SRI LANKA TECHNOLOGICAL CAMPUS
ශ්‍රී ලංකා තාක්ෂණික විශ්වවිද්‍යාලය
இலங்கை தொழில்நுட்ப பல்கலைக்கழகம்

School of Engineering

B.Sc (Hons) in Eng in Electronics

Automated greenhouse controller with IOT

August 2022

AA2129 - J.C.B. Kehelwatta

AA2122 - Thesara Wickramaarachchi

AA1378 - H.L.Adiththa Imasha

AA1424 - Nimesh Mendis

AA1391 - Dinusha Werapitiya

Department of Electronics Engineering

Abstract

The demand for a system capable of maximizing crops in an efficient and convenient way is higher than ever. The aim of this project is to make an automated greenhouse with IOT that is capable of monitoring, controlling and automating conditions of crops remotely and recording data across long periods of times. The core of the product consists of a Raspberry Pi 3b+ microprocessor, a firebase real-time database and an android app. The raspberry pi is connected with temperature, humidity, moisture and light sensors, a lighting circuit and relays that control a fan, dripping and sprinkling circuits. The firebase real-time database acts as the connection between the raspberry pi and the android app. The readings from sensors are uploaded to the firebase and then can be monitored from the app. The android app has two main modes of operation, namely manual and automatic. In the manual mode each of the relays can be switched on or off at any time by the user while in the automatic mode the system manages the circuits according to a set of conditions entered by the user. Another feature of this product is saving the readings of sensors along with time so that they can be used for projections and analysis. The multiple modes of operation and advanced features were added to make sure that the product is accessible to novices and hobbyists while still being a valuable tool to professional level farming.

Plagiarism Declaration

All the members of this group hereby declare that except where stated, all content in this report were created by our own work and extensive uses of third-party content were accurately referenced as required. We acknowledge and understand that copying another group's material for this report is classified as plagiarism and will be dealt as a serious matter.

Acknowledgements

This project would not have been possible without the support of many individuals. We would like to give our special appreciation to the project supervisor Mr. Nicoloy Gurusinghea for his guidance and support throughout the project time period. Our special thanks go to the laboratory staff who allowed us to use the equipment for the completion of our projects and indeed we are thankful for letting us borrow such an expensive Raspberry Pi from the Laboratory for the project use.

Software Used

- KiCAD 6.0
- Kodular.io
- PyCharm
- Proteus 8 Professional
- Solidworks 2021

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Greenhouse, also called glasshouse, is a building designed for the protection of tender or out-of-season plants against excessive cold or heat. (greenhouse)

Greenhouse controls the inner environment including temperature, humidity, light intensity, and soil moisture. Greenhouse Environment control system develops to control the Greenhouse environment. In our system there are three sensors inside the greenhouse, one is to measure internal humidity and temperature of the greenhouse environment, another is to measure soil moisture and the other one is to measure the light intensity of the greenhouse.

The objectives of the project

- This system is designed to control the environment of the greenhouse including temperature, light, intensity, humidity and soil moisture specific room (space).
- To Monitor and Control the environment using mobile phones.

The objective of the project is to automate the manual system of Greenhouse.

It includes the ability to maintain and control

- The temperature with fan
- Humidity with sprinkler
- Light intensity and color with light
- Soil moisture with water dripping

In traditional greenhouses, all the work are done manually. It requires a considerable amount of labor and time and causes waste of resources such as water and fertilizer, yet ends with a result of a lower harvest that might be lacking quality.

But by using IOT, greenhouses can be efficiently maintained with a minimum level of labor and resources for a maximum harvest. Most importantly, it can be remotely controlled. Greenhouse automation lets tailor an environment for the crops that can offer reactive solutions to outside influences that may otherwise jeopardize the yield of crops as well as the pleasure of home gardening.

Chapter 2

Method

The purpose of using this app is control the greenhouse, Receive real time values of greenhouse conditions and send data to Firebase real time data base. Here we have created a very fast, simple and easy method to use. It has a feature that we can control conditions that plants need.

2.1 Components Used

1. Raspberri Pi
2. TEMT6000 Photo-Transistor
3. DHT11
4. Capacitive Soil Moisture Sensor V1.2
5. TIP120
6. PCF8591
7. Relays
8. Variable Resistors
9. Capacitors, Resistors, LEDs
10. Connector Cables and Sockets

2.1.1 Raspberry Pi

The heart of our project is the Raspberry PI 3 which is a single board computer which contains SOC.Raspberry PI 3 serves as a budget desktop and the core consists of the Arm processor. Some of its features include 40 GPIO pins, CPU speed of 1.2GHZ, 1 GB RAM of memory, Inbuilt Wi-Fi. The Raspberry PI 3 will cost the same as its antecedent, but with an additional feature of Bluetooth and Wi-Fi. With its built-in wireless connectivity, the new Raspberry PI 3 is clearly placed as a low-cost hub for Internet of Things (IOT) devices, or flexible, low-cost basis on new types of connected gadgets.We have used Python programming as our programming language.

