Efficient Light Control System for Plant Growth

By

Christelle Seri (seri2),

Heonjang Lee (hl8),

Sungjoo Chung (sungjoo2)

ECE 445 Senior Design Document

Fall 2022

TA: Zhicong Fan

09/29/2022

Project 5

Contents

1. Introduction

1.1. Problem

Greenhouses in the industry are essential in the agricultural fields, but also costly. Greenhouses are proven to be an effective solution to growing plants, but over time, the electricity costs will begin to add up. According to research, a 200ft * 100 ft greenhouse costs about $6000 per month [1].

1.2. Solution

This project proposes an energy efficient blinds system with UVA lights as a solution to simulate a cost effective greenhouse system. A sensor would be placed on the plant vase to measure the amount of light received. The blinds would adjust via the attached motor so as to optimize the amount of light to the plant. The UVA lights will turn on when the maximum sunlight from the blinds is insufficient.

Thus the UVA lights would only be used when strictly necessary, cutting down on electricity costs as a result. Additionally, the blinds system could be scheduled and adjusted to user needs as well.

This system will be easily controlled by a user using a mobile application, and also statistics will be provided on the application.

1.3. Visual Aid

Figure 1: Model of the system

1.4. High Level Requirements

- The artificial light-source, combined with available natural light, should provide light of wavelength 400-700 nm, which has been proven to be optimal for plant growth [2], and a maximum of 3,500 lux over a 12 hour period [3], to provide sufficient light for high-light plants, when needed.
- The photosensors on the vase should correctly calculate the illumination on the plant to minimize the discrepancy between the actual illumination on the plant and the expected illumination within ±5%.
- The application should have enough modes to cover various types of plants including cactus, tropical plants, conifers, etc

2. Design

2.1. Block Diagram

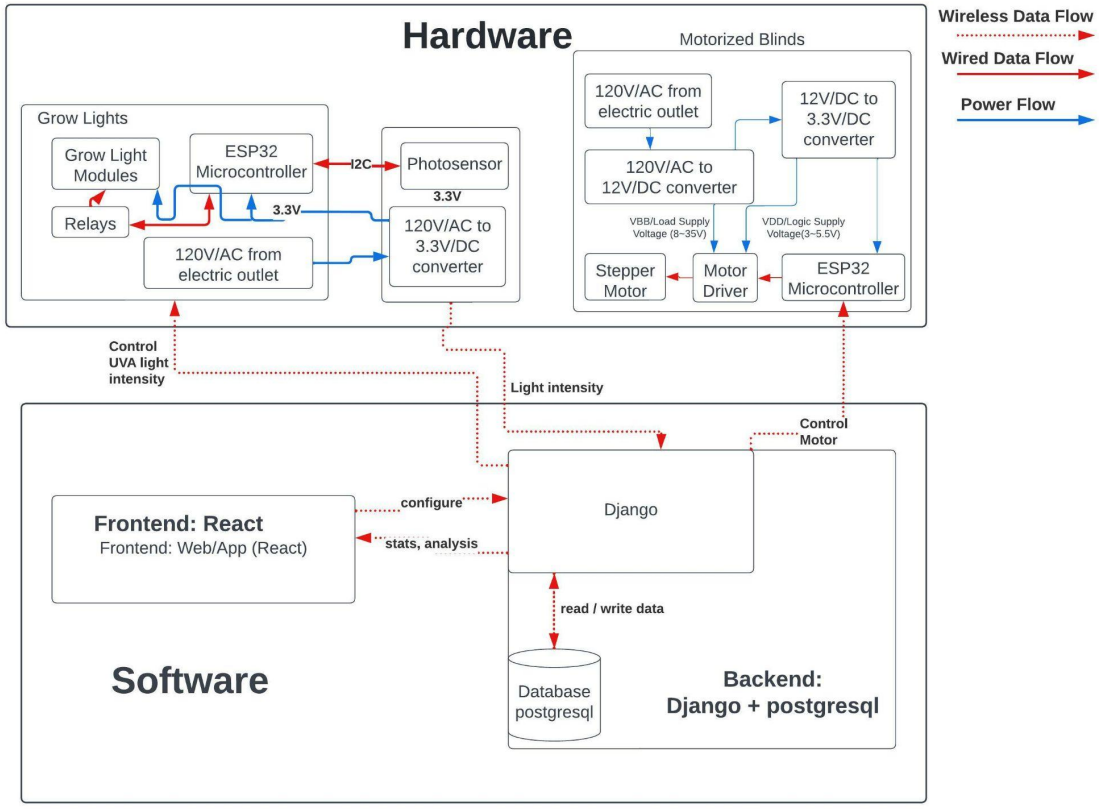

Figure 2: Block Diagram of the Project

2.2. Subsystem Overview

Photosensor

The photosensor used will be a BH1750FVI Luminosity sensor. The advantage of this sensor is that it is a precise digital device, and compact in size as well. There will be an ESP32-WROOM-32E which will communicate directly with the BH1750FVI via I2C. Luminosity data will be transmitted to the app through bluetooth. The ESP32 was chosen as it is a well documented microcontroller with both bluetooth and wifi capabilities. The ability to communicate with the microcontroller over both I2C and SPI allows for a lot of flexibility in our design. Additionally, it can be programmed reliably using the Arduino IDE, and there are a lot of prior project implementations to reference. This subsystem will share a microcontroller with the grow light subsystem.

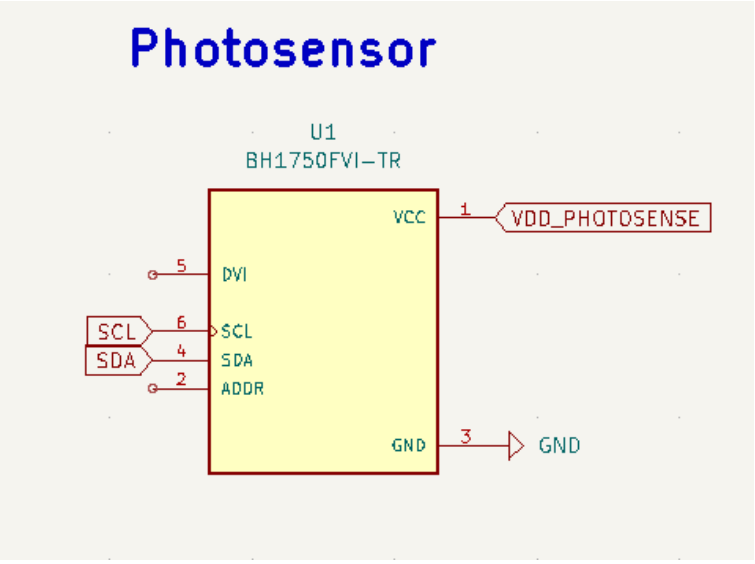

Figure 3: Photosensor Schematic

Requirements	Verification
1. The photosensor subsystem will transmit real time luminosity data to the rest of the system via bluetooth (every second) 2. The photosensor's measurements will reflect changes to the light to the plant with a delay no greater than 2 seconds	1. The data from the photosensor will be polled for a period of 30 seconds at the end of which 30 accurate measurements should be uploaded to the server 2. A phone's flashlight will be used to adjust light to the plant, in a range of none to the maximum possible light. The readings from the photosensor should reflect changes in the flashlight intensity.

