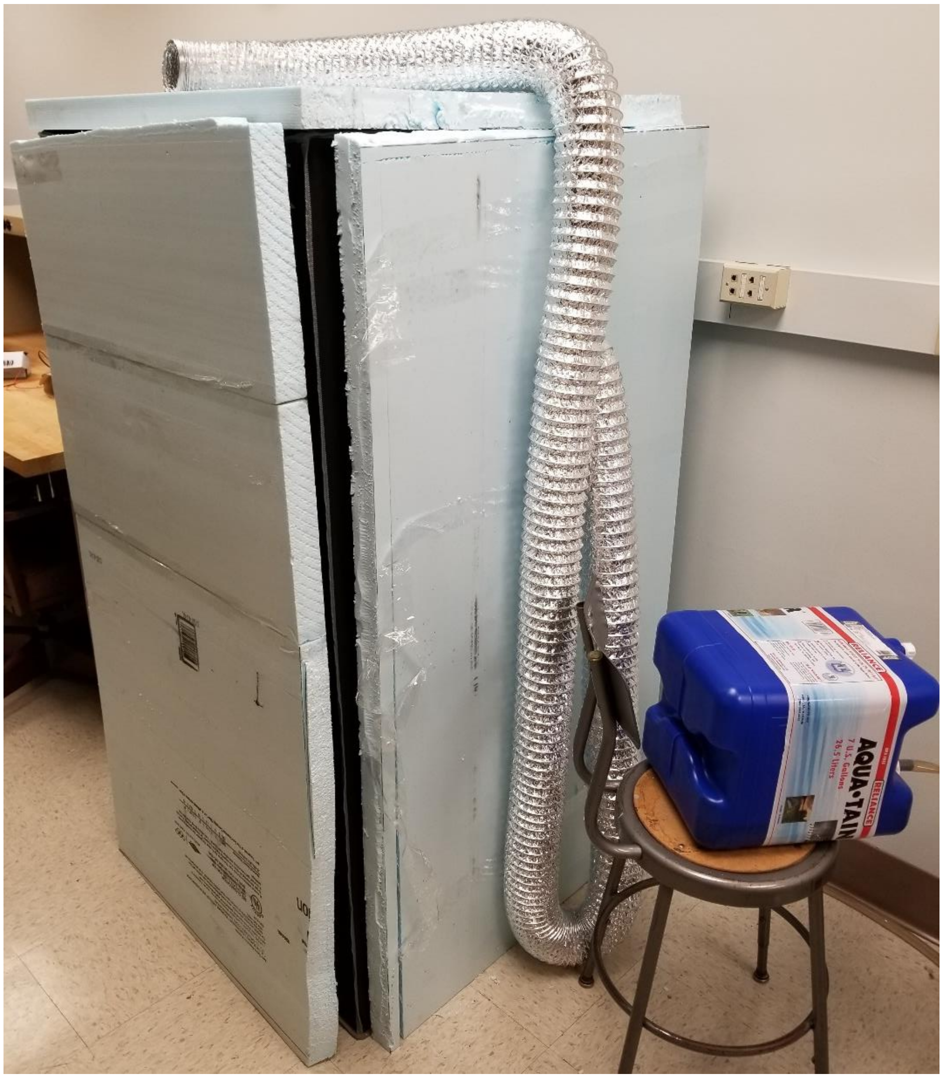

Figure 4: The Automated Gardening System

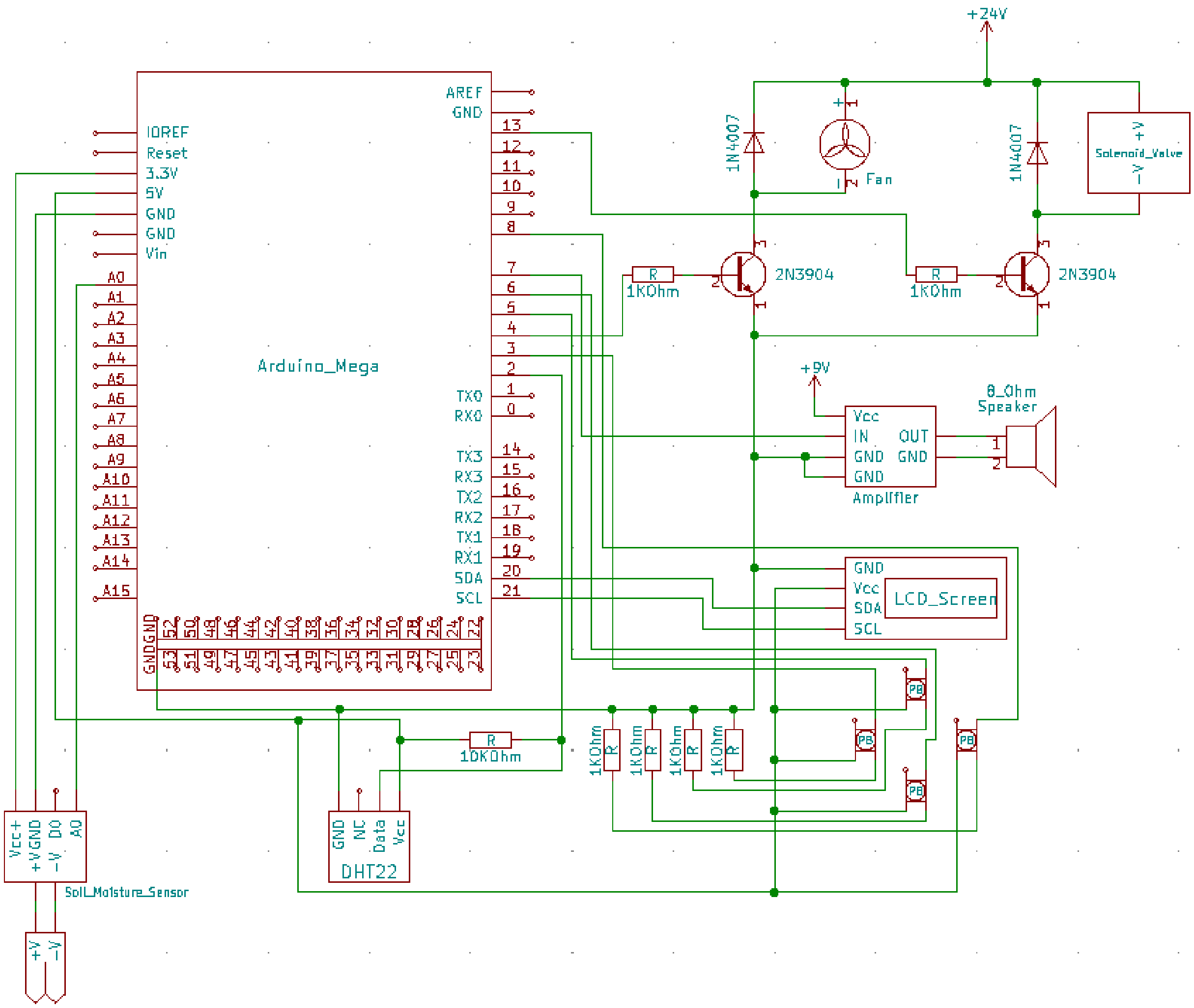

Figure 5: Automated Gardening System Circuit Diagram

This is the entire circuit design for the *Automated Gardening System*. To demonstrate the simplicity our modular design, we have broken down the description for each circuit per each individual system.

3.1.1 Watering System

The watering system was designed to provide precise mounts of water to plants without interfering with the grow light above. The initial plan was to have the watering system underground in the soil, but we did not have enough pressure to have the water seep deep enough into the soil. The only option was to have the watering system run above ground. Our biggest concern was blocking the grow light from the plants. We designed the watering system to run alongside the plants, allowing them to grow around the watering system. The watering system was designed to be able to accommodate for all types of plants by allowing the user to be able to set how frequent the watering system would run and for how long.

Figure 6: Watering Plants Close-up

Figure 4 shows the PVC feed to the plants releasing water to the plants. Note the water is not heavily pressurized. Having the water gently flow out allows the plants to not be damaged.

Figure 7: Watering System Soil Moisture Sensor

This is the moisture that resides in the soil of the plants box. The sensor sends feedback to the Arduino of the current state of the plants soil.

Block Diagram:

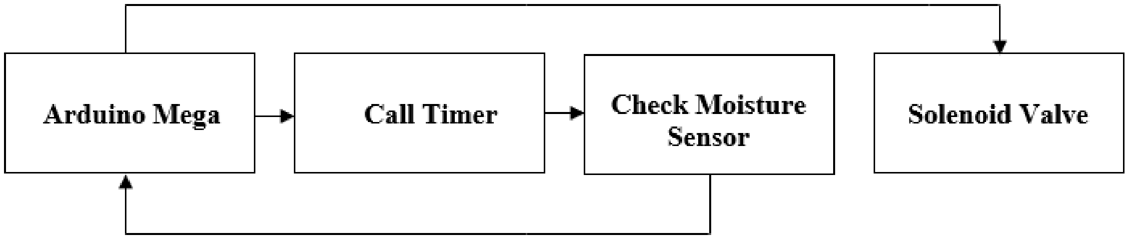

Figure 8: Watering System Block Diagram

The Arduino Mega is utilized in creating an internal timer to open the solenoid valve for a desired time frame at how ever often desired. When the timer from the Arduino Mega goes off, it checks the soil moisture sensor to see if the plants need watering. The value of the soil moisture sensor is set by the user to accommodate a variety of plants. The soil moisture sensor sends a feedback value to the Arduino Mega for the Arduino Mega to determine whether to open the solenoid valve.

Circuit Diagram:

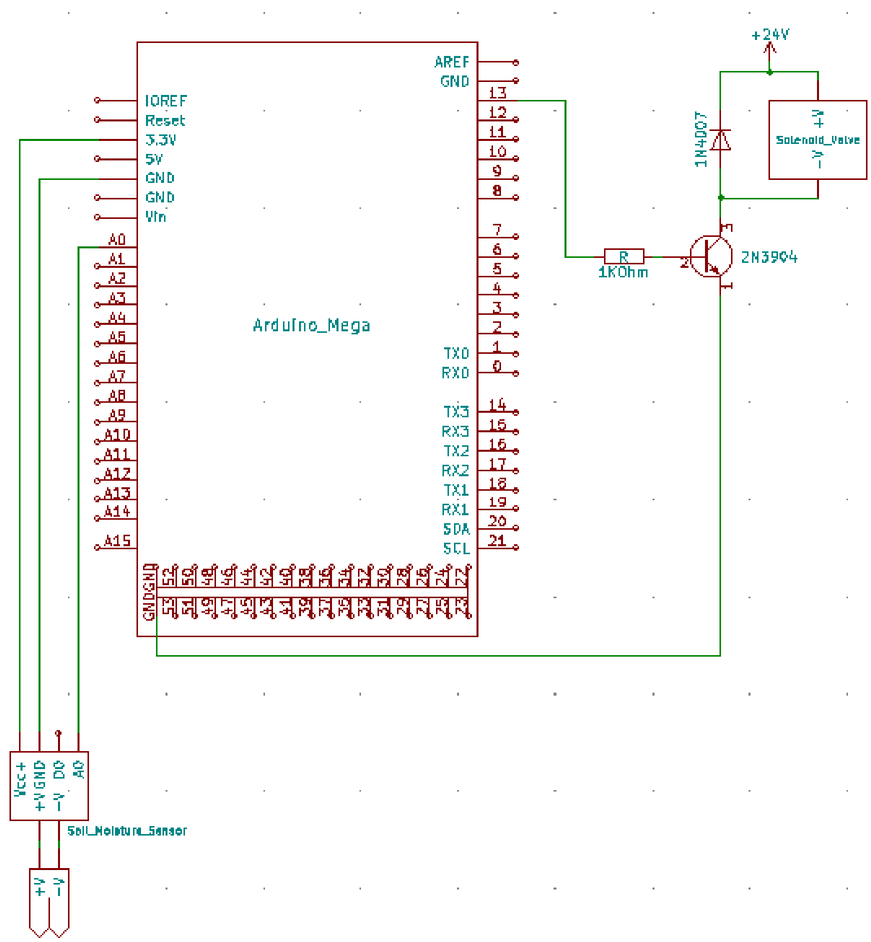

Figure 9: Watering System Circuit Diagram

The watering system is constructed of an Arduino Mega, a soil moisture sensor, a 1KOhm resistor, an NPN transistor, a diode, a solenoid valve, and a 24V DC power supply. The Arduino Mega can't power the solenoid valve because it does not provide enough power, but it can be used as a timer. An NPN transistor acts as a switch to power the solenoid valve when the Arduino powers the pin connected to the base of the transistor.