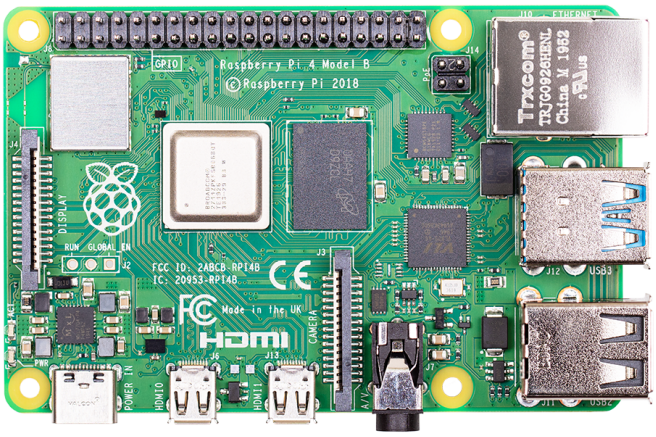


Figure 2.1: Raspberry PI B3+

2.1.2 TEMT6000 professional light sensor module

The TEMT6000 ambient light sensor is an ordinary sensor which is very similar to a LDR (light-dependent resistor). Essentially, the TEMT6000 can be defined as a silicon NPN epitaxial planar photo-transistor used for sensing the visible spectrum of light. It is sensitive to visible light much like the human eye and has peak sensitivity at 570 nm. TEMT6000X01 has analog output and is packaged in a small surface mount package.

Figure 2.2: TEMT6000 professional light sensor module

Technical Specifications

Operating Ratings		
Vcc Range	Typical	3.0V – 5.5V
Collector Light Current	20 lux	10uA (typical)
	100 lux	50uA (typical)
	Max Bright Conditions	3.8V (typ with Vdd = 5V)
Spectral Detection	Detection Range	440nm-800nm
	Peak Detection	570nm
Angle of Half Sensitivity		± 60°
Dimensions	$L * W(PCB)$	$14.5 * 8mm(0.57 * 0.32")$

Table 2.1: Technical Specifications

2.1.3 DHT11

DHT11 temperature and relative humidity

- The sensor's operating voltage ranges from 3.5V to 5.5V.
- DHT11 has a sampling period of more than two seconds with a standby current of 60uA and an output current of 0.3mA
- The sensor also has a 4-pin single row pin package

Figure 2.3: DHT11 Sensor

Additional relative humidity features

- Output signal: digital signal via a single-bus
- Measuring range: At 50℃ 20–80% humidity readings
- sensing element: polymer resistor
- Interchangeability: fully interchangeable
- Long-term stability: $<\pm$ 0.5% RH/Yr
- Accuracy: At 25℃ is $\pm$ 5% RH
- Lag: $<\pm$0.3% RH
- Resolution: 1% RH

The Temperature specifications

- Temperature resolution: 1℃
- Repeatability: $\pm$1℃
- Operating range: 0-50℃

- Accuracy: +-2.0℃

2.1.4 Capacitive Soil Moisture Sensor V1.2

- Works from 3.3v to 5.5v
- Output analog Voltage: 0 3.0 VDC
- Supports 3-Pin Sensor interface
- Interface: PH2.0-3P

Figure 2.4: Capacitive Soil Moisture Sensor V1.2

A capacitive moisture sensor works by measuring capacitance changes caused by the changes in the dielectric. It does not measure soil moisture directly (as pure water does not conduct electricity well), instead it measures the ions that are dissolved in the moisture. Capacitive measuring basically measures the dielectric that is formed by the soil and the water is the most important factor that affects the dielectric.

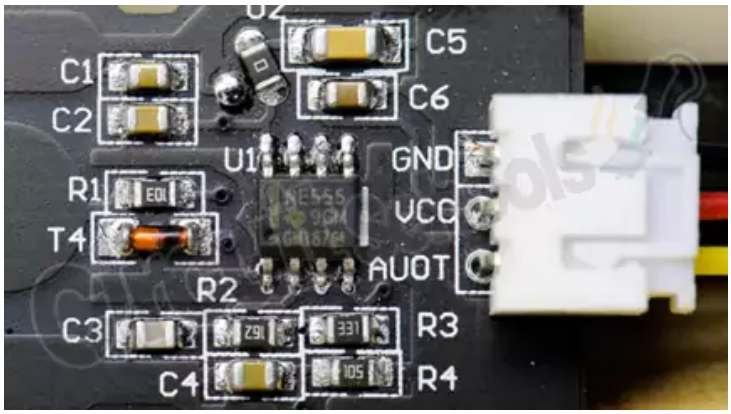


Figure 2.5: Moisture Sensor Circuit

The capacitance of the sensor is measured with the help of a 555 timer-based circuit that creates a voltage which is directly proportional to the capacitor inserted in the soil. We then measure this voltage by use of an Analog to Digital Converter which produces a value that we represent as percentage of soil moisture.

2.2 Hardware Development

The true challenge for us was the design. It was very important that our design is a unique one as well as a better one. In order to achieve that, we spent weeks coming up with different kinds of circuit designs and analyzing them for further optimizations.As mentioned above we used the TEMT6000 professional light sensor modular as its output voltage is proportional to the light intensity and we had to take account the availability of it in Sri Lankan stores. Though it was available at the time we made the decision, later when we were going to buy, it was out of stock locally and at the end we had to order it online.