Table 1: Requirements and Verification for Photosensor subsystem

Grow Lights

To maximize the energy efficiency of the system, an adjustable LED light circuit will be implemented. LEDs were chosen for their energy efficiency, low cost and low heat emission [4]. Implementing an LED grow light circuit allows for optimization of the light provided to the plant. Below is a table showing the benefits of different types of LEDs [2].

Blue Light: increases chlorophyll production	**Yellow-Red Light:** improves chlorophyll absorption, germination and bud development
Green Light: helps with photosynthesis and can improve plant size	**UVA Light:** can enhance plant pigmentation, and thicken leaves

Table 2: Improvement on Plant Growth from Different LEDs

As such, the design includes a UVA, Yellow-Red, Green and Blue LED. The combination of these four different wavelengths of light should provide light optimal to plant development. These arrays of four LEDs will constitute individual grow light modules in our overall circuit design.

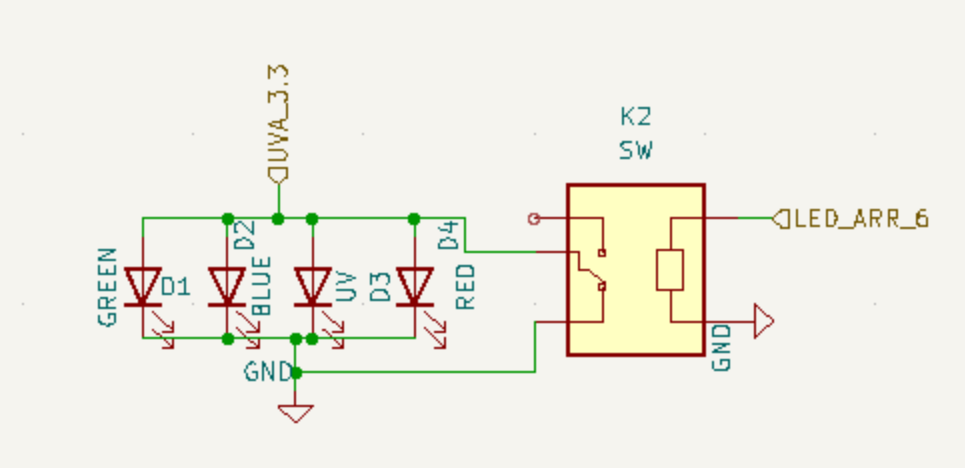

Figure 4: Single Grow Light Module

These grow light modules will be switched on and off using relays. The microcontroller will switch modules on and off accordingly to adjust the light to the plant. To provide sufficient light to the plant, 18 of these grow light modules will be implemented as shown below.

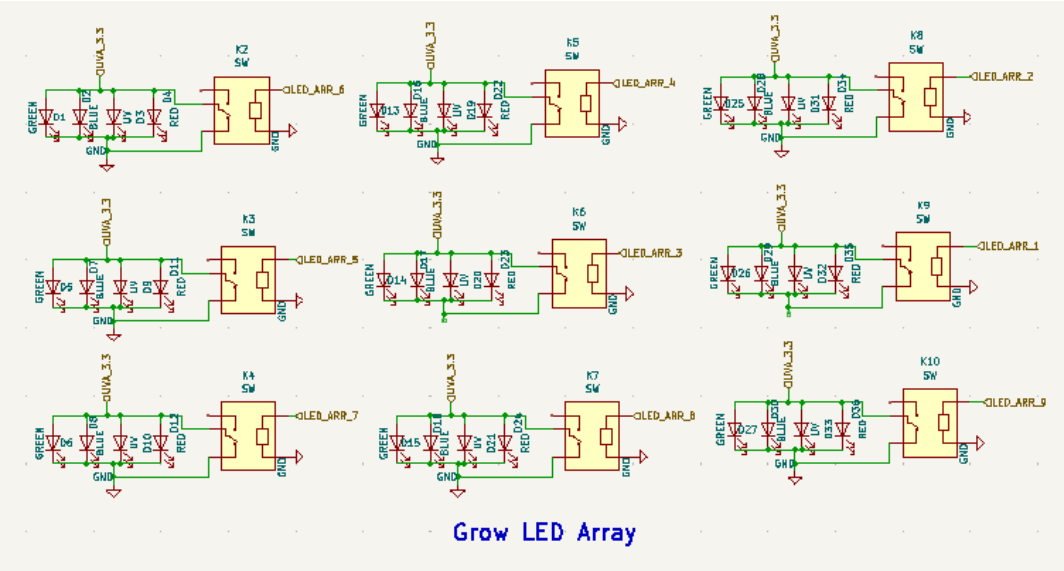

Figure 5: Overview of Grow Light Modules

To power the microcontroller used in the grow light and photosensor subsystems, the following circuit will be used. 120V AC will be drawn from the wall and be converted to 3.3V using the LM25-23B03 AC to DC converter. As a safety precaution, there is a failsafe emergency shutoff included. If the circuit were to need to be turned off in an emergency, a manual switch could be flipped to cut all power to the LEDs. Additionally, there will be a setting in the app to shut off the power via a relay as well.

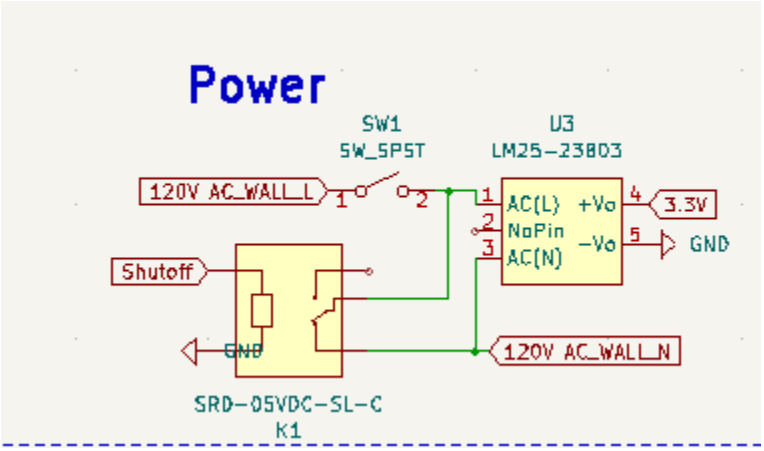

Figure 6: Power Schematics with Emergency Shut Off

In order to be able to program the microcontroller properly, a USB peripheral as well as Enable and Boot circuits have to be implemented. Referencing the design of Team 47 from Spring 2022, the boot and enable circuit will be implemented as it proved successful for programming their ESP-32 [5]. To flash the code, both the Enable and Boot buttons will be pushed.