Software Description:

The Arduino code is a timing mechanism that reads valves sent from the soil moisture sensor from the assigned analog pin. There is an internal clock inside the Arduino that lets the Arduino have a time for reference, so it knows when to activate timers. When the time is 10am, the timer turns the WaterOn routine is activated it checks what day of the week it is to know if the plants should be watered during that day. The soil moisture sensor is also checked to determine if the plants need water. This is a double check system to ensure the plants will not be over watered. After the routine finishes, it increases the day count for the WaterOn routine.

```
#include <TimeAlarms.h>
#include <Time.h>
#include <TimeLib.h>

int sensorPin = A0; //moisture sensor
int sensorValue;  //moisture sensor
int limit = 600; //moisture sensor

void setup() {
  setTime(16,0,0,10,30,19);//set time for 4pm October 30th, 2019
  Alarm.alarmRepeat(10,0,0,WaterOn);  // 10:00am every day
  Serial.begin(9600);
}

//Timer for watering system
void WaterOn() {
  Serial.println("daycount: ");
  Serial.println(daycount);
  if (daycount == 1 or 4) {
    if (sensorValue>limit) {
      Serial.println("Alarm: - its Raining!");
      digitalWrite(13, HIGH);
      delay(30000);
      digitalWrite(13, LOW);
    }
    else {
      Serial.println("Plants have enough water");
    }
  }
  daycount = (daycount + 1);
  if (daycount > 7){
  daycount = 1;
  }
  Serial.println("daycount: ");
  Serial.println(daycount);
}
```

Figure 10: Watering System Program Code

3.1.2 Air Circulation System

The air circulation system was designed to maintain a suitable temperature and humidity level in the tent. In the original design for the axial fan, it was going to be placed above the plants and blow air downward on the plants, however there was not a suitable place in the tent to mount it. Instead a small, wooden structure was made for the fan and mounted to the side of the garden so the fan could blow air directly next to the plants.

Figure 11: Axial Fan

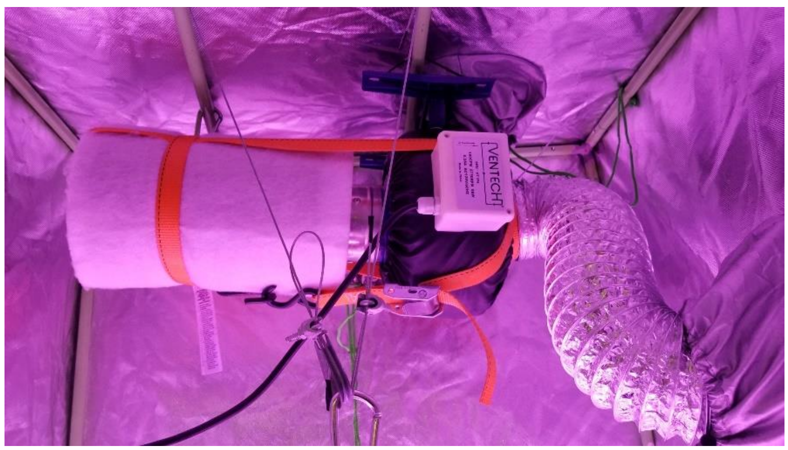

Figure 12: Carbon Air Filter and Pump

A carbon air filter pump was installed on the top of the tent. This allowed for the air inside the tent to be sucked out of the tent to help remove temperature and humidity.

<u>Block Diagram:</u>

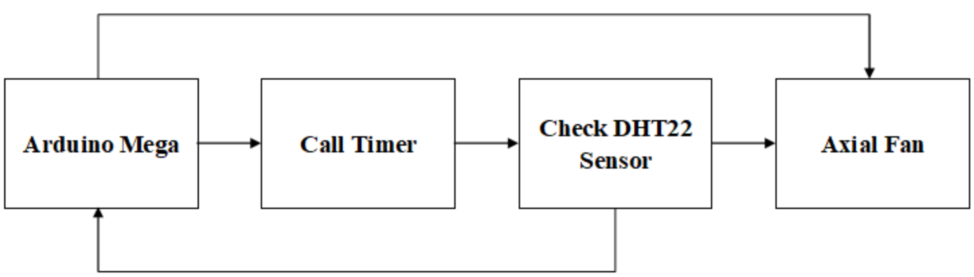

Figure 13: Air Circulation System Block Diagram

The air circulation system is controlled by the Arduino Mega which controls the fan in two aspects. The fan is turned on by the Arduino once every hour for ten minutes by a timer programed into the Arduino. The Arduino also checks the DHT22 sensor every minute to see if the temperature and humidity are not exceeding the limit set for the temperature or humidity. If the limits are exceeded, the Arduino turns on the fan until those values are below those limits.

Circuit Diagram:

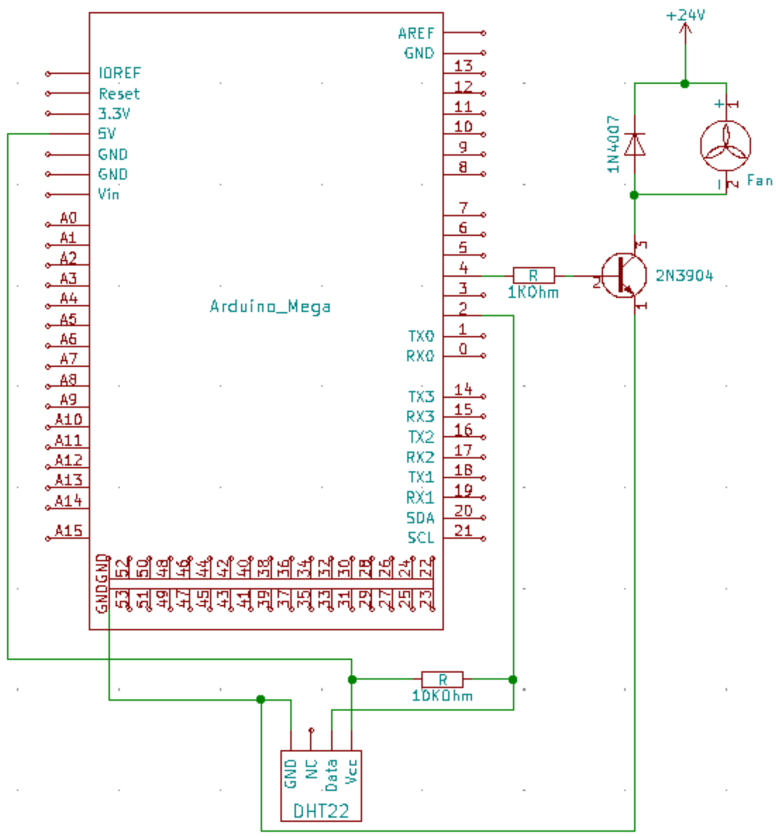

Figure 14: Air Circulation System Circuit Diagram

The air circulation system is constructed of an Arduino Mega, a DHT22 temperature and humidity sensor, a 1KOhm resistor, an NPN transistor, a 1N4007 diode, a 24V DC power supply, and an axial fan.

Software Description:

```
#include <TimeAlarms.h>
#include <Time.h>
#include <TimeLib.h>
#include <DHT.h>
#define DHTPIN 2     // what pin we're connected to
#define DHTTYPE DHT22   // DHT 22  (AM2302)
#define fan 4
int fanCount = 11;
int maxHum = 60;
int maxTemp = 74;
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  setTime(16,0,0,11,21,19);//set time for 4pm October 30th, 2019
  Alarm.timerRepeat(3600, FanOn);    //Repeats every hour 3600 secs
  Alarm.timerRepeat(60, FanOff); //Checks every minute if X minutes have passed since FanOn
  Alarm.timerRepeat(60, PrintLoop);
  pinMode(fan, OUTPUT);
  Serial.begin(9600);
  dht.begin();
}
void PrintLoop()
{
  sensorValue = analogRead(sensorPin);
  Serial.println("Analog Value : ");
  Serial.println(sensorValue);
  float h = dht.readHumidity();
  // Read temperature as Celsius
  float t = dht.readTemperature();
  float TempF = (t*1.8)+32;
  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
  if (h > maxHum || TempF > maxTemp) {
      digitalWrite(fan, HIGH);
      Serial.print("Turning on Fan it's hot! ");
  } else if(fanCount > 10) {
        digitalWrite(fan, LOW);
  }
  Serial.print("Day count: ");
  Serial.print(daycount);
  Serial.print(" \t");
  Serial.print("Humidity: ");
  Serial.print(h);
  Serial.print(" %\t");
  Serial.print("Temperature: ");
  Serial.print(TempF);
  Serial.println(" *F ");
}
```

Figure 15: Air Circulation System Program Code

The air circulation system uses a repeating timer to handle the fan. The function FanOn is called every hour and turns the fan. FanOff is called every minute to check how long the fan has been on for. The DHT22 sensor values are checked every minute. If the temperature or humidity exceed the limit, the fan will be turned on until those values are back below the margin.

```
void FanOn() {
  float h = dht.readHumidity();
  // Read temperature as Celsius
  float t = dht.readTemperature();
  float TempF = (t*1.8)+32;
  Serial.print("Turning on Fan");
  digitalWrite(fan, HIGH);

  fanCount = 0;
}

void FanOff(){
  if(fanCount == 10){
    digitalWrite(fan, LOW);
    Serial.print("Fan Turned Off");
  }
  if(fanCount <= 11)
  {
    Serial.print("Fan on for ");
    Serial.print(fanCount);
    Serial.print("minutes.");
    fanCount++;
  }
```

Figure 16: Air Circulation System Program Code

The function FanOn converts the temperature into Fahrenheit and turns the fan on. FanOff counts the minutes the fan has been on for consecutively and turns the fan off after ten minutes have passed.

3.1.3 Sound Frequency System

The design process employed to develop the sound frequency system for the project includes identifying the needs & constraints, researching specifics, develop & create possible solutions, selecting a solution, building a prototype, evaluate and redesign, and finalize the process design. The needs & constraints that this portion of the project required included variable frequency distribution ranging from at least 1 kHz to 20 kHz and produce 80-120dB of noise. This range was desired for the system to be adjustable for various types of plants. There were multiple possible options to implement a variable frequency generator. We had to decide against a hardware and software option but settled on the software implementation to reduce the amount of wires within the overall system. After setting up the circuitry from the Arduino Mega to the speaker we determined that the noise produced of ~90dB was below our desired value. We added in a 5V to 9V converter to connect to a power amplifier that amplifies the signal from the Arduino Mega to the speaker. We furthered the project through connection to a user interface to adjust the time the frequency is implemented and options to select a frequency.