In our design, what we basically did was dividing the 0-5V output of the sensor into 4 equal ranges and amplifying them separately so at the end each range would give a 0-5V output. Each range line would go five circuit elements consequently along the branch. After feeding those four inputs to four analog input pins of the micro-controller we summed the four voltages up so that in the end the 0-5V is effectively amplified into 0-20V only using low voltages.

2.2.1 Controller

In our design, what we basically did was Converting supplied 12V to 5V and 3.3 V. Also It was able to Control the Blue and Red Intensity using PWM method. furthermore three relays were

added to control two watering methods and Fan.

2.2.2 Light

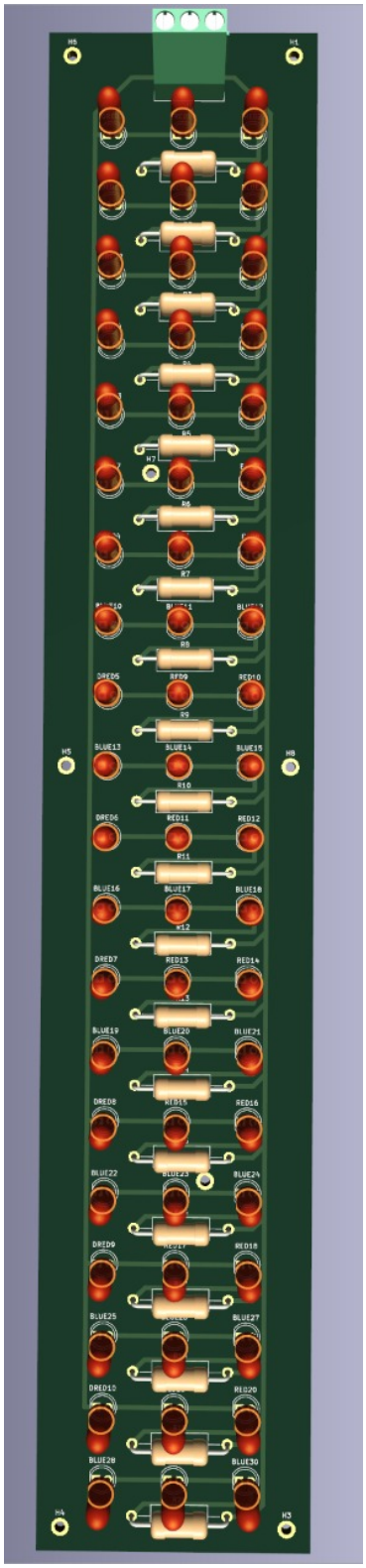


Figure 2.6: 3D View of Light

LED Grow-lights and Indoor Plant Growth

A grow-light is an artificial and generally electric source of light, designed to stimulate plant growth. Grow-lights are used in applications where there is either very less naturally occurring light or where supplemental light is required for plant growth. They are used for horticulture, indoor gardening, plant propagation and food production, including indoor hydroponics and aquatic plants. Although most grow-lights are used on an industrial level, they can also be used in households. A range of bulb types can be used as grow-lights such as incandescent, fluorescent, high intensity discharge (HID) and light-emitting diodes (LED). Indoor flower and vegetable growers typically use high-pressure sodium (HPS) and metal halide (MH) HID lights, but the LEDs are replacing metal halides due to their efficiency and economy.

Different wavelengths of light have different effects on plants which can be summarized as follows

Color of Light	Wavelength (nm)	Effect on Plants
Ultra Violet	250-380	Long exposure is detrimental
Violet	380-445	Color, Taste and Aroma
Blue-Green	445-570	Growth and Maturity
Yellow	570-590	Long exposure slows down growth
Red	590-720	Vegetation and Plant Mass
Far Red	720-1000	Flowering and Reproduction

Table 2.2: Effects of colors of light on plants

LED Grow light

The proposed LED Grow-light system,12V input is the main source of power for the LED array. The light system consists with 30 red LEDs,30 blue LEDs, resistors and LED drive.

LED Array

Here, the LED Array has been built as a series-parallel combination of LED Grow-lights of 2 colors viz. Red and Blue. Controlled intensity of light of each of colors has a direct impact on the plants.

Current Controller

The current controller is a Proportional-Integral (PI) control system in which reference input is the current settings given by the user based on readings of TEMT6000 professional light sensor module, feedback signal is the measured LED current and the controller output is the duty ratio for Pulse Width Modulated (PWM) switching of the DC-DC power converter.