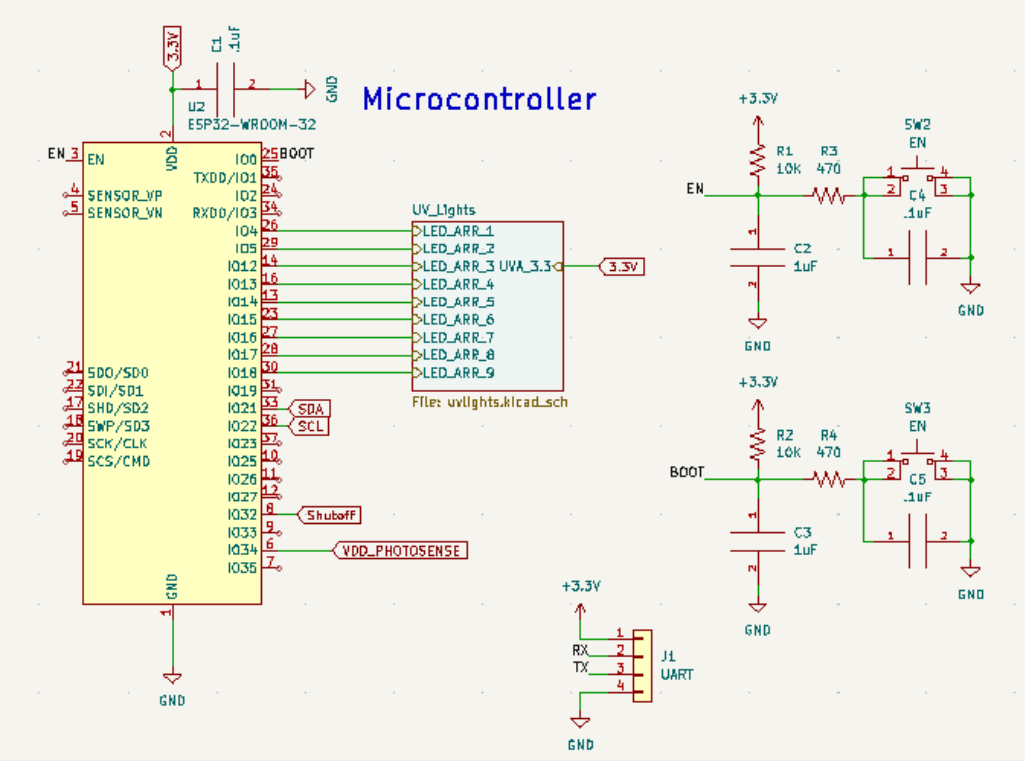

Figure 7: Microcontroller Connections

Requirements	Verification
1. The grow lights should be able to adjust to either increase or decrease the light to the plant 2. The grow light subsystem will transmit real time data on the power used (every second)	1. All of the grow light modules will be gradually switched on. The light to the plant should increase visibly. Additionally, the photosensor measurements should show an increase. 2. The data from the microcontroller will be polled for a period of 60 seconds, during which each of the grow lights will be switched on/off. At the end of this period,

	60 measurements accurately reflecting the changes should be uploaded to the server.

Table 3: Requirements and Verification for Grow Light Subsystem

Motorized Blinds

The motorized blinds subsystem will be in charge of controlling the tilt angle of the blinds for the plants to receive the desired amount of light. The system includes a ESP32 microcontroller, c stepper motor, A4988 motor driver, LM2596 buck converter and a 12V DC power supply.

The decision was made to use a stepper motor, more specifically the 28BYJ-48 stepper motor, rather than a servo motor, due to the requirements of this subsystem. In our system, the blinds will be adjusted mostly at low speed, which the stepper motor excels at as it provides high torque, reliability and precision, at a much affordable price than the servo motor [6]. This motor will be used to control the tilt of the blinds.

To stay consistent with the choice of microcontroller, this subsystem will also be operating through the ESP32 microcontroller due to the reasons discussed in the above section. The microcontroller will be in charge of receiving instructions from the application, and controlling the tilt of the blinds via the motor depending on the current state of the system.

A motor driver is implemented between the microcontroller and the stepper motor because the microcontroller operates in low current whereas the motor operates in high current. The A4988 Stepper Motor Driver was chosen as it is compatible with our stepper motor, it allows the control of maximum current output which translates to maximum voltage for the motor and that it has an over-temperature thermal shutdown system for safety measures [7].

The LM2596 buck converter and 12V DC power supply are needed in order to supply appropriate voltages to the components above.

Lastly, for the same reasons that we've included UART for the microcontroller of the UV light subsystem, they have been included in this subsystem as well.

The following figures are the schematics of the motorized blinds subsystem.