Figure 17: Sound Frequency System

Block Diagram:

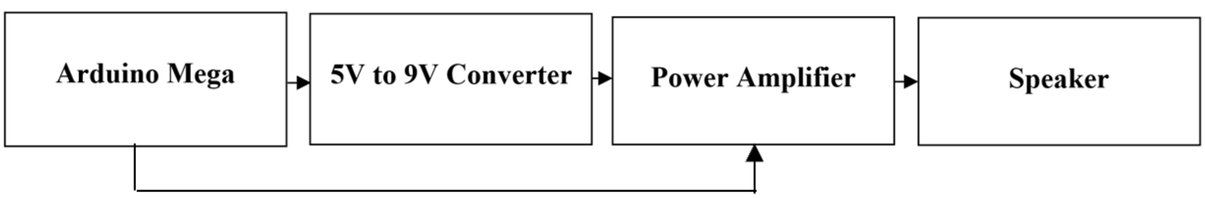

Figure 18: Sound Frequency System Block Diagram

The Arduino Mega is utilized to produce a set frequency at a set time for a pre-selected duration as well as provide power to the power amplifier. The 5V to 9V converter is utilized to convert to voltage from the Arduino Mega to power the power amplifier to increase the amplitude of the frequency coming from the Arduino Mega. This signal is then sent to the speaker to articulate the frequency audibly.

Circuit Diagram:

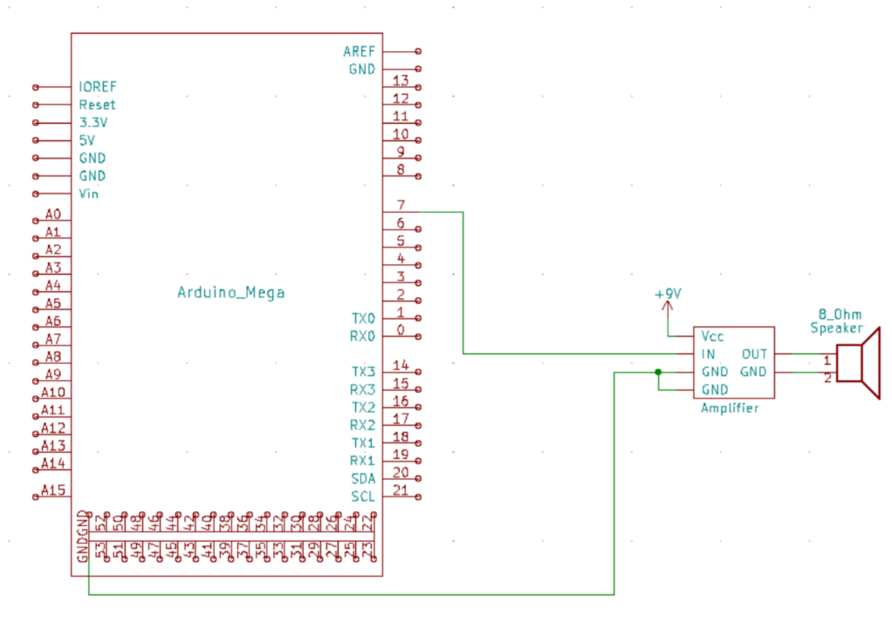

Figure 19: Sound Frequency System Circuit Diagram

<u>Software Description:</u>

```
 const int freq = 2700;
 setTime(16,0,0,10,30,19);//set time for 4pm October 30th, 2019
  Alarm.alarmRepeat(13,0,0,FreqOn);
  Alarm.alarmRepeat(13,0,30,FreqOff);  // 4:00pm every day
  pinMode(spkr, OUTPUT);

//Timer for speaker
void FreqOn() {
  // write here the task to perform every morning
  Serial.println("Sound on");

  tone(spkr, freq);
}

void FreqOff() {
  //write here the task to perform every evening
  Serial.println("Sound off");
  noTone(spkr);
}
```

Figure 20: Sound Frequency System Program Code

The frequency is set as an integer. The code then sets a time so that the system's alarms have a set numerical value to compare to. An alarm is then set to call the frequency every morning at the same every day for a set amount of time and display "sound on". Utilizing the tone function, we were able to produce the frequency we desired to be sent to the speaker. The function FreqOff was set to ensure that the tone function was not on when it was called.

3.1.4 Lighting System

The lighting system was a separate system from the other system. The LED light itself came with its own power cord and internal timing mechanism. The lighting system provided full-spectrum LEDs with spectrum control to provide an optimum wavelength of light throughout the entire growth cycle. The light was programmable to accommodate for different plants growth cycles. The lighting system was controlled by a remote control. The team was in working on an infrared communication device to be able to control all the system from one device, but the team ran out of time and budget.

Figure 21: Lighting System LED Array

3.1.5 User Interface

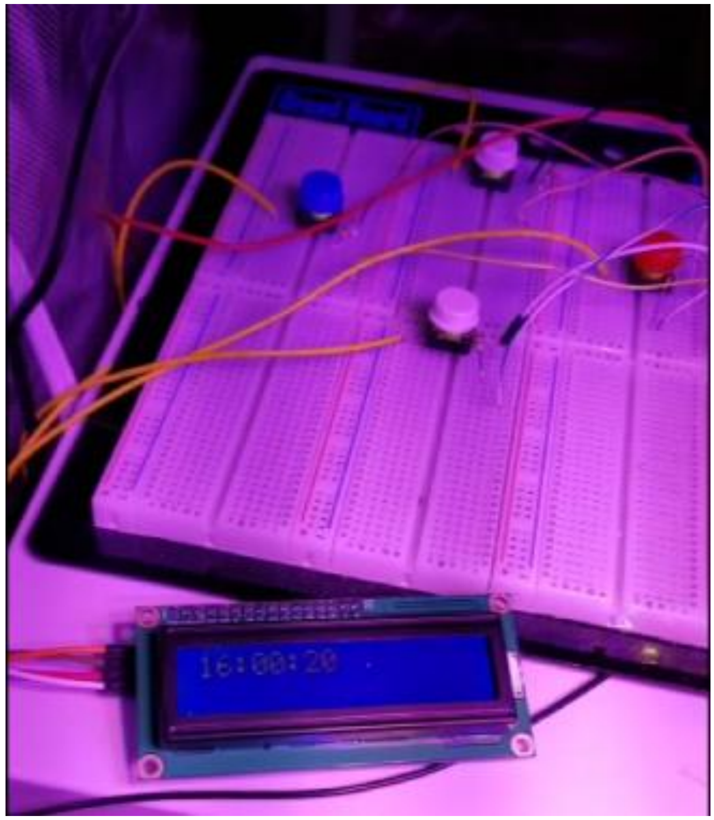

Figure 22: User Interface

The user interface is compiled of an LCD screen and four toggle pushbuttons. This displays the clock and allows the user to set the time. This is done by using the buttons to control a cursor and increment or decrement the digit highlighted by the cursor.

<u>Block Diagram:</u>

Figure 23: User Interface Block Diagram

The block diagram for the user interface is very simple. The user pushes a button in the array which is registered by the Arduino. The Arduino then visually changes what's displayed on the LCD screen.

Circuit Diagram:

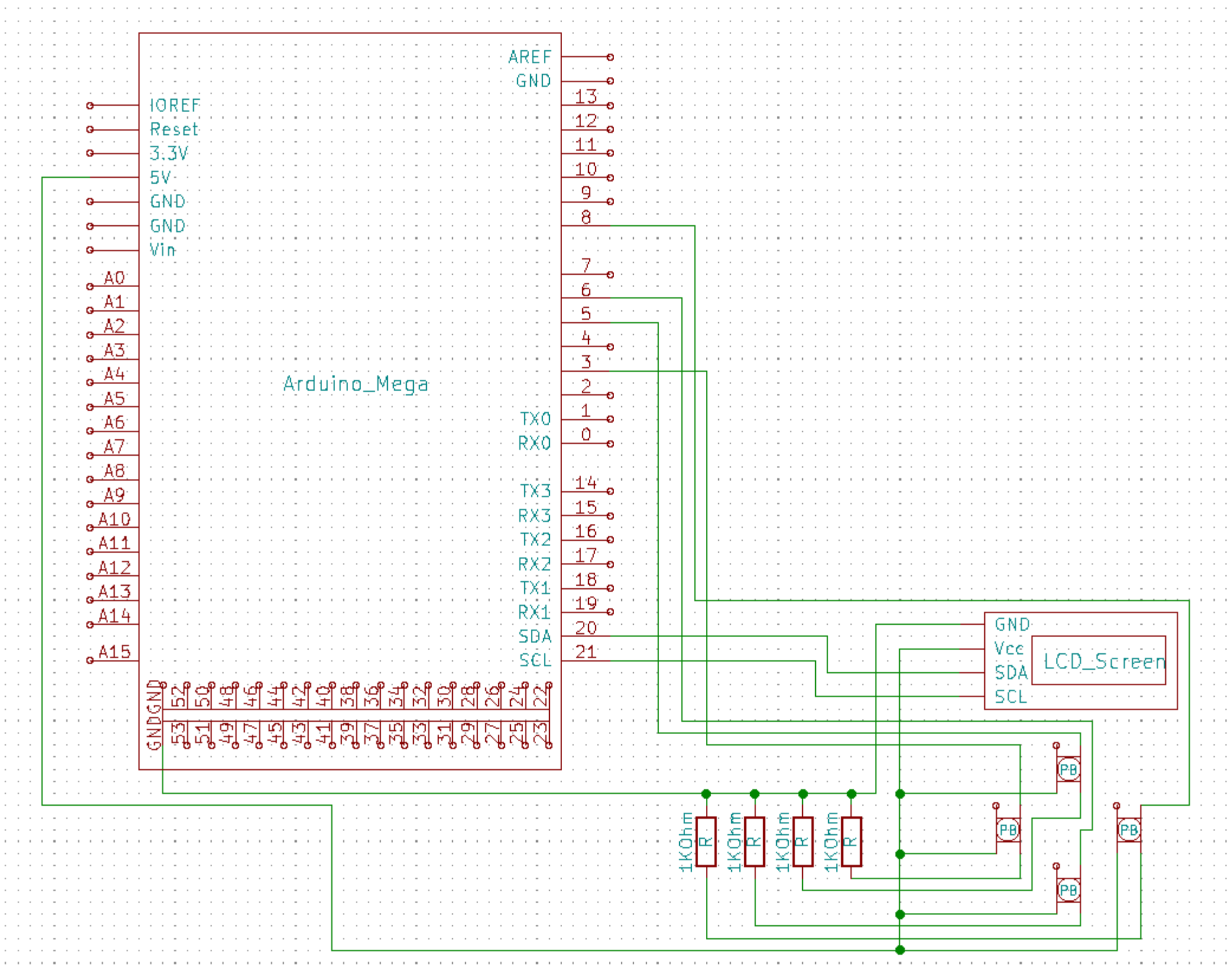

Figure 24: User Interface Circuit Diagram

The user interface was constructed with four push buttons, four 1KOhm resistors, an LCD screen, and the Arduino Mega. The left and right pushbuttons allowed the user to select which digit is highlighted, and the top and bottom push buttons let the user increase or decrease the time.