LED Grow light Calculations

Input = +12V

$Red \rightarrow 1.7V, 20mA(5mm)$

$Blue \rightarrow 3.0V, 30mA(5mm)$

Number of bulbs in the series = 3

Resistor that could be used per series

$$R_{Red} = \frac{V}{I} = \frac{12 - (1.7 * 3)}{0.02} = \frac{12 - 5.1}{0.02} = \frac{6.9}{0.02} = 345\Omega$$

$$R_{Blue} = \frac{V}{I} = \frac{12 - (3 * 3)}{0.03} = \frac{12 - 9}{0.03} = \frac{3}{0.03} = 100\Omega$$

Power consumption per series

$$P_{Red} = V * I = 5.1 * 20 * 10^{-3} = 0.102W$$

$$P_{Blue} = V * I = 9 * 30 * 10^{-3} = 0.27W$$

Total power consumption by bulbs

$$P_{Red} = 0.102W * 10 = 1.02W$$

$$P_{Blue} = 0.27W * 10 = 2.7W$$

$$P_{Total} = 1.02W + 2.7W = 3.72W$$

Total power consumption by resistors

$$P_{345\Omega} = \frac{V^2}{R} = \frac{6.9^2}{345} = 0.138W \rightarrow 0.138 * 10 = 1.38W$$

$$P_{100\Omega} = \frac{V^2}{R} = \frac{3^2}{100} = 0.09W \rightarrow 0.09 * 10 = 0.9W$$

$$P_{Resistors} = 1.38W + 0.9W = 2.28W$$

Total power consumption by the system

$$P_{Total} = P_{Bulbs} + P_{Resistors} = 3.72W + 2.28W = 6W$$

2.3 Software Development

For this project the software development portion consisted of developing the android app, the script for the Raspberry Pi and the firebase realtime database. The raspberry pi script was developed using Python language and the android app was developed using kodular.io. As the firebase database had to be configured to be compatible with both the script and the app, all the 3 units required development with a large amount of coordination. Both Github and google drive were used extensively to track versions and compatibility. In the subsections below, each aspect of software development is explored in greater detail.

2.3.1 Python Script

For this project a python script was used in the raspberry pi. The reason for selecting python was great support for the selected hardware and runtime not being a big concern due to relatively short script. Establishing connection with firebase real-time database and handling communication was done with the pyrebase module, a simple python wrapper for Firebase API. The Raspberry Pi 3b+ model does not have built in analog reading. Therefore, a PCF8591 analog to digital convertor was used with the moisture and light sensors to get readings of 8-bit accuracy using I2C communication. For this script, Input and output ports were referenced with the BCM pinout convention. The whole main portion of the script repeats on a loop unless interrupted by user. At the start of the loop new values are read from sensors and uploaded to firebase. After that instructions and conditions are retrieved from firebase/android app to

decide the process for the rest of the script.

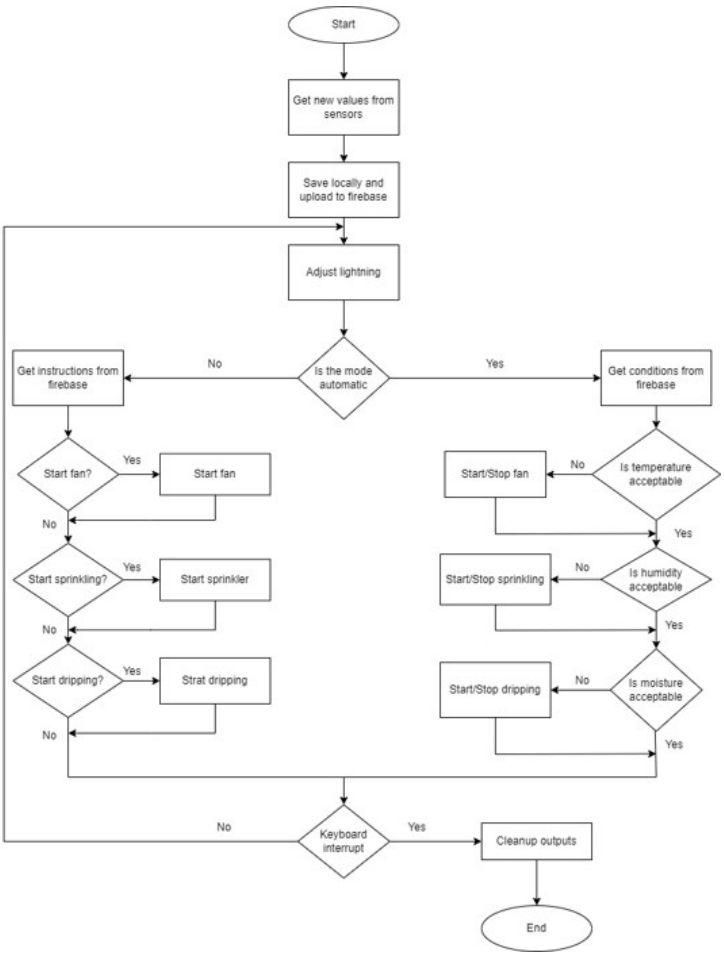


Figure 2.7: Flowchart for script

For saving readings from sensors locally, built in CSV module for python was used. Using the module, every sensor value that gets uploaded to the firebase realtime database also gets saved on a local file in comma separated value format. This can be used to make analysis about how conditions of the greenhouse affect crops in a given period of time.