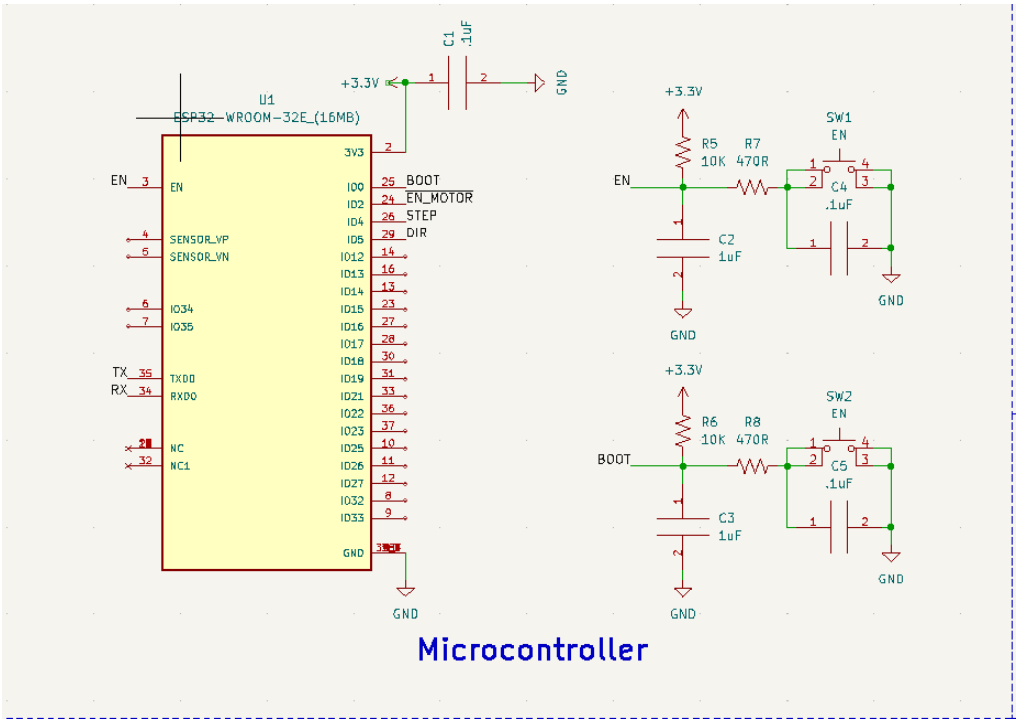

Figure 8: Microcontroller Circuit Schematic for the Motorized Blinds Subsystem

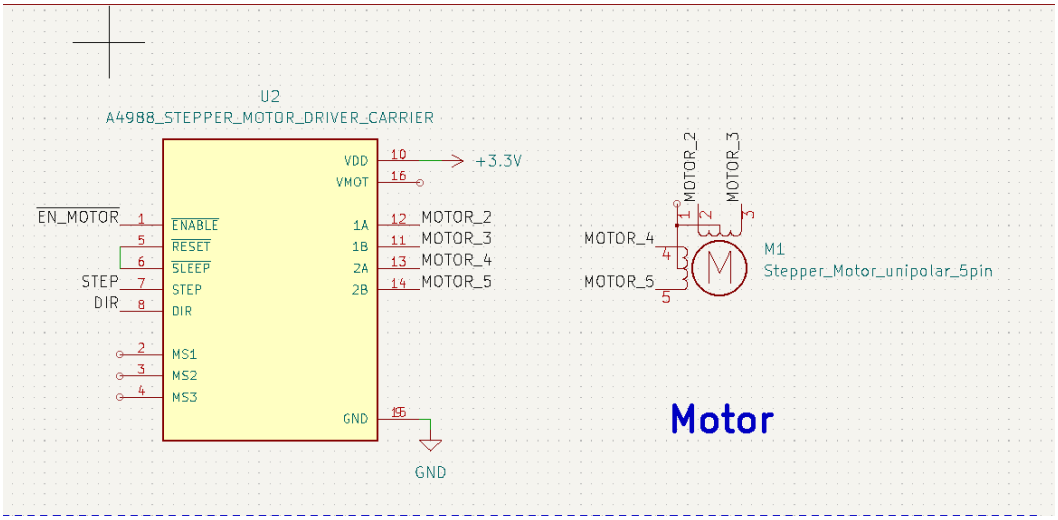

Figure 9: Motor Circuit Schematic for the Motorized Blinds Subsystem

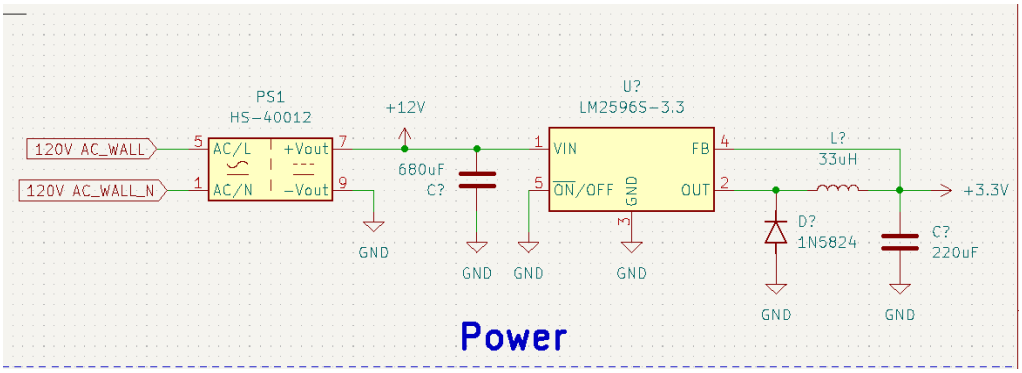

Figure 10: Power Circuit Schematic for the Motorized Blinds Subsystem

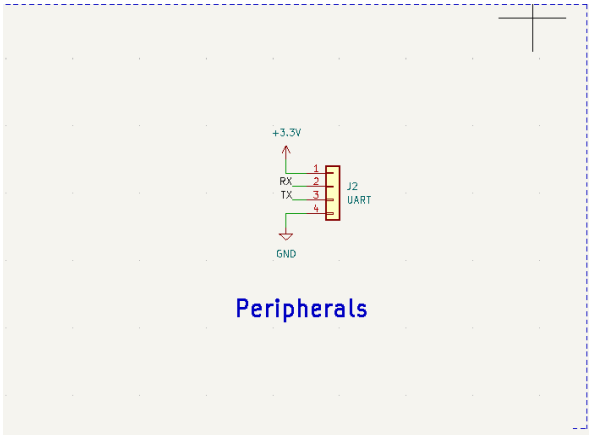

Figure 11: Peripherals Circuit Schematic for the Motorized Blinds Subsystem

The following table includes the requirements for this subsystem to function properly and methods to verify such functionality.

Requirements	Verification
1. The microcontroller should instruct the motor to rotate in the desired direction to tilt the blinds 2. The microcontroller should be able to instruct the motors to angle the blinds at a desired angle within an error no bigger than $\pm$ 2.5°, which is half the minimum increment the blinds will be adjusted at 3. The motor should start rotating within 5 seconds, for both directions, of the ESP32 receiving instruction from the application	1. Verification step for requirement 1 a. Have application to instruct ESP32* to tilt the blinds in a specified orientation b. Confirm if the motor rotates in appropriate direction to perform its instruction c. Repeat steps a~b but with opposite orientation 2. Verification step for requirement 2 a. Have application to instruct ESP32* to angle the blinds at an arbitrary angle b. Verify that the blinds are at an angle within the error margin using a protractor 3. Verification step for requirement 2 a. Have application to instruct ESP32* to tilt the blinds in one orientation for an arbitrary amount b. Start the timer, preferably a stopwatch app or a digital timer c. Stop the timer when the motor starts rotating d. Repeat steps a~c but with opposite orientation

Table 4: Requirements and Verification for Motorized Blinds Subsystem

Phone Web Application

To let a user control and monitor the entire system, a phone web application will be built. The application consists of a backend and a frontend.

1. Frontend

   - This part is where the user makes an interaction with the system. Users will enter the mode they want to run in the system, and this frontend will deliver that configuration to the backend. Also, the statistics/analysis of the system will be passed from backend to frontend and will be shown to the user in a user-friendly way. Also, the user will be able to manually control(open/close blinds, turn on/off lights) the system through frontend.

Requirements	Verification
1. The frontend server should be able to manually control the system 2. The frontend server should be running in the cloud consistently even under frequent usage 3. The frontend server should only allow verified users to control the system 4. The manual commands (turning light on/off, tilting/opening blinds) should not go beyond its limit (minimum of 0°, maximum of 90°) to not harm the components	1. List a of command lines to be tested and check whether those commands control the hardware as requested 2. Create a spamming request simulator that requests the system 10 times/s and see if all requests are responded in 500ms. 3. Create a request simulator without any identification and check if any of the requests controls the system. 4. Create a repeated manual commands simulator and check if the system ignores when the system reaches the limit.

Table 5: Requirements and Verification for Frontend

2. Backend

   - This part is where all business logic happens. Light intensity data points will be passed from the photosensor subsystem and then stored here, and the system's configuration will be passed from the frontend. Using those two data, there will be a continuously running process that calculates the need adjustment on the system. Based on that calculation, the backend will send commands to the UV light and Motorized Blinds subsystems. Also, it will summarize the statistics of the system and deliver it to the frontend in a daily basis.