Software Description:

```
#include <Wire.h>
#include <Streaming.h>
#include <DS1307RTC.h>
#include <TimeAlarms.h>
#include <Time.h>
#include <TimeLib.h>
#include <LiquidCrystal_I2C.h>

// Set the LCD address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(0x27, 16, 2);
const int rightButton = 8;     // the number of the pushbutton pin
const int leftButton = 3;     // the number of the pushbutton pin
const int upButton = 5;     // the number of the pushbutton pin
const int downButton = 6;     // the number of the pushbutton pin
int buttonStateRight = 0;         // variable for reading the pushbutton status
int buttonStateLeft = 0;
int buttonStateUp = 0;
int buttonStateDown = 0;
bool right = 0;
bool left = 0;
bool up = 0;
bool down = 0;
int hold = 0;
int clockCursor = -1;

void setup() {
  lcd.init();
  lcd.backlight();
  setTime(16,0,0,10,30,19);//set time for 4pm October 30th, 2019
  Alarm.timerRepeat(60, PrintLoop); //Prints every min
  pinMode(rightButton, INPUT);
  pinMode(leftButton, INPUT);
  pinMode(upButton, INPUT);
  pinMode(downButton, INPUT);
  Serial.begin(9600);
  dht.begin();
}

void loop() {
  ButtonCheck();
  Alarm.delay(100);
}
```

Figure 25: User Interface Program Code

When the program initially starts up, 4 pins are assigned to the buttons and several variables are initialized. The clock cursor starts at position -1, indicating that it is not initially shown on screen. The function ButtonCheck is ran every 100 milliseconds to see if the user has pressed a button.

```
void ButtonCheck(){
  buttonStateRight = digitalRead(rightButton);
  buttonStateLeft = digitalRead(leftButton);
  buttonStateUp = digitalRead(upButton);
  buttonStateDown = digitalRead(downButton);
  if (buttonStateRight == HIGH) {
    RightButton();
  } else if (buttonStateRight == LOW) {
    right = 0;
  }
  if (buttonStateLeft == HIGH) {
    LeftButton();
  } else {
    left = 0;
  }
  if (buttonStateUp == HIGH) {
    UpButton();
  } else {
    up = 0;
  }
  if (buttonStateDown == HIGH) {
      DownButton();
  } else {
    down = 0;
  }
  digitalClockDisplay();
}
void RightButton(){
    if (right == 0) {
      if(clockCursor < 7){
        clockCursor++;
        if(clockCursor == 2 or clockCursor == 5)
          clockCursor++;
      }
      else
        clockCursor = -1;
    }
    right = 1;
}
void LeftButton(){
    if (left == 0) {
      if(clockCursor > -1){
        clockCursor--;
        if(clockCursor == 2 or clockCursor == 5)
          clockCursor--;
      }
      else
        clockCursor = 7;
    }
    left = 1;
}
```

Figure 26: Unser Interface Program Code

The function ButtonCheck when called checks each button to see if it is being pressed. If so, the button calls a function tied to the button pressed called UpButton, DownButton, LeftButton, or RightButton. The variables up, down, left and right are used to ensure that only one button press registers. When the respective button is pressed, they allow the function to perform, then are set to 1. While they are set to 1, the function cannot perform again. They are only reset to 0 if ButtonCheck is called while the button is released. LeftButton and RightButton simply toggle the cursor left and right by incrementing or decrementing the clockCursor value. The clockCursor variable skips values that represent the colons in the clock display and wraps around if the value goes to high or low.

```
void UpButton(){
  if (up == 0) {
    if(clockCursor == 0)
      adjustTime(36000);
    else if(clockCursor == 1)
      adjustTime(3600);
    else if(clockCursor == 3)
      adjustTime(600);
    else if(clockCursor == 4)
      adjustTime(60);
    else if(clockCursor == 6)
      adjustTime(10);
    else if(clockCursor == 7)
      adjustTime(1);
  }
  up = 1;
}
void DownButton(){
 if (down == 0) {
    if(clockCursor == 0)
      adjustTime(-36000);
    else if(clockCursor == 1)
      adjustTime(-3600);
    else if(clockCursor == 3)
      adjustTime(-600);
    else if(clockCursor == 4)
      adjustTime(-60);
    else if(clockCursor == 6)
      adjustTime(-10);
    else if(clockCursor == 7)
      adjustTime(-1);
  }
  down = 1;
```

Figure 27: User Interface Program Code

The UpBotton and DownButton are functions that will either increase or decrease the time of the clock. There are six columns for the time of the clock. The clock is in military time. The “0” and “1” clockCursor positions refer to the hours, the “3” and “4” clockCursor positions refer to the minutes, and the “6” and “7” clockCursor positions refer to the seconds.

```
void digitalClockDisplay() {
  // digital clock display of the time
  Serial.print(hour());
  printDigits(minute());
  printDigits(second());
  lcd.setCursor(0,0);
  if(hour() < 10)
  {
    lcd.print(0);
    lcd.setCursor(1,0);
  }
  lcd.print(hour());
  printLCDmin(minute());
  printLCDsec(second());
  Serial.println();
  lcd.setCursor(0, 1);
  lcd.print("                ");
  if(clockCursor >= 0)
  {
    lcd.setCursor(clockCursor, 1);
    lcd.print('^');
  }
}
void printDigits(int digits) {
  Serial.print(":");
  if (digits < 10)
    Serial.print('0');
  Serial.print(digits);
}
void printLCDmin(int digits) {
  lcd.setCursor(2,0);
  lcd.print(":");
  lcd.setCursor(3,0);
  if (digits < 10)
  {
    lcd.print('0');
    lcd.setCursor(4,0);
  }
  lcd.print(digits);
}
void printLCDsec(int digits) {
  lcd.setCursor(5,0);
  lcd.print(":");
  lcd.setCursor(6,0);
  if (digits < 10)
  {
    lcd.print('0');
    lcd.setCursor(7,0);
  }
  lcd.print(digits);
}
```

Figure 28: User Interface Program Code

The digitalClockDisplay formats and prints the time to the LCD screen. It also prints to the serial monitor to help with troubleshooting purposes.

3.2 Results

The *Automated Gardening System* was tested in two different batches. The first batch study ran for ~5 weeks and had many problems because of social disputes causing irregular frequency data. In the process of settling those disputes, there were several days in which we didn't run our experiment. Another problem we noticed was that we originally had too many members of the team recording data. This led to several problems, such as inconsistent measuring practices and excessive strain on the plants as were handling them too frequently. For the second batch of tests the team had resolved many of the issues that had caused discrepancies for the first batch of plants and improved upon areas to retrieve more accurate results. This batch of tests contained the following 3 groups:

- Four hours of 2.7 kHz frequency at ~110dB & eighteen hours of full spectrum lighting
- No frequency stimulation & eighteen hours of full spectrum lighting
- Indoor control group

Each batch had five mung bean plants and grew for ~1 month. Each group received the same amount of water and nutrients. The parameters we were testing for included: average plant height, stem width, leaf length, & leaf count. These parameters were decided upon because they act as a reliable representation of plant health. Looking over all these parameters, one can get a good view a plants current well-being.

3.2.1 Plant Height

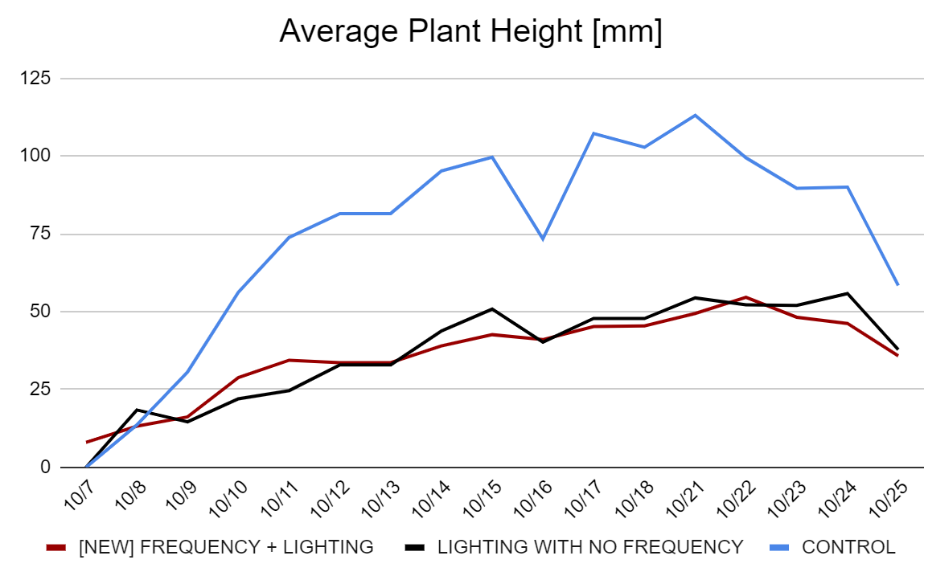

Figure 29: Trial 1 Average Plant Height

The plants that received the frequency in the first trial were the first to grow. With all the discrepancies, it is hard to say if this trial data is accurate. The control group had the tallest height due to them trying to reach the sunlight outside. It makes sense the plants with frequency and no frequency did about the same. During this trial we didn't have the frequency on most of the time due to having to move the garden and noise complaints.

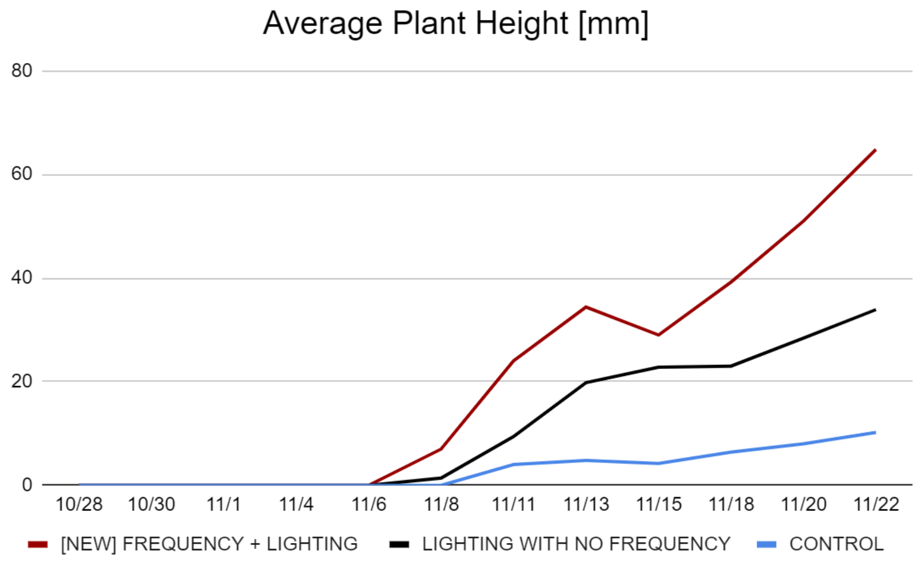

Figure 30: Trial 2 Average Plant Height

Throughout testing, the group with the fastest germination rate is the group that received lighting and a four-hour dosing of a 2.7 kHz frequency. The following group which only received lighting grew at a slightly slower rate than that of the group that received the frequency stimulation. Overall, the control group grew at a much slower rate than either of the other groups.

At the end of testing the group of mung beans that received the four hours of 2.7 kHz frequency stimulation and 18 hours of full spectrum lighting grew 47.7% taller than the group that did not receive frequency stimulation & grew 84.3% taller than the indoor control group.

3.2.2 Plant Stem Width

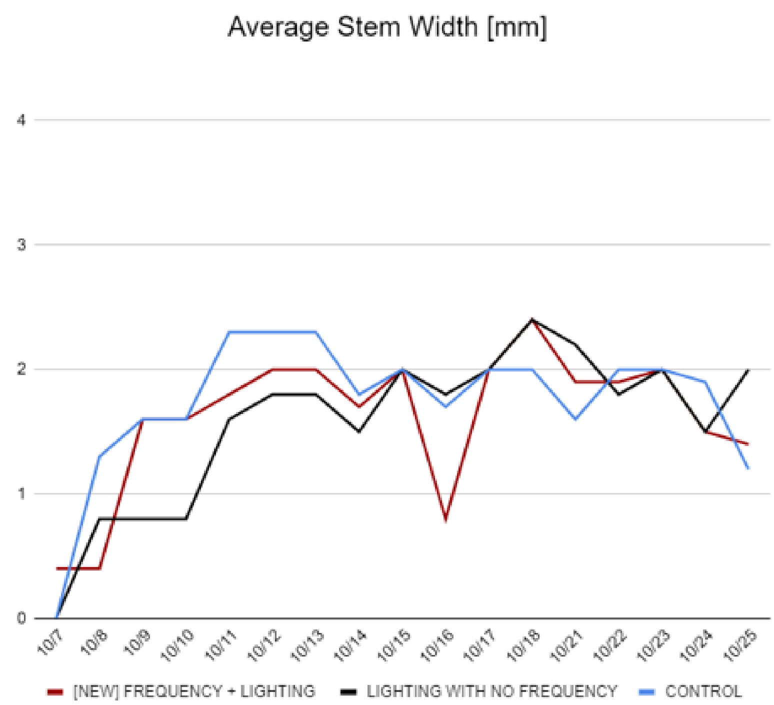

Figure 31: Trial 1 Average Stem Width

During the first trial we had multiple people recording data on the plants. Since there wasn't one person collecting data. The data got skewed. Some people recorded data differently than others. The objective was to measure the thickest part of the stem, but sometimes the person recording the data measured the bottom of the stem. This caused the data to oscillate up and down and gave the team unreliable data to compare the different plant environments.