1	Date	Time	Temperatu	Humidity	Light	Moisture
2	31/07/202	10:28:16	30	95	478.43	5.88
3	31/07/202	10:31:58	30	95	501.96	5.1
4	31/07/202	10:32:40	30	95	513.73	4.71
5	31/07/202	10:33:09	30	95	513.73	5.49
6	31/07/202	10:33:38	30	95	474.51	6.27
7	31/07/202	10:33:41	30	95	482.35	4.71
8	31/07/202	10:34:22	30	95	505.88	5.88
9	31/07/202	10:34:51	30	95	521.57	2.75
10	31/07/202	10:34:57	30	95	513.73	5.1
11	31/07/202	10:35:14	30	95	513.73	5.1
12	31/07/202	10:35:43	30	95	529.41	5.1
13	31/07/202	10:36:02	30	95	564.71	5.49
14	31/07/202	10:36:06	30	95	556.86	4.71
15	31/07/202	10:36:15	30	95	560.78	4.31
16	31/07/202	10:36:49	30	95	192.16	5.1
17	31/07/202	10:37:30	30	95	200	5.1
18	31/07/202	10:37:39	30	95	219.61	4.71
19	31/07/202	10:38:20	30	95	541.18	0
20	31/07/202	10:38:24	30	95	545.1	5.1
21	31/07/202	10:38:30	30	95	545.1	6.27
22	31/07/202	10:39:11	30	95	513.73	5.49
23	31/07/202	10:39:20	30	95	454.9	7.84
24	31/07/202	10:39:39	15	175	388.24	6.27
25	31/07/202	10:40:11	30	95	466.67	4.31
26	31/07/202	10:40:52	30	95	505.88	5.49

Figure 2.8: Sensor values saved in a few minutes

The Raspberry Pi was configured to make the script run automatically on startup, ensuring that the user does not need to login to the raspberry pi or need other hardware.

2.3.2 Mobile Application

Login

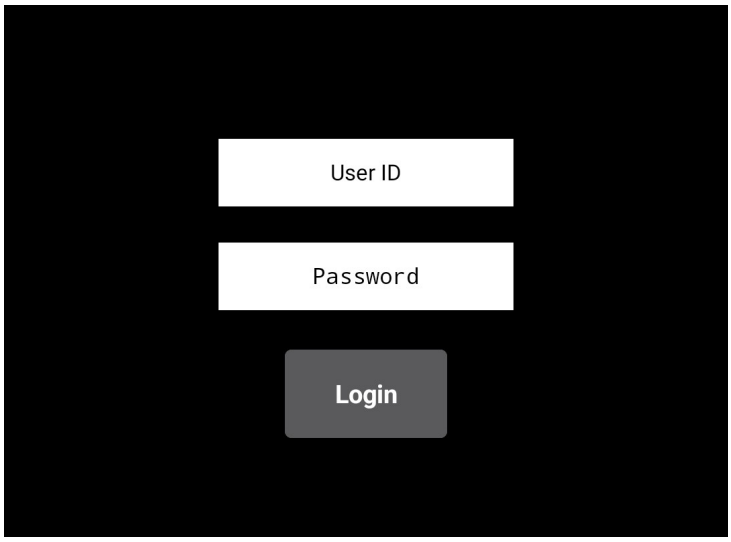


Figure 2.9: Login Page

This is how display first page when user log in to App

How to Login

First of all, User must enter the correct email and password given to you in the correct fields. After recording everything correctly, click the login button below. If the user enters something wrong, it will be marked as "login failed" in RED. Then by entering everything correctly again, user can easily log into this app.

Manual Mode

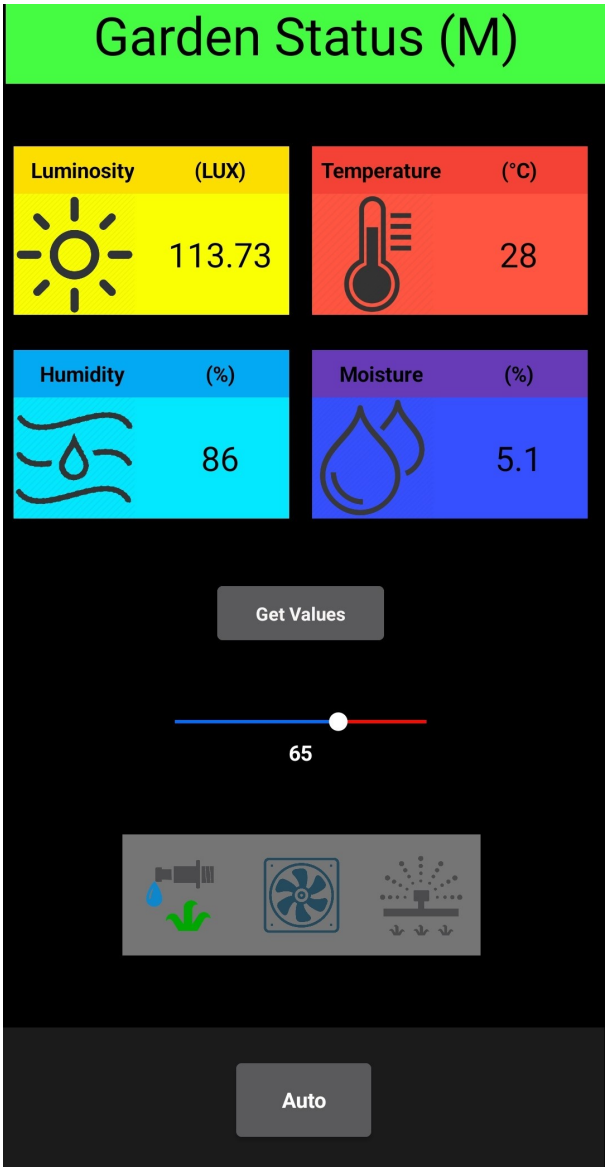


Figure 2.10: Manual Page

This is the Manual screen. Here user can easily see Temperature, Luminosity, Humidity and Moisture. User can get the current real-time value by clicking the "Get Value" button below. After that user can see the slider that controls the light and the 3 buttons that control the conditions. These 3 buttons are used to control Dripping, Fan and Sprinkling. Finally, the "Auto" button is installed to switch to Auto mode.