Requirements	Verification
1. The backend server should accept light intensity data stably 2. The backend's k8s cluster should recover from a node failure 3. The backend should aggregate the data correctly and make a correct command 4. The backend should correctly summarize the usage in one day	1. Create a light intensity reporting simulator and see if the series of report matches with the stored data 2. Manually kill one of nodes in the cluster and checks if a new node is created 3. Create a precalculated simulation of the environment, and check whether the backend's command matches. 4. Create a mock data of one date with a precalculated summary and check if the backend's summary matches on that mock data.

Table 6: Requirements and Verification for Backend

2.3. Plots

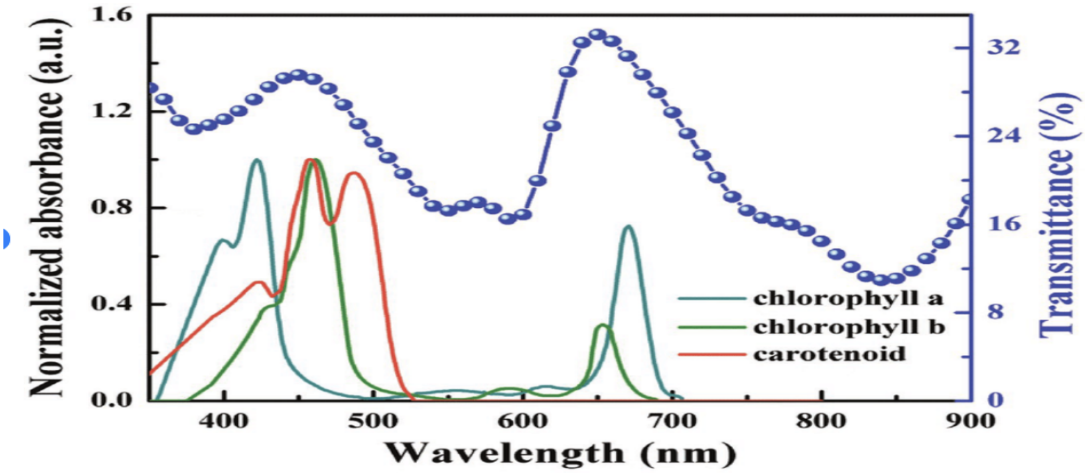

Figure 12: Wavelength vs Normalized Absorbance for Plants

Chlorophyll a and b absorption rate is critical for a plant's photosynthesis [8]. In the diagram, the absorption rate of those two are effective between 400 nm ~ 450 nm and 610 nm ~ 700 nm.

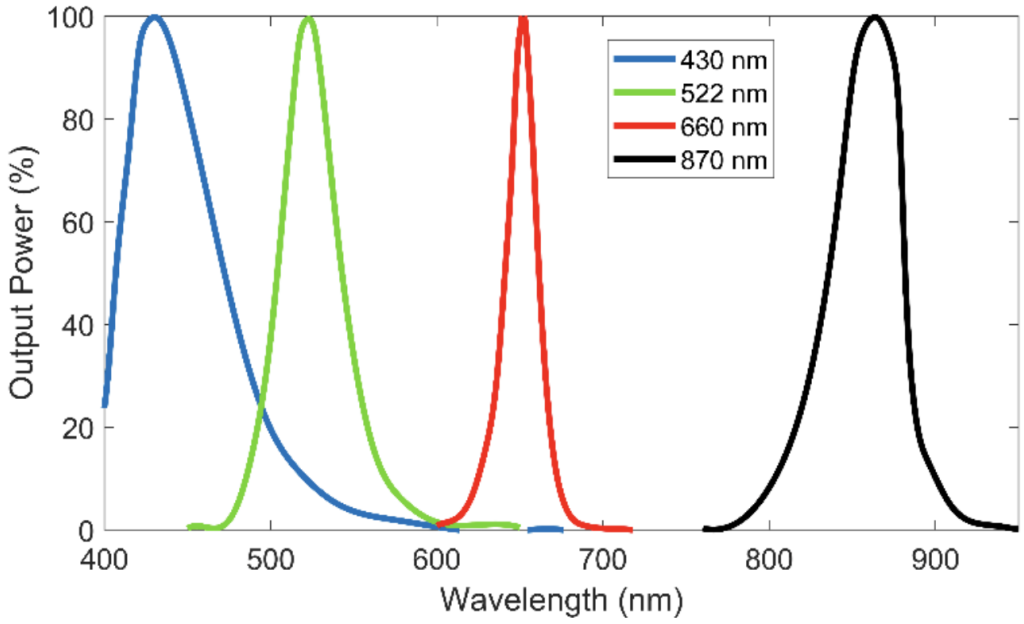

Figure 13: Wavelength vs Output Power for LEDs

According to the output power spectrum of the light-emitting diodes (LEDs) used in the sensor identified by its central wavelength in the legend, the targeted ranges (400~ 450

nm, 610 ~ 700 nm) are optimized by blue and red LEDs [9]. Therefore, LEDs are efficient enough to support plant growth