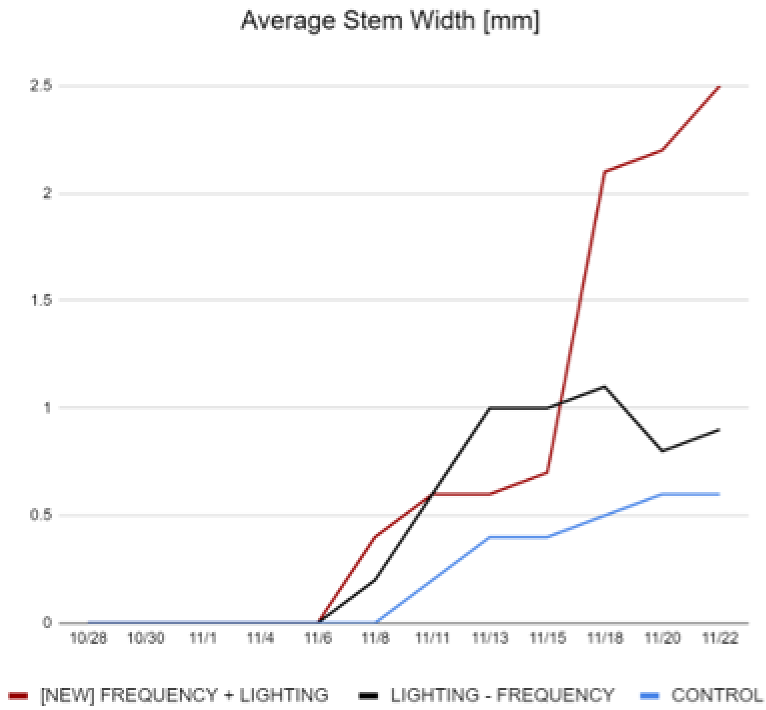

Figure 32: Trial 2 Average Stem Width

Plants with a thicker stem tend to be healthier than those with thinner stems. Thins stems tend to display signs that the plant is not receiving the appropriate amount of lighting. Both the group that received the 2.7 kHz frequency stimulation and the group that only received the lighting stayed similar over the first 3 weeks but around the final week of testing they started to differentiate. At the end of testing the group that received the three hours of 2.7 kHz frequency and eighteen hours of full spectrum lighting had a larger average stem width then its' counterparts. This group had an average stem width that was 64% larger than the group that did not receive frequency stimulation and 76% thicker than the indoor control group.

3.2.3 Number of Plant Leaves

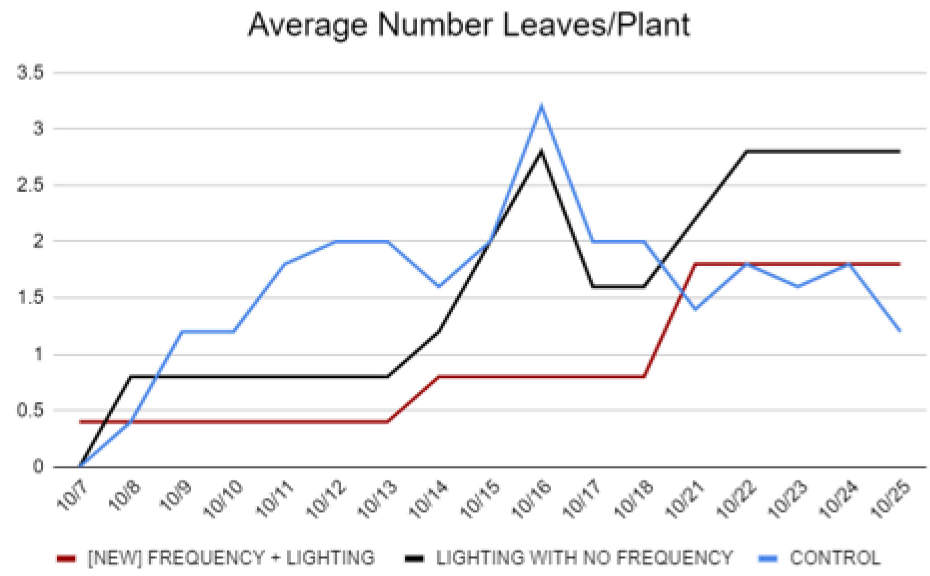

Figure 33: Trial 1 Average Number Leaves/Plant

The control group had the highest number of leaves per plant at one point. The team believes this happened due to the plants trying to reach for the sunlight. Their increase in height caused them to sprout leaves more quickly than others, but not necessarily healthier than the other environment of plants. If compared to the other graphs of data. The control plants had the tallest and skinniest plants. These were the least healthy plants out of the three. If we grew this trial for a longer amount of time. The team believes the plants with the frequency would've surprised the other two groups of plants. As previously states numerous times. The team says the first trial of plants cannot be used as reliable data due to many interruptions of frequency dosing, and over watering of the plants. This data is being shown to show the team's trial and error during this overall experiment.

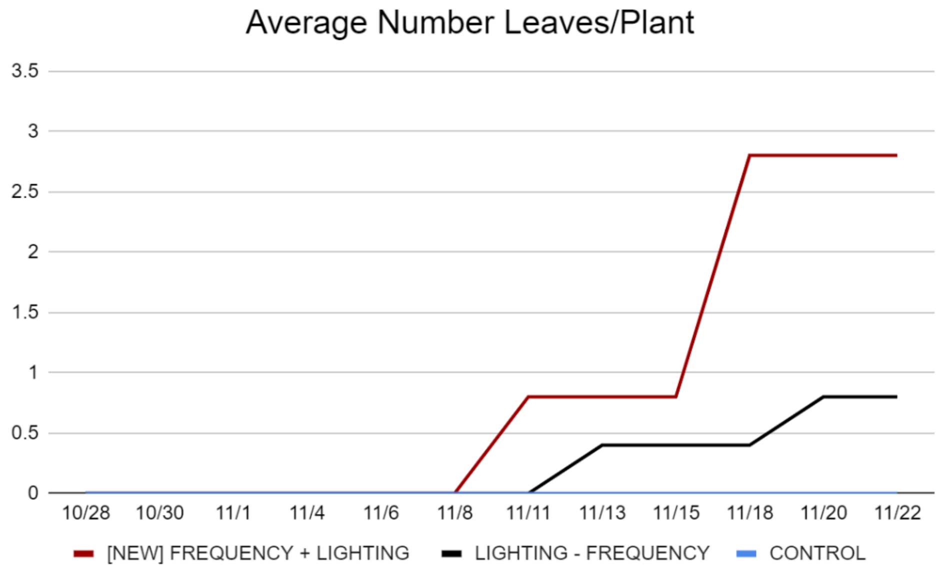

Figure 34: Trial 2 Average Number Leaves/Plant

The group that received the three hours of 2.7 kHz frequency stimulation and eighteen hours of full spectrum lighting was the first group to produce any leaves, followed by the group that only received the full-spectrum lighting. The control group was unable to produce any leaves during the ~4 weeks of testing. By the end of testing the group that received frequency dosing produced on average 71.4% more leaves then the group that did not receive frequency stimulation.

3.2.4 Average Total Length of Leaves

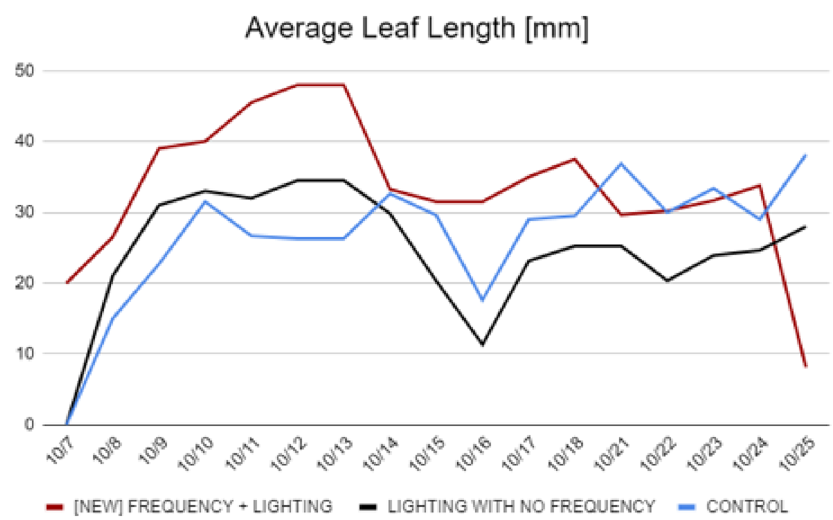

Figure 35: Trial 1 Average Leaf Length

The team was over watering the first trial which caused the plants to get sick. This caused the plants to lose leaves as time went on. Notice the frequency plants had the longest leaves, but the team can't say for certain this was caused by the frequency because it was not on all the time. Since the plants were being interrupted so frequently, the team could not rule out the frequency being the only parameter effecting the plants.

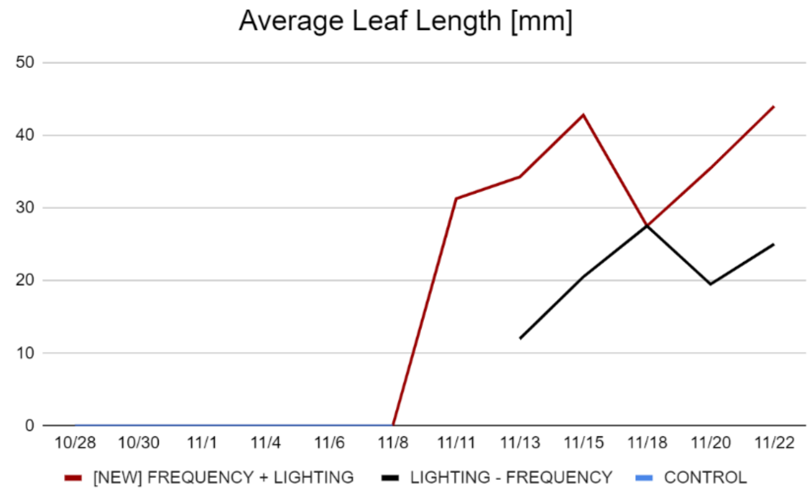

Figure 36: Trial 2 Average Leaf Length

Since only the group that received the three hours of 2.7 kHz frequency dosing and eighteen hours of full spectrum lighting and the group that only received the full spectrum lighting were the only groups to produce leaves, data for leaf length was only collected for these two corresponding groups. Dips in leaf length can be caused due to new growth of a budding leaf or loss of a leaf due to either wind or human intervention. At the end of testing the group that received the frequency dosing had an average leaf length that was 43% longer then the group that had only received the lighting treatment.

3.2.5 Plant Visuals

The following plants are the end results of our two trials. The first trial's data was discarded due to many parameters that altered the plants growth such as: over watering, moving the garden, and having to unplug the sound frequency during testing. The second trial collected good data which we used to come to our conclusion. We wanted to include the first trial because we believe our failure is what brought us to our success for the second trial.