How to Operate Manual Mode

By clicking the get value button as mentioned above, User can get updated real-time data related to that moment. The color of the light can be changed to blue and red by the slider.

By moving the slider to the left, the red color of the light will be activated and by moving the slider to the right, the light will be activated as the blue color. The three buttons below are shown in gray when they are OFF State, and when they are colored they are ON State. The first button is used for Dripping. By turning it On, the plant roots in the greenhouse will be supplied with water through the pipes installed below.The second button is used for the fan. By turning it On, the fan in the greenhouse system will turn on and bring down the temperature when it rises above the level required by the system. The last button is used for sprinkling.By turning it On, the water needed by the greenhouse system starts to spray into the system by the sprinklers at the top.

Auto Mode

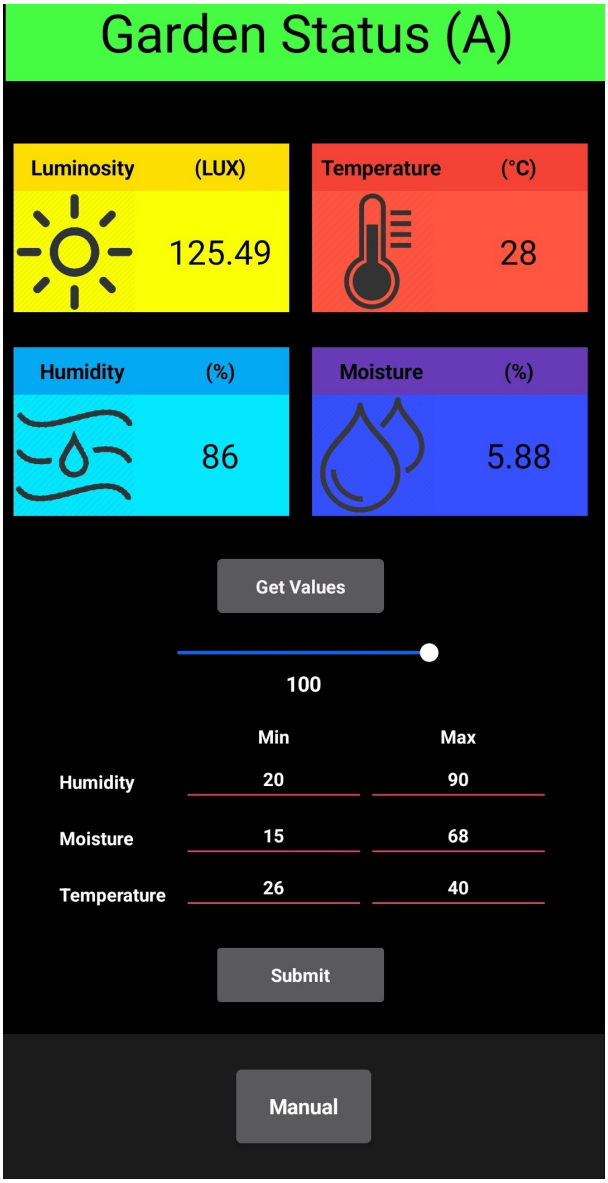


Figure 2.11: Auto Page

This is Auto screen. Here Also user can easily see Temperature, Luminosity, Humidity, Moisture and User can Easily get the current real-time value by clicking the "Get Value" button below same as manual mode. But the main difference in this auto mode is that there is no control of the situation by the user. All conditions are controlled automatically based on the data provided by the user. By clicking the Manual button at the bottom, you can easily enter the Manual mode from the Auto mode.

How to Operate Auto Mode

As mentioned above, everything is controlled automatically in this Auto mode. In this screen, User can see that Humidity, Moisture and Temperature are set to three conditions and a maximum and minimum value can be specified for each of them. There, the user has the ability to enter the maximum and minimum values related to the three mentioned conditions related to the plant grown by the user.After specifying the necessary data, by clicking the submit button below, all the data will be included in the Firebase Real-time database. Thus, the system records the data in the database and automatically controls all conditions equal to or greater than the respective minimum value and equal to or less than the maximum value. When the humidity is lower than the minimum value, the sprinkling option is automatically activated and it is maintained between the specified values, and as soon as the maximum value is reached, the sprinkling option is deactivated. The fan also automatically turn on when the temperature exceeds the maximum value and automatically turn off when it reaches the minimum value. Also, when the moisture condition is lower than the minimum value, the Dripping option will be activated automatically and when it reaches the maximum value, the Dripping Option will be automatically deactivated and the value will be maintained between maximum and minimum.