2.4. Overall Schematics

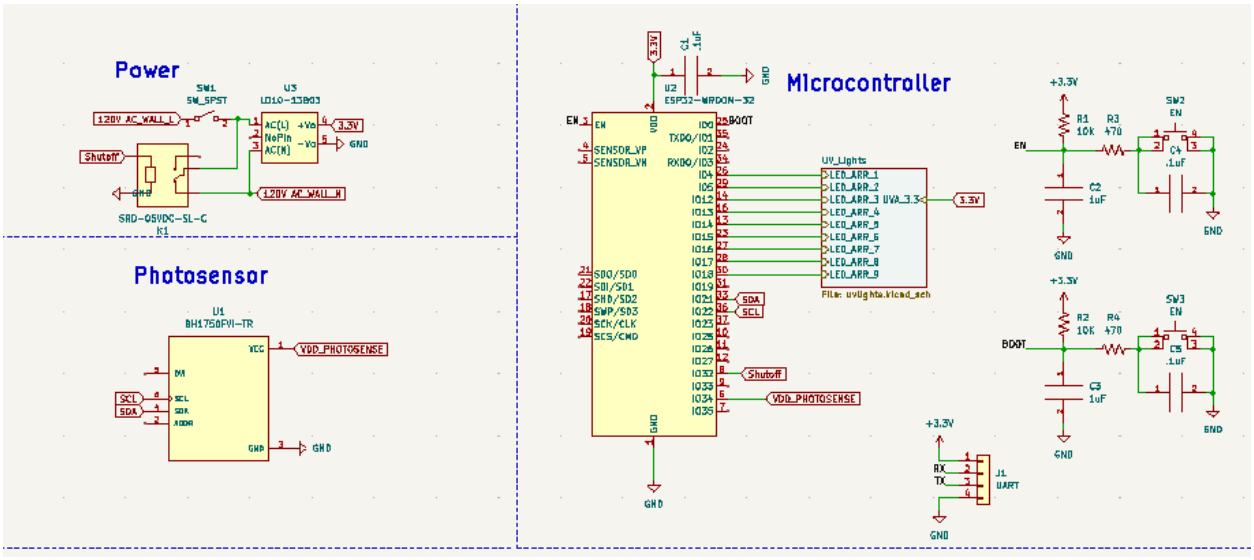

Figure 14: UV Light, Photosensor Subsystem

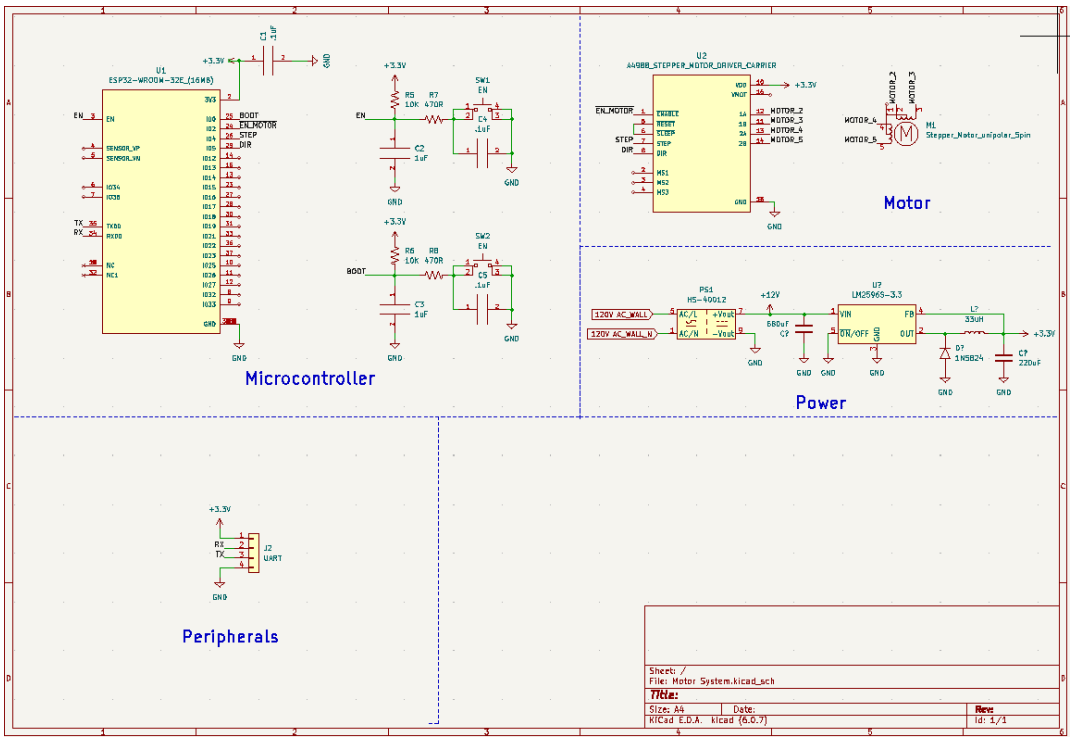

Figure 15: Overall Circuit Schematic of the Motorized Blinds Subsystem

2.5. Software Flowcharts

In the backend server, the systems will be running multiple threads to handle various jobs efficiently. In order to avoid any possible complexity in the logic flows, each thread will be allocated a clear role.

1. Data acquisition - collect data and save it into the database
2. Analyzer - Aggregate the collected data, calculate the adjustment needed to meet the light intensity target.
3. Adjuster - Adjust hardware components(motor, UVA lights) to achieve the required adjustment reported in the Analyzer.
4. StatsAnalyzer - Aggregate the stats daily

2.5.1. Data acquisition

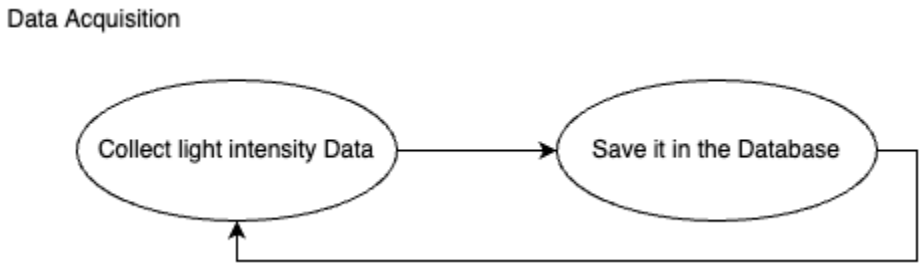

Figure 16: Flowchart for Data Acquisition

Data acquisition thread's role is straightforward. It will accept the light intensity data from the system's photosensor continuously.

2.5.2. Analyzer

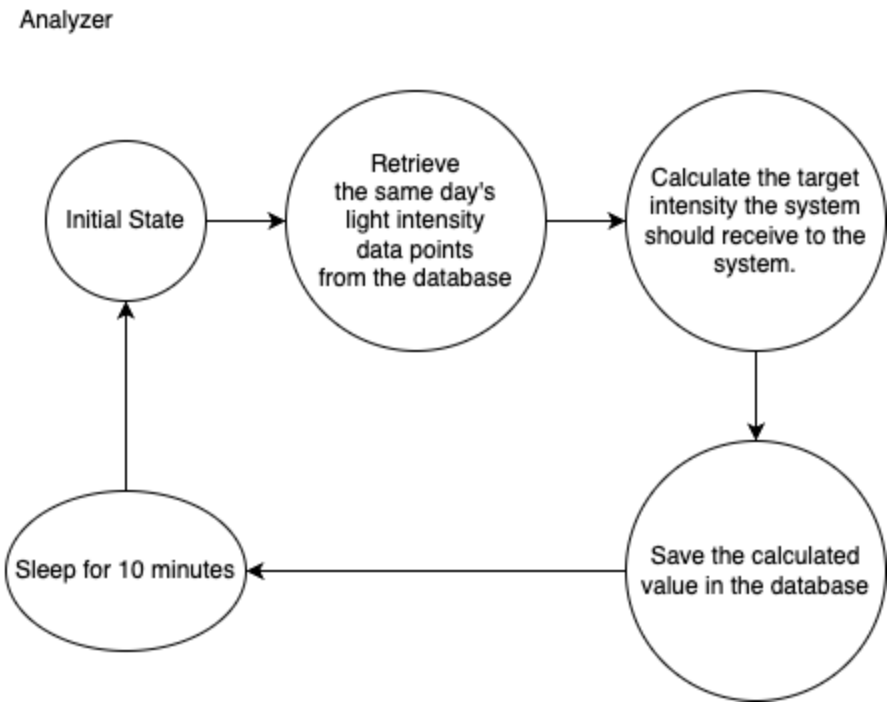

Figure 17: Flowchart for Analyzer

The Analyzer's goal is to calculate how much of light intensity the system is lacking or overloaded. The calculated number will be saved into the database so that the other thread(Adjuster) can take an action accordingly to achieve the system's goal. In order to synchronize with the Adjuster which runs every 10 minutes, the system will also wait 10 minutes after one cycle.