Figure 37: Control Group Trial 1 End Result

The control group in the first trial had the tallest plants out of the three groups, but they looked weak and frail. They appeared to be struggling to receive the sunlight from the outside. The stems of the plants looked like spaghetti noodles and weren't strung enough to hold themselves up straight.

Figure 38: Control Group Trial 2 End Result

The control group in the second trial barely broke ground. The team believes this happened because of the cold weather at the time of growth. Also, the daylight hours were shorter than the previous trial which we believe impacted the plants growth. The mung beans like a warm environment and being next to a window during the colder season of the year probably didn't help the plants.


Figure 39: Lighting and No Frequency Trial 1 End Result

As seen in Figure 29, the plants look wilted. This was caused by us over watering the plants. We concluded the first trial at this point because the plants were not making any progress in their plant growth. We made the plants sick and halted their growth as they began to die off.


Figure 40: Lighting and No Frequency Trial 2 End Result

The second trial did much better. Notice how the plants are not wilted and green. The LED grow light made a difference in the growth of the plants. The plants look strong and health.

Figure 41: Lighting and Frequency Trial 1 End Result

As seen in Figure 31, the plants look wilted. This was caused by us over watering the plants. We concluded the first trial at this point because the plants were not making any progress in their plant growth. We made the plants sick and halted their growth as they began to die off. The frequency plants did manage to be the best-looking plants even though we did not stimulate the plants as much as we would've liked.

Figure 42: Lighting and Frequency Trial 2 End Result

The second trial of plants for the frequency plants were the best plants out of all the trials and environments. Visually, these plants are the most mature out of the other two groups. The plants are green, thicker, and healthy. The frequency plants did look like the plants in Figure 30 two weeks prior to this picture. We believed the plants with the light and without the frequency would've had the same result as these plants, but it would've taken them much longer.

3.2.6 Results Overview

Overall, through all parameters of testing the group that received four hours of 2.7 kHz frequency dosing at 110bB and eighteen hours of full spectrum lighting were significantly heathier and further along in overall growth than that of the other groups. The group that received only the eighteen hours of full spectrum lighting produced results that were almost half of that of the group that received frequency stimulation. The control group developed at a much slower rate than both other groups that were tested.

4 Management

The team voted Alan Paaren to be the team leader. Since there are multiple factors contributing to this project, every team member was the leader within their system. Every team member was responsible for helping each other, but it was up to the system leader to do most of the research and design for their system. The team met weekly to discuss findings and work on the garden.

4.1 Project Milestones

The team created a Gantt chart at the end of the spring 2019 semester to set goals of what the team needed to accomplish on a weekly basis. The original planned days are colored in green, and the tasks accomplished per week in the fall 2019 semester are colored in purple.

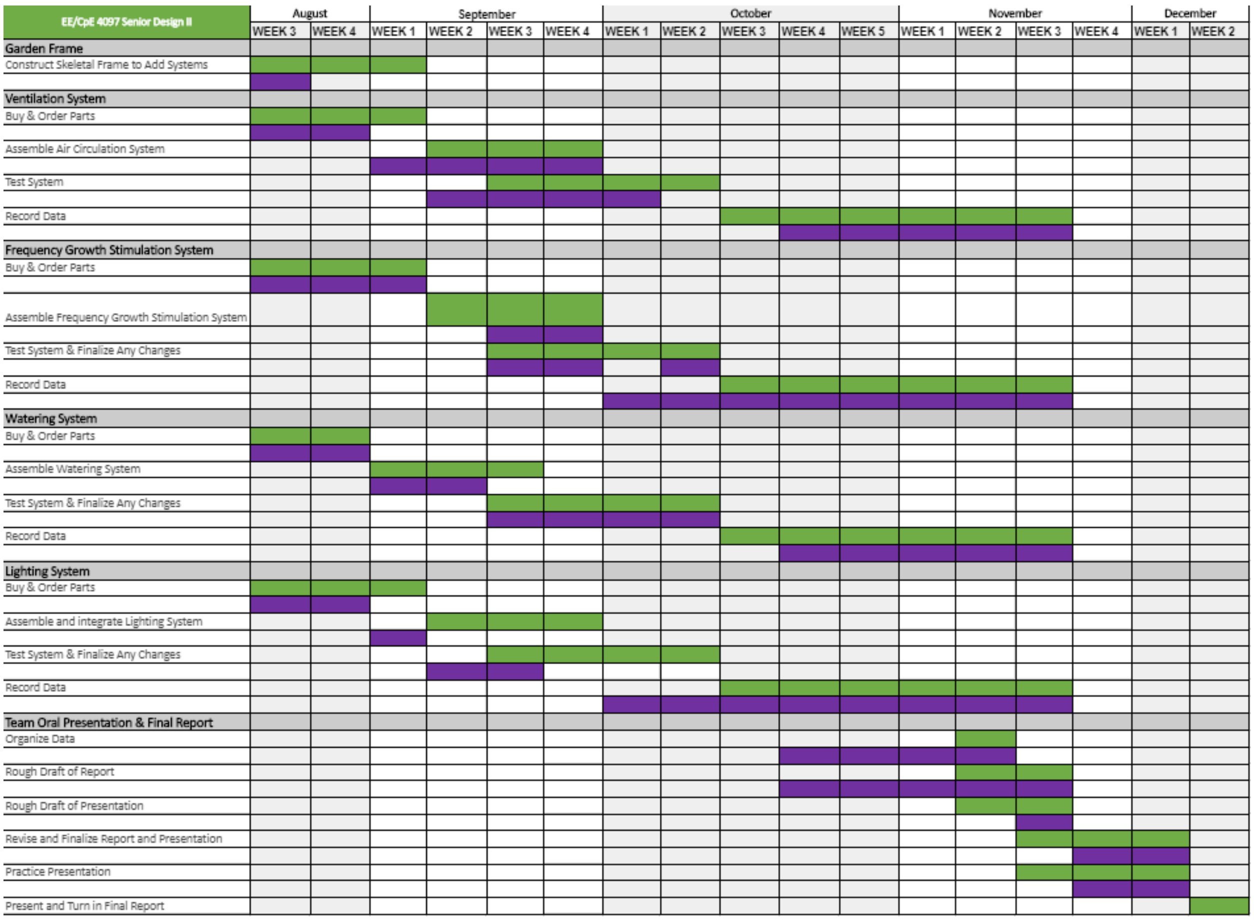

Table 2: Project Timeline and Milestones

4.2 Encountered Challenges and Lessons Learned

Throughout the semester we've encountered numerous scenarios where our initial plans didn't work out the way we wanted to. Whether that be electronically, or on our ability and knowledge of our gardening skills, at the end we've grown in every aspect. Overall, the most important thing we've learned is to properly plan. When we initially ordered our parts, we were optimistic that everything was going to be a simple connection. We encountered problems such as components frying, lack of power, and plants dying. We came together as a team and were able to overcome all these obstacles to create a working product. The watering system took numerous trials to get functioning properly. We did not account for the output pressure of the solenoid valve being less than the natural freefall pressure of gravity. The solenoid valve restricted the flow of water which did not provide enough power for the water to shoot out from the sides of the plants. We had to recreate a new PVC pipe layout to accommodate for the lack of pressure.

None of us are garden experts by any means. For the first trial of plants, we over watered them which caused them to get sick and die off. This failure made us prepare for the second batch of mung beans.

While constructing our circuit for our air circulation system, we fried our temperature/humidity sensor, which controlled the fan. In hindsight, we should have tested our circuit on a software simulation before wiring everything together. This taught us two things: conduct thorough testing and have a backup plan. Having fried our sensor let to us having to order a new one with delayed us weeks from installing the air circulation system. When we initially ordered our parts, we should've ordered more than more for this possibility, but at the same time we should've tested the circuit beforehand.

Throughout the semester I faced many challenges. Being the team leader was a challenge itself having to be on top of everyone to know what is being accomplished. Every week I had to make sure our team delivered the goals we set out to make and get everyone to fill out their portion of the weekly status report before it was due. Luckily the team I was a part of did an outstanding job with doing what they said they would do. I gave my team members input when they would present me a design to make sure it would be compatible with all the other systems. The most challenging task I faced was designing the watering system. We had constraints to keep the cost low, but to also provide water in an efficient way without interrupting any other system. The biggest concern was blocking the LED grow lights. I had to design something that would provide water without covering the plants area of growth and be in the way of the light. Since the tent was only so big the light was very close to the plants unlike a warehouse. In a warehouse the foot-candles wouldn't have been a problem if the watering system was in the way. I ended up using PVC and tubing to dispense the water. The tubing provided flexibility while the PVC was structurally fixed. I used gravity as my friend to provide water without having to purchase a pump.

-Alan Paaren

One of the largest challenges I faced in developing the frequency stimulation system was determining whether it would be better to generate the frequency through code or to hardwire circuitry to produce the desired frequency. Another challenge I had faced while developing the system was figuring out how to amplify the noise the speaker produced. After research I was able to determine a way to amplify the signal to produce a ~110dB output, but this required the utilization of a 9V battery. My next challenge was determining a way to utilize the Arduino Mega board to feed the 9V input that the signal amplifier needed to produce the desired output. I settled on a small s ale transformer to convert the 5.5V from the board to the 9V output I needed to feed the amplifier.

-Nicole Voss

A challenge I faced early in the semester was trying to figure out how I wanted to construct the circuit for the air circulation system. Initially, I tested the fan by connecting it to a 24V power supply and it worked perfectly, however I had to figure out how to make it work with the Arduino Mega and the DHT22 temperature and humidity sensor. Luckily, there were plenty of circuit designs that I found on the web to pull from and constructed a working configuration. My next challenge came from when we connected the DHT22 sensor to the circuit. While testing the circuit to see if the DHT22 sensor was reading any values to activate the fan, the sensor was fried and stopped working. This caused a delay in implementing the system with the rest of the garden because we had to order a new sensor. Once the sensor came in, I connected it to the circuit and the system worked perfectly.

-Samuel Ogunmolawa

The most challenging aspect that I faced for this senior design project was trying to integrate IR communication into the Arduino. One of the first problems that I encountered was that our funding was exhausted around the time I planned on purchasing the parts. Luckily the main components were inexpensive, so I decided to purchase them using my own funds. Once the parts had arrived, I began testing the design on the Arduino Due that we used as a test controller. Since lighting system used an infrared remote to control the settings. I needed to first copy the inputs for the various buttons already being used with an infrared receiver. Then decode the data using libraries that I found online. Finally, I needed to then retransmit the desired signal using an infrared diode connected to the Arduino. The problem I had with this was that the libraries needed to decode the data were not compatible with the Arduino Due. After some time troubleshooting the problem, I found out the Mega being used to run the system was compatible with libraries. However, at this point it was already very late in the semester and didn't want to risk miswiring anything on the Mega as the demo was soon approaching. Another problem was that the Arduino was placed underneath the lighting system and the infrared transmitter would need to be place level to lighting system. This would have required even more parts, as I would now have to purchase some type of conduit and mount to run the wiring to where it needed to be placed.

-Mohamed Mohamed

I faced a number of challenges when working on this project. One of the earliest challenges occurred when we had started testing and noticed that the system's clock was off. We figured out that the issue was caused by a short power outage on campus where we had set up our system. Such a system failure occurred a couple times during the semester and was difficult to detect as there was no way to view the clock at the time. Although I had no control over the campus' power issues, I was able to find a fix by implementing an LCD screen to display the clock and a 4-button array to manually reset the clock. I also helped my team members tackle code-related challenges. Each team member had assembled code for their individual system by following guides and tutorials online, but getting these individual systems to all run off the same Arduino with the same clock called for some parts of these systems to be reworked. Implementing the button array was especially difficult because the buttons must be checked frequently to be functional but can interrupt other processes while being checked. Lastly, unforeseen constraints from the department and our advisor Dr. Ferguson posed additional challenges. As Dr. Ferguson was helping fund the project, we had to predict the majority of the parts that we'd require during the first week of the semester before he no longer had access to his MST account. Although we were able to get access to most of the materials we needed, many of the previously mentioned challenges required materials for solutions. Getting shuffled around by the department also gave us a significant challenge. Although we had clearly communicated our intention for the system and our need for a space where we could use our system, we were placed in a space where our system was disruptive for other students. Because we had to move our system, we lost our first few weeks of data. Luckily, we were ahead of schedule, so this major setback only put us even with our expected progress.