2.3.3 Real-time Database

In the system, the firebase realtime database works as the medium of communication between the android app and the raspberry pi script. It consists of fields that represent conditions and instructions for the operation of device. All the data from raspberry pi is uploaded to the realtime database before it can be viewed from the app. Likewise, instructions put in the app by user must go through firebase and then retrieved by the script running on raspberry pi. Because of this, it is possible to override the app and give instructions from the firebase realtime database if desired by user.

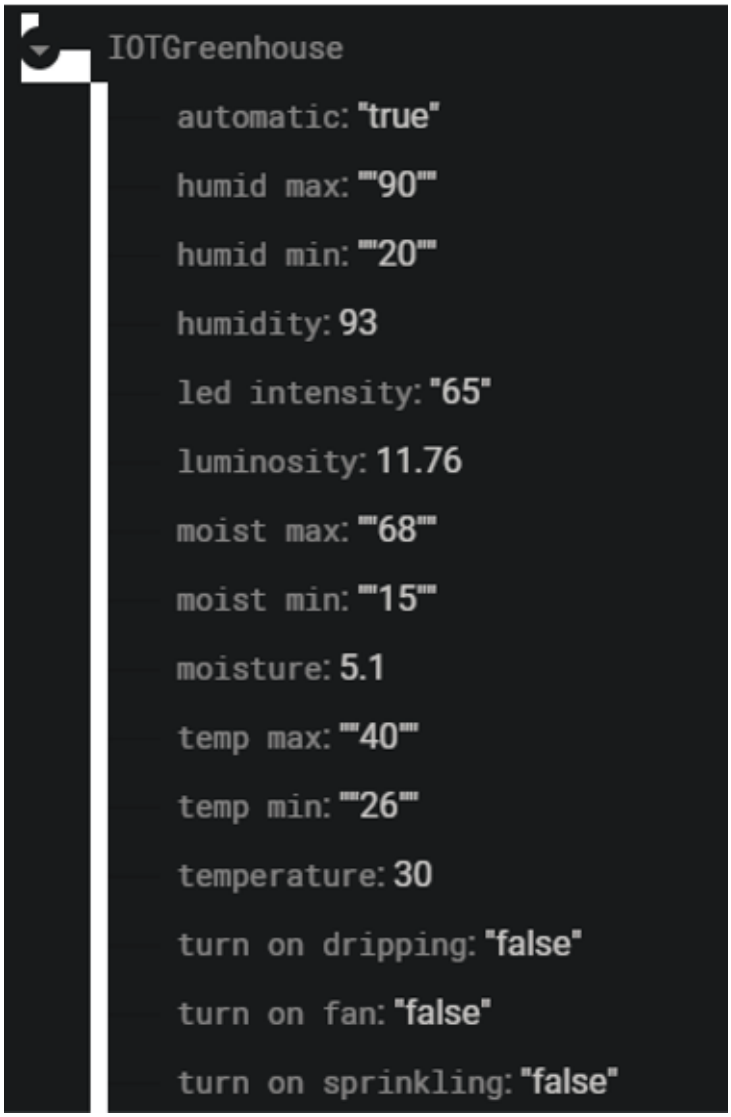


Figure 2.12: Firebase fields

The user authentication for the system is also handled using firebase. Only the users added in the firebase can login to the android application and make changes to the system.

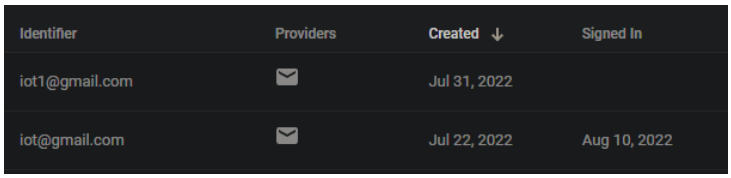


Figure 2.13: Firebase authenticated users

2.4 Prototyping and Fabricating the PCB

Designing the PCB was done using KiCAD software as recommended. Before making the Prototype, it was thoroughly tested on bread board ensuring the correct functionality of it. For the testing, we made a small Green House with a LED Light, Fan, and two watering methods. This was important because our system was has three sensors and each of then should have control system. After ensuring the functionality the layout of the double sided PCB was designed using Kicad software.

2.5 Physical Structure

The wooden box is made out from Gliricidia (Gliricidia Sepium) with the dimensions of 24 * 16 * 4.5 in inches. The reasons for using Gliricidia are it is light in weight and no permission is required for transport.