2.5.3. Adjuster

The Adjuster is responsible for making a decision on how to control the hardware system. Based on the calculated adjustment from the Analyzer, Adjuster takes actions to meet that adjustment. In order to prioritize energy consumption efficiency, the Adjuster first tries to minimize the artificial light source usage. If the blinds system's adjustment is not sufficient to meet the request, the system then tries to achieve the target by adjusting the artificial light source.

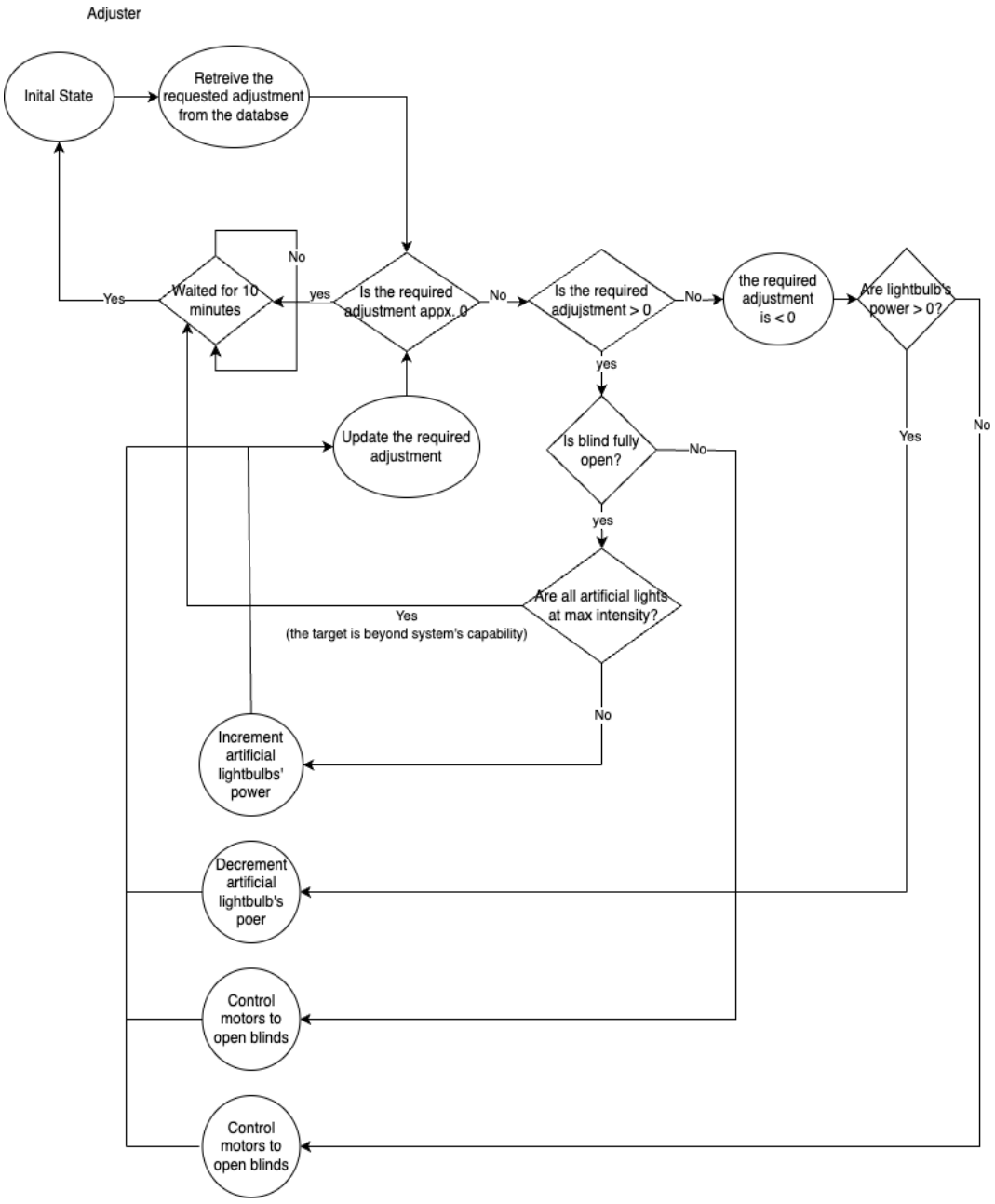

Figure 18: Flowchart for Adjuster

2.5.4. StatsAnalyzer

The StatsAnalyzer will summarize the system's statistics on its performance in a daily basis

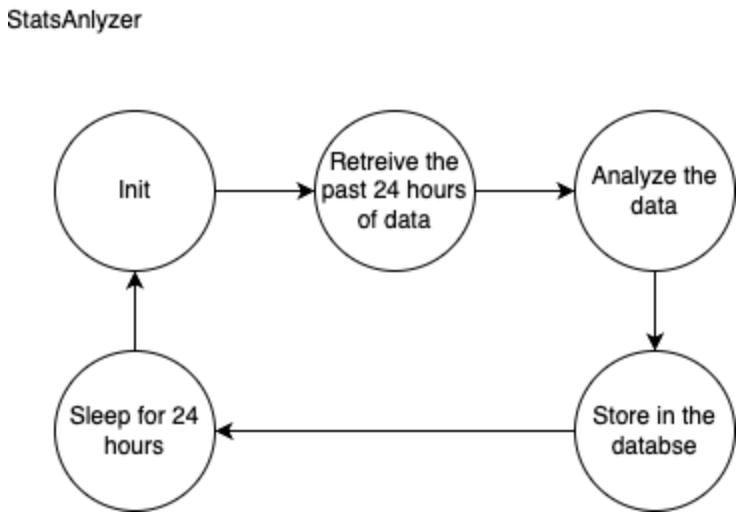

Figure 19: Flowchart for StatsAnalyzer

2.6. Tolerance Analysis

2.6.1. Hardware Component

The most critical component of the hardware aspect of our project is the grow light system composed of LEDs. It needs to be ensured that the LEDs are provided with sufficient power such that it can be turned on for a desired amount of time. Failure to do so results in diminished or in the worse case unlit LEDs which poses a great threat to the success of our project. Thus, a tolerance analysis will be performed on the amount of power our grow light system will be using, and prove that the power supply that is implemented in the project is sufficient to supply the system. To account for the worst case scenario, we will conduct all calculations assuming LEDs are at maximum brightness thus consuming maximum power.

For one grow light module, the LEDs will dissipate the following amounts of power as indicated in their datasheet

- UVA: 120 mW
- Blue 120 mW
- Green: 123mW
- Yellow-red: 72mW

The design has a total of 18 grow modules, and with 4 LEDs per module, there will be a total of 72 LEDs. The overall power dissipated by all the LEDs is then

435mW *18=7.83 W

From these calculations, the LM25-23B03 AC/DC converter should supply more than enough power. The power rating of the converter is 20W which greatly exceeds the minimum 7.83W needed for the LEDs.