-David Szatkowski

4.3 Team Ethics Discussion

Communication was key in our team being successful this semester. Lack of communication causes confusion and possible overlap of work. Having everyone on the same page informed the team of what needed to be accomplished in order to proceed further to our project. The team communicated regularly on an app called "GroupMe". Every team member oversaw their own system. They oversaw designing their system, and if the member was unsure of their design, other team members would valid forth their input. Having five modular designs in the garden broke up the workload evenly. Each individual had plenty of work throughout the semester, but if they were slump for work, the team leader would assign team members things to do.

Once a design was planned out. The team leader oversaw approving the system design to ensure it wouldn't interfere with another systems operation. The assembly of the garden was a team effort. Everyone was accommodating of one another. The team understood we all have other commitments and assignments throughout the semester and would give an extra hand of help if needed.

After the assembly of the garden, everything was a team decision. This was because the result of those decisions would impact the whole team. We discussed how we wanted to test our plants, and record data. We created a schedule of who was assigned to measure and move the boxes of plants on certain days to make sure our data did not get corrupted.

Whenever an issue did occur, the team got together to discuss a solution. Since there were only five of us, we made sure we all concurred on the same outcome. If one or two people disagreed with a suggestion, the team would talk it out to understand the perspective of each member to create common ground. All members think differently in some way. Most disputes were caused by lack of information or lack of thought. Therefore, us discussing as a team was so effective. We could fill in the gaps with one another.

4.4 Budget

The original budget was designed when we began initial planning at the end of the 2019 spring semester. The team had a good idea of how they wanted to implement their design. However, as the 2019 fall semester progressed, the team encountered issues with designing with some of the parts on hand. Highlighted in yellow were additional items that were added to the budget to accommodate feasibility of the garden. Parts were added due to either lack of tools, components frying, or additional features to the garden. As the semester went on, some of the previous parts ordered were not used which are highlighted in green. The additional expenses were funded with the money the team raised through volunteer hours or out of pocket.

The expensive items such as the grow light and the axial fan were a necessity to the garden. The grow light provided energy to the plants and the fan helped control the temperature inside the tent and simulated wind to help strengthen the plants.

	Items	Unit cost ($)	Quantities	Total cost ($)	Comments
Parts	Arduino Mega	$38.50	1	$38.50	Control center of project
	555 Timers	$0.86	1	$0.86	Originally purchased for frequency generator
	Wood 2” X 4” X 10’	$4.18	8	$33.44	Building materials
	24V Axial Fan	$49.72	1	$49.72	Air Circulation System
	Speaker	$12.99	1	$12.99	Sound Frequency System
	5V to 24V PLC Signal Converter Module	$5.58	1	$5.58	Initially wanted to power 24V sources from Arduino, lacked power
	Plastic Water Solenoid Valve - 24V - ½”	$16.99	1	$16.99	Watering System
	PVC Pipe ½” X 10’	$1.85	3	$1.85	Watering System
	PVC ½” X ½” 90-Degree Elbow	$0.35	6	$2.10	Watering System
	1/2 in. Schedule 40 PVC Cross	$1.44	1	$1.44	Watering System
	1/2 in. PVC Sch. 40 Female S x FPT Adapter	$0.75	1	$0.75	Watering System
	Humidity & Temp. Sensors	$7.93	3	$23.79	Originally purchased one, but the part fried
	Soil Moisture Sensor	$6.99	1	$6.99	Watering System
	300W Grow Light	$98.99	1	$98.99	Lighting for plant growth
	32oz Phosphorus Fertilizer	$27.49	1	$27.49	Nutrient Dosing
	32oz Nitrogen Fertilizer	$14.95	1	$14.95	Nutrient Dosing

	32oz All Purpose Fertilize	$24.99	1	$24.99	Nutrient Dosing
	7 Gallon Water Container	$14.97	1	$14.97	Watering System
	Bluetooth Module- JY-MCU	$12.97	1	$12.97	Discarded and replaced with button user interface
	Breadboard	$7.99	1	$7.99	
	Wire	$15.39	1	$15.39	
	Potentiometers	$11.97	1	$11.97	
	Potting Soil	$7.94	1	$7.94	
	Wire Nuts	$6.25	1	$6.25	
	24V DC Power Supply	$14.96	1	$14.96	
	Flex Seal	$14.99	1	$14.99	Waterproof wood material
	LCD Screen	$5.99	1	$5.99	User interface
	4' X 8' Sheet of Plywood	$18.00	1	$18.00	Building materials
	150 Count of ½" Wood Screws	$3.00	1	$3.00	Building materials
			Total cost	**$ 495.84**	

Table 3: Project Budget

4.5 Funding Source

Funding was provided by Dr. Ferguson for the initial parts list created by the team. As the semester progressed, other parts were needed that we did not previous purchase through Dr. Ferguson's SRI account. The team funded additional parts that were necessary to complete a working project.

4.6 Human Safety Assessment

Safety risks listed below pose as potential risks to either the team members during development or potential customers.

Sharp Edges

a) Probability of occurrence: Medium
b) Effects and their severity: Puncture Wound - High
c) Appropriate alleviation and mitigation steps:
   - Call or visit the nearest health professional. Ensure safety gear is worn during all times and that proper training is in place.

Exposed Wires

a) Probability of occurrence: Medium
b) Effects and their severity: Electric Shock - High
c) Appropriate alleviation and mitigation steps:
   - Call or visit the nearest health professional.
   - Ensure safety gear is worn during all times and that proper training is in place.

Power Tool Misuse

a) Probability of occurrence: Low
b) Effects and their severity:
   - Eye Injury - High, Puncture Wound - High, Broken Bone - High, Fire - High
c) Appropriate alleviation and mitigation steps:
   - Eye injury, puncture wound, or broken bone:
     - Call or visit the nearest health professional.
     - Ensure safety gear is worn during all times and that proper training is in place.
   - Fire:
     - Retrieve a fire extinguisher and sweep the extinguisher at the base of the fire.
     - If the fire is too large or no fire extinguisher is in the area alert the appropriate authorities.

<u>Fire</u>

a) Probability of occurrence: Low
b) Effects and their severity: Open flames - High, Fumes - High,
c) Appropriate alleviation and mitigation steps:
   - Retrieve a fire extinguisher and sweep the extinguisher at the base of the fire. If the fire is too large or no fire extinguisher is in the area alert the appropriate authorities.

4.7 Member Credentials and Responsibilities

4.7.1 Teamwork

- Alan Paaren is organized and has a great work ethic. Alan has worked with wiring small electronic components and sensors while working at The Clorox Company and knows power distribution while working at Burns & McDonnell. Overall Alan is familiar with lots of different things and learned lots of hands on skills.

- Mohamed has experience with Arduino's and integrating sensors from his involvement with the RDT on campus. This was useful during the assembly of the project. Mohamed has helped build some of the circuit for the project. He also has experience with programming which impacted the team when it came to program the Arduino.

- Nicole's experience in AutoCAD, software, photography, and gardening assisted in a variety of ways. Her prior experience in gardening allowed for a more in-depth interpretation of the plant data retrieved from the system. As well as her background in AutoCAD allowed for development of the speaker casing.

- Samuel has experience in graphic design, Photoshop, and photography. He used his skills to design the team's PowerPoint presentation, poster, and assisted in taking pictures and recording footage for the final presentation.

- David's experience in problem solving, coding in C++, and reading documentation for programming languages were very useful in developing the code for the project. Additionally, he used his skills in technical writing, teamwork, and communication to assist in the write-ups of team reports and to work well with the rest of the team.

- The project requires knowledge and experience in mechanical engineering, microcontrollers, sensors, coding, and power electronics.
    - Alan's experience with power electronics, sensors, and AutoCAD, helped the team design the garden frame and wire the electronic components.
    - Mohamed's experience with sensor integration helped the team during assembly when the various systems need to interact with one another.

4.7.2 Alan M. Paaren (ampcn7@mst.edu, EE major):

Member profile: Mr. Paaren served as **team leader** and arranged regular team meeting to discuss team's progress. He was involved in the construction of the plant boxes, the watering system, the speaker frame, the axial fan mount, and coding for the system. He was responsible for making sure deadlines were met, and that other team members did their agreed upon contributions to the project. He oversaw the communication role between departments faulty and our advisors to ensure we had everything needed to conduct this project. Mr. Paaren organized meetings and sent out files to advisors and instructor.

4.7.3 Nicole L. Voss (nlvdkc@mst.edu, EE major):

Member profile: Ms. Voss served as a team member specialized in the development of the frequency generator system. This included development of the circuitry of the frequency generator and reassuring that the frequency system operated properly. Also, Ms. Voss oversaw designing and developing the frequency system casing. She regularly attended team meetings and assisted in system troubleshooting discussions. Also, she researched through various amounts of studies on frequency stimulation's impact on plant growth to ensure that the system met the right parameters for the overall systems tests.

4.7.4 Mohamed S. Mohamed (mamt6f@mst.edu, EE major):

Member profile: Mr. Mohamed served as a team member and the lead for the Lighting and Nutrient Dosing system. He was mainly responsible for the lighting schedule, as well as, the NPK ratio and its administration. He helped launch the initial prototypes that were used to gather data on the project's functionality and identify areas for improvement. Mohamed also regularly attended meetings to assess the projects progress. Lastly, he also took part in recording data on the condition of the plants.

4.7.5 Samuel A. Ogunmolawa (saopr4@mst.edu, EE major):

Member profile: Mr. Ogunmolawa served as a team member and the lead for the air circulation system. He was responsible for constructing a circuit for the axial fan and temperature and humidity sensor. He also designed a mounting structure for the fan that was attached to the side of the garden box. Additionally, he regularly checked to ensure the fan was functioning and monitored the temperature and humidity of the tent. Mr. Ogunmolawa regularly attended meetings, communicated frequently with the team, and recorded data on the plants during the first trial.

4.7.6 David D. Szatkowski (ddsp86@mst.edu, CpE/CS major):

Member profile: Mr. Szatkowski served as a team member specialized in the development of the frequency stimulation system. He worked closely with Ms. Voss to create the logic for the speaker and to research the specifications needed for frequency stimulation. When issues with the system occurred during testing, he was responsible for resetting the system in a timely manner. He also assisted other team members in troubleshooting code-related issues in their respective systems and was responsible for tying the sketches for all the individual systems together into the same sketch. He applied his technical writing skills when working on team deliverables. Lastly, he designed the circuitry and logic for the LCD interface and button array. Like his fellow team members, he attended weekly team meetings, assisted in recording data, and communicated frequently with the team.

5 Conclusions and Future Work

5.1 Conclusions and Lessons Learned

Our goal was to design a garden with modular components that was self-reliant. People grow different things all over the world, and we needed to create a garden that could be easily adapted to the particularly needs of users. We wanted to build something that could be scaled to accommodate different applications as well. Whether the user is hobbyist at home or an industrialist that operates across multiple acres, the *Automated Gardening System's* design allows users to modify the area of the garden by altering the number of components. The final design we implemented is geared towards the typical home grower but could easily be modeled for larger applications by increasing the number of lights, fans, speakers, and PVC for water.

The key feature of the garden that separates us from other indoor garden is through our usage of sound frequency stimulation to increase plant growth, germination time, and crop yield. We conducted tests to verify our research and from the data collected we proved this to be true. We test using three different samples; two which were inside the garden, and a third outside that was used as a control. These samples were classified as the following: LED lighting and sound frequency, LED lighting and no sound frequency, and a control group outside of the garden which received neither. The control group outside the tent was cared for manually and used ambient light from the sun. Aside from this, it received the same amount of water and nutrients as other two. The two environments inside the garden were autonomously cared for by the garden. The only difference between the two samples inside the garden was that one received four hours of frequency stimulation at 2700 Hz and the other did not. From the data recorded in the trials, it's clear that frequency simulation has a significant impact on the growth and well-being of plants. We came to this conclusion after comparing the two samples inside the tent and noticing the differences between in terms of height, number of leaves, and stem thickness. Since frequency simulation was the only difference between these two samples. Our data showed that the frequency decreased the germination time, increased the yield, and overall improved the plants growth.

The team has encountered numerous obstacles and challenges over the course of this project. Although these challenges exposed flaws in our design, they also uncovered various ways in which we could improve. One of the biggest takeaways that we took from this project was the importance of planning ahead. Even though we tried to be thorough and anticipate problems ahead of time. There were still instances which caused us to question how well we planned for contingencies or unforeseen circumstances. One such example was when our DHT22 sensor fried. This set us back two weeks and prevented us from installing the air circulation system into the garden. We didn't think any of our components would just outright fail, so during our initial purchases we didn't account for spare parts. This didn't result in real impact on the results from our plants, however, we lost a fair chunk of time simply waiting for a replacement.

Another valuable lesson that we learned is that "you get what you pay for". We purchased a 24V solenoid valve for $15.00. One of our goals was to keep the cost of the project on the low side, but the PSI rating on the solenoid wasn't the best. The watering system was feed by gravity to dispense from the solenoid valve. Depending on the height of the reservoir, the solenoid valve leaked water through the pipe because it could not withstand the back pressure.

5.2 Suggested Improvements

The team accomplished a lot this semester, but if we had more time, we'd like to do a couple of things. The first thing we'd like to do is implement a better user interface. Currently we only have the clock time on LCD display, and these are the only parameters the buttons currently control. One thing that would improve the gardens functionality is giving users the ability to adjust individual parameters on the garden without going into the code. These parameters could include things such as the soil moisture limit, temperature limit, humidity limit, as well as, the how long the fan, watering, and sound frequency occur. Another thing that could also be implemented is incorporating the control of the lighting system onto the Arduino. The lighting system is currently controlled independently from the rest of the garden. However, centralizing control for the entire system is more ideal in terms of usability.

For the design we currently have implemented, the only thing we would do different is replace a few of the parts for higher quality ones. One such component is the solenoid valve. The valve had trouble controlling the water pressure from the reservoir as it would leak water to plants. Because of this, we had to accommodate for the leakage when we set the timer for watering the plants. We also used particle board for the boxes of our plants. We noticed the water from the plants began warping the boards and were not suitable for long term use as they would rot. The reason we use them in our project was because of our limited budget and a need for particular dimensions for the multiple sample. Plastic would be a better alternative because it's waterproof and the dimensional requirements only existed for research purposes.

6 References

[1] "Open-Source CNC Farming," *FarmBot*. [Online]. Available: https://farm.bot/. [Accessed: 12-Feb-2019].

[2] J. McGrath, "Eight Devices to Help You Grow a Garden Indoors," *Digital Trends*, Jan. 16, 2019. [Online]. Available: https://www.digitaltrends.com/home/indoor-garden-devices/. [Accessed: Feb. 10, 2019].

[3] "Automated Indoor Farming," *CityCrop*. [Online]. Available: https://www.citycrop.io/. [Accessed: Feb. 10, 2019].

[4] AeroGarden (2019). *https://www.aerogarden.com/bounty-elite-wi-fi.html*. [image].

[5] W. Cai, H. He, S. Zhu, and N. Wang, "Biological Effect of Audible Sound Control on Mung Bean (Vigna radiate) Sprout," *BioMed Research International*, vol. 2014, pp. 1–6, Aug. 2014

7 Appendix A: Entire Program Code

```
#include <DHT.h>
#include <Wire.h>
#include <Streaming.h>
#include <DS1307RTC.h>
#include <TimeAlarms.h>
#include <Time.h>
#include <TimeLib.h>
#include <LiquidCrystal_I2C.h>

#define DHTPIN 2     // what pin we're connected to
#define DHTTYPE DHT22   // DHT 22  (AM2302)
#define fan 4
#define spkr 7

// Set the LCD address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(0x27, 16, 2);

const int rightButton = 8;     // the number of the pushbutton pin
const int leftButton = 3;     // the number of the pushbutton pin
const int upButton = 5;     // the number of the pushbutton pin
const int downButton = 6;     // the number of the pushbutton pin

int buttonStateRight = 0;         // variable for reading the pushbutton status
int buttonStateLeft = 0;
int buttonStateUp = 0;
int buttonStateDown = 0;

bool right = 0;
bool left = 0;
bool up = 0;
bool down = 0;

int hold = 0;

int clockCursor = -1;
const int freq = 2700;
int fanCount = 11;
int maxHum = 60;
int maxTemp = 74;
int daycount = 1;
int sensorPin = A0; //moisture sensor
int sensorValue;  //moisture sensor
```

```
int limit = 600; //moisture sensor
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  lcd.init();
  lcd.backlight();
  setTime(19,0,0,11,21,19);//set time for 4pm October 30th, 2019
  Alarm.alarmRepeat(13,0,0,FreqOn);
  Alarm.alarmRepeat(17,0,0,FreqOff);  // 4:00pm every day
  Alarm.alarmRepeat(10,0,0,WaterOn);  // 10:00am every day
  Alarm.timerRepeat(3600, FanOn);     //Repeats every hour 3600 secs
  Alarm.timerRepeat(60, FanOff); //Checks every minute if X minutes have passed since
FanOn
  Alarm.timerRepeat(60, PrintLoop); //Prints every min
  pinMode(fan, OUTPUT);
  pinMode(13, OUTPUT);
  pinMode(spkr, OUTPUT);
  pinMode(rightButton, INPUT);
  pinMode(leftButton, INPUT);
  pinMode(upButton, INPUT);
  pinMode(downButton, INPUT);
  Serial.begin(9600);
  dht.begin();
}

void loop() {
  ButtonCheck();
  Alarm.delay(100);

}

void ButtonCheck(){
  buttonStateRight = digitalRead(rightButton);
  buttonStateLeft = digitalRead(leftButton);
  buttonStateUp = digitalRead(upButton);
  buttonStateDown = digitalRead(downButton);


  if (buttonStateRight == HIGH) {
    RightButton();
  } else if (buttonStateRight == LOW) {
    right = 0;
  }
```

```
  if (buttonStateLeft == HIGH) {
    LeftButton();
  } else {
    left = 0;
  }

  if (buttonStateUp == HIGH) {
    UpButton();
  } else {
    up = 0;
  }

  if (buttonStateDown == HIGH) {
      DownButton();
  } else {
    down = 0;
  }
  digitalClockDisplay();
}
void PrintLoop()
{
  //digitalClockDisplay();

  sensorValue = analogRead(sensorPin);
  Serial.println("Analog Value : ");
  Serial.println(sensorValue);

  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  // Read temperature as Celsius
  float t = dht.readTemperature();
  float TempF = (t*1.8)+32;

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }

  if(h > maxHum || TempF > maxTemp) {
      digitalWrite(fan, HIGH);
      Serial.print("Turning on Fan it's hot! ");
  } else if(fanCount > 10){
```

```
        digitalWrite(fan, LOW);
    }
    Serial.print("Day count: ");
    Serial.print(daycount);
    Serial.print(" \t");
    Serial.print("Humidity: ");
    Serial.print(h);
    Serial.print(" %\t");
    Serial.print("Temperature: ");
    Serial.print(TempF);
    Serial.println(" *F ");
}

void RightButton(){
    if (right == 0) {
      if(clockCursor < 7){
        clockCursor++;
        if(clockCursor == 2 or clockCursor == 5)
          clockCursor++;
      }
      else
        clockCursor = -1;
    }
    right = 1;
}

void LeftButton(){
    if (left == 0) {
      if(clockCursor > -1){
        clockCursor--;
        if(clockCursor == 2 or clockCursor == 5)
          clockCursor--;
      }
      else
        clockCursor = 7;
    }
    left = 1;
}

void UpButton(){
    if (up == 0) {
      if(clockCursor == 0)
        adjustTime(36000);
      else if(clockCursor == 1)
```

```
        adjustTime(3600);
      else if(clockCursor == 3)
        adjustTime(600);
      else if(clockCursor == 4)
        adjustTime(60);
      else if(clockCursor == 6)
        adjustTime(10);
      else if(clockCursor == 7)
        adjustTime(1);
    }
    up = 1;
}

void DownButton(){
  if (down == 0) {
      if(clockCursor == 0)
        adjustTime(-36000);
      else if(clockCursor == 1)
        adjustTime(-3600);
      else if(clockCursor == 3)
        adjustTime(-600);
      else if(clockCursor == 4)
        adjustTime(-60);
      else if(clockCursor == 6)
        adjustTime(-10);
      else if(clockCursor == 7)
        adjustTime(-1);
    }
    down = 1;
}
void FanOn() {
  float h = dht.readHumidity();
  // Read temperature as Celsius
  float t = dht.readTemperature();
  float TempF = (t*1.8)+32;
  Serial.print("Turning on Fan");
  digitalWrite(fan, HIGH);

  fanCount = 0;
}

void FanOff(){
  if(fanCount == 10){
    digitalWrite(fan, LOW);
```

```
    Serial.print("Fan Turned Off");
  }
  if(fanCount <= 11)
  {
    Serial.print("Fan on for ");
    Serial.print(fanCount);
    Serial.print("minutes.");
    fanCount++;
  }
}
//Timer for speaker
void FreqOn() {
  // write here the task to perform every morning
  Serial.println("Sound on");

  tone(spkr, freq);
}

void FreqOff() {
  //write here the task to perform every evening
  Serial.println("Sound off");
  noTone(spkr);
}

//Timer for watering system
void WaterOn() {
  Serial.println("Alarm: - its Raining!");
  Serial.println("daycount: ");
  Serial.println(daycount);
  if (daycount == 1 or 4) {
    if (sensorValue>limit) {
      digitalWrite(13, HIGH);
      delay(30000);
      digitalWrite(13, LOW);
    }
    else {
      Serial.println("Plants have enough water");
    }
  }
  daycount = (daycount + 1);
  if (daycount > 7){
  daycount = 1;
  }
  Serial.println("daycount: ");
```

```
  Serial.println(daycount);
}


void digitalClockDisplay() {
  // digital clock display of the time
  Serial.print(hour());
  printDigits(minute());
  printDigits(second());
  lcd.setCursor(0,0);
  if(hour() < 10)
  {
    lcd.print(0);
    lcd.setCursor(1,0);
  }
  lcd.print(hour());
  printLCDmin(minute());
  printLCDsec(second());
  Serial.println();
  lcd.setCursor(0, 1);
  lcd.print("                ");
  if(clockCursor >= 0)
  {
    lcd.setCursor(clockCursor, 1);
    lcd.print('^');
  }
}

void printDigits(int digits) {
  Serial.print(":");
  if (digits < 10)
    Serial.print('0');
  Serial.print(digits);
}

void printLCDmin(int digits) {
  lcd.setCursor(2,0);
  lcd.print(":");
  lcd.setCursor(3,0);
  if (digits < 10)
  {
    lcd.print('0');
    lcd.setCursor(4,0);
  }
```

```
  lcd.print(digits);
}

void printLCDsec(int digits) {
  lcd.setCursor(5,0);
  lcd.print(":");
  lcd.setCursor(6,0);
  if (digits < 10)
  {
    lcd.print('0');
    lcd.setCursor(7,0);
  }
  lcd.print(digits);
}
```