2.6.2. Software

In the software aspect of things, the system should have some level of authentication to protect it from the hackers. If this system was used at an industry level, failing to maintain a desirable environment would seriously damage the plants under this system. Also, because system failure is critical, the backend server should be able to recover itself from unexpected failures.

[10]In order to recover from an unexpected failure, this system should be built using Kubernetes which is a container orchestration software. Whenever a container(node) of a server cluster fails, another container(node) will be introduced automatically to maintain a desired status of the system.

3. Cost and Schedule

3.1. Cost Analysis

UIUC's ECE AY20-21 grad students have an average starting salary of $92824. Our team consists of 3 members and each of us expect to work 10 hrs/week. A full time employee works 40 hrs / week and this project is scheduled for 10 weeks so the total human resource cost will be $92824 per year / 52 weeks * 10 weeks * / (10 hours / 40 hours) * 3 = $362013

The following table contains all the parts needed for our project.

Component	Quantity	Manufacturer	Cost/Quantity($)	Total Cost($)
BH1750 Ambient Light Sensor	1	Rohm	4.50	4.50

XZVS54S-9C UV LED	18	SunLED	2.77	49.86
XPCBLU-L1-0000-00W01 Blue LED	18	CreeLED	1.10	19.8
150080SG54050 Red/Green LED	18	Wurth Electronik	0.62	11.16
AA3528ZGSK Green LED	18	KingBright	0.62	11.16
J105D1AS3VDC.45 Relay	18	CIT Relay and Switch	1.26	22.68
SRD-05VDC-SL-C Relay	1	SONGLE Relay	2.03	2.03
10kΩ Resistor	4	EDGELEC	2.00	8
470Ω Resistor	4	EDGELEC	2.00	8
100Ω Resistor	4	EDGELEC	2.00	8
1uF Capacitor	10	KEMET	0.94	9.4
680uF Capacitor	1	KEMET	0.93	0.93
220uF Capacitor	1	KEMET	0.38	0.38
33uH Inductor	1	Bourns	0.86	0.86
1N5822 Schottky Diode	1	NTE Electronics	0.56	0.56
LM25-23B03 AC/DC Converter	1	Mornsun American	11.43	11.43
ESP32-WROOM-32E Microcontroller	2	Espressif Systems	3.00	6.00
A4988 Stepper Motor Driver	1	HiLetgo	1.98	1.98
28BYJ-48 Stepper Motor	1	HiLetgo	2.87	2.87
LM2596 Buck converter	1	Texas Instruments	3.20	3.20

12V Power Supply Adapter	1	GANGQI	9.00	9.00
Generic 4 pin connector	3	Sparkfun Electronics	1.69	5.07
Total Cost				196.87

Table 7: Component Costs

3.2. Schedule

	Major Deadlines	**Christelle**	**Sungjoo**	**Heonjang**
10/3	Design Review	Finalize PCB	Finalize PCB	Prepare a working environment (cloud setup)
10/10	**1st round PCBs**	Order Parts, Simulate Circuit	Finalize design and print 3D part	Implement a backend server
10/17		Soldering	Soldering	Soldering
10/24	Testing	Test the board, program the board to receive and send data	Test the board, program the board to receive and send data	Implement a frontend server
10/31	**2nd Round PCBs**	Order a second final board	Order a second final board	Integrate the software system with the hardware system and test
11/7		Soldering, testing	Soldering, testing	Implement statistics analyzer, soldering
11/14	**Mock Demo**	Prepare for Demo	Prepare for Demo	Prepare for Demo
11/21	**Break**			

11/28	**Final Demo**	Finalize adjustments for demo	Finalize adjustments for demo	Finalize adjustments for demo
12/5	**Final Presentation**	Finalize presentation	Finalize presentation	Finalize presentation

Table 8: Schedule

4. Ethics and Safety

This project is subject to the ACM Code of Ethics 1.3*Be honest and trustworthy* [11]. Because we are integrating all existing technologies into one system, we are destined to borrow ideas or approaches made by other people. There, we should cite those ideas properly to recognize the original author. Also, privacy should be taken seriously which is related to the ACM Code of Ethics 1.6 [11]. Our system stores user's authentication and they use histories in our database. This should be most firmly protected to avoid any privacy leakage.

This project is also subject to potential fire accidents due to UVA light systems at an industry level. Both UVA lights, motors, powers should never be overloaded to comply with the ACM Code of Ethics 1.2 *Avoid harm* [11]. In addition to that, UVA light bulbs get hot when they are turned on for a certain period of time. The users of this system should avoid touching the bulbs directly so that they do not get burned on their hands. Therefore, the housing frame in the system should encapsulate the light bulbs cluster except the bottom. Then, the users are nudged to not touch the light bulbs.

Lastly, the light bulbs cluster should be located far enough away from the plants. Plants touching the bulbs will cause fire which will damage not only the plant, but also the system itself and potentially beyond. Because this is an automated system which has a minimum human interaction, it is possible that plant growth can occur without the user's expectation.

5. References

[1] Brumfield, Robin. (1992). Greenhouse Cost Accounting: A Computer Program for Making Management Decisions. HortTechnology. 2. 10.21273/HORTTECH.2.3.420.

[2] "The visible wavelength range and its impact on plant growth", *Light Science Technologies*, https://lightsciencetech.com/visible-wavelength-range-plant-growth/

[3] Navvab, M. (2009, January). *Daylighting aspects for plant growth in interior environments*. ResearchGate. Retrieved September 30, 2022, from https://www.researchgate.net/publication/259043901_Daylighting_Aspects_for_Plant_Growth_in_Interior_Environments

[4] "LED Grow Lights for Plant Production" *OSU Extension*, https://extension.okstate.edu/fact-sheets/led-grow-lights-for-plant-production.html

[5] Xie, Ben & Goel, Pranav, & Wang, Honru. (2022) TimeTable Productivity Device. https://courses.engr.illinois.edu/ece445/getfile.asp?id=20494

[6] *Motors and Selecting the Right One*. Motors and selecting the right one. (n.d.). Retrieved September 29, 2022, from https://learn.sparkfun.com/tutorials/motors-and-selecting-the-right-one/all

[7] Allegro MicroSystems. (n.d.). *DMOS Microstepping Driver with Translator And Overcurrent Protection Datasheet*. Retrieved September 30, 2022, from https://www.pololu.com/file/0J450/a4988_DMOS_microstepping_driver_with_translator.pdf

[8] Shi, Hui & Xia, Ruoxi & Zhang, Guichuan & Yip, Hin-Lap & Cao, Yong. (2018). Spectral Engineering of Semitransparent Polymer Solar Cells for Greenhouse Applications. Advanced Energy Materials. 9. 10.1002/aenm.201803438.

[9] Duarte, Daniel & Nogueira, Rogério & Bilro, Lucia. (2019). Turbidity and RI Dependency of a Polymer Optical Fiber-Based Chromatic Sensor. Sensors (Basel, Switzerland). 20. 10.3390/s20010019.

[10] "Overview." *Kubernetes*, https://kubernetes.io/docs/concepts/overview

[11] "Code of Ethics", https://www.acm.org/code-of-ethics