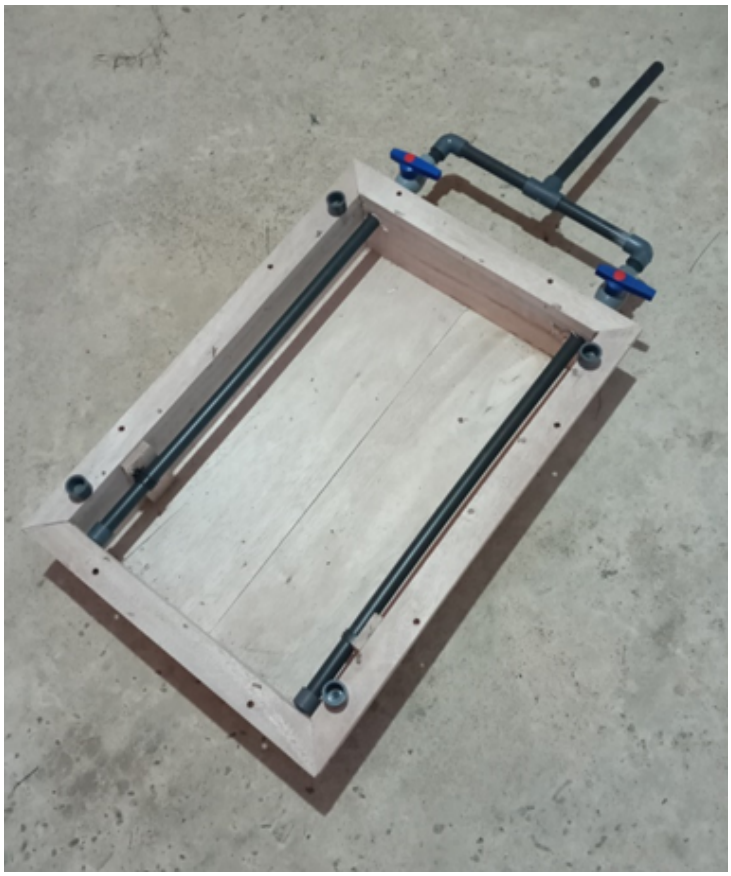


Figure 2.14: Physical Structure Wooden box

The height and the width of the PVC structure are 26 and 16 in inches respectively

Figure 2.15: Height of Physical structure

The green net is used as an alternative for glass

Figure 2.16: Physical Structure

Chapter 3

Discussion and Results

In traditional cultivation, labor work is vastly involved. Also, the resources such as water, fertilizer are used in mismanage or wasted since such resources were used only based on the experience which were gained by the years of cultivating but without proper data and management. As a result of that, though a lot of manual workload is involved at the end the framers get comparably a little amount of harvest which might be damaged. In addition to that, the farmers always have to visit the cultivation. It consumes a considerable amount of effort and time. Plus, the weather conditions are uncontrollable and unavoidable when it comes to outdoor farming. Moreover due to extreme weather conditions, majority of cultivation and also the harvest is wasted. For those drawbacks, "Automated greenhouse with IOT" is a better solution since the cultivation can be monitored and controlled in remote. As well as, the resources can be used efficiently since it can be used based on data. We managed to complete the building of the project up to a breadboard prototype stage. The PCB circuit was fully designed but could not be physically printed due to the ongoing situation at country. The prototype itself is capable of complete function on a real greenhouse if installed properly. We also gathered sample readings of conditions of crops and circuitry across testing sessions. These data would be valuable to innovate and optimize the design more in the future.

Chapter 4

Conclusion

In conclusion, "My greenhouse" project of greenhouse with IOT is a point where traditional agriculture meets the IOT technology. A smart greenhouse helps to reduce and manage the demand of labor and also the sources for example water and fertilizer for a more amount of harvest since greenhouses protect cultivation from extreme weather conditions such as temperature, humidity, light intensity and soil moisture. It helps to increase the, The IOT productivity as well. Most importantly, the greenhouse is able to be managed remotely. The IOT-enabled process provides accurate information. It mitigates the traditional load of data gathering, refining and determining the accuracy percentage. My greenhouse is able to integrate with other sustainable and reliable solution for both home gardening and commercial cultivation.

Chapter 5

Budget

While we selecting components we had to consider about several things. Those are availability of the items and its costs. We have struggled to find the items and however we have managed to buy them. One of the main things that effected to the budget is PCB printing. There was no place in Sri Lanka that prints PCBs. So, we had to order PCBs from China. However, because of the ongoing situations in Sri Lanka we couldn't get the printed PCBs. But we add the costs of PCB because it should be in the budget.

Description	Qty	Unit Price	Net Amount	Total
Capacitive soil moisture sensor V1.2	1	Rs. 420.00	Rs. 420.00	
TEMT6000 professional light sensor module	1	Rs. 440.00	Rs. 440.00	
DHT11 temperature and relative humidity	1	Rs. 540.00	Rs. 540.00	
GL-12 project board Breadboard solderless	1	Rs. 360.00	Rs. 360.00	
5.08mm pitch 2-pin 2-way screw terminal	2	Rs. 30.00	Rs. 60.00	
PC812 optocoupler	2	Rs. 30.00	Rs. 180.00	
5V DC miniature relay 10A 250V AC	6	Rs. 90.00	Rs. 540.00	
Solder flux injection type RMA-223 10ml	1	Rs. 220.00	Rs. 220.00	**Rs. 2,760.00**
IN5817	6	Rs. 8.00	Rs. 48.00	
BC547	6	Rs. 5.00	Rs. 30.00	
LED 3mm yellow green	6	Rs. 1.50	Rs. 9.00	
LED 3mm pink(D)	6	Rs. 4.00	Rs. 24.00	
1KΩ 1/4W	10	Rs. 0.50	Rs. 5.00	
510Ω 1/4W	16	Rs. 0.50	Rs. 8.00	
LM2596 DC-DC step down	2	Rs. 360.00	Rs. 720.00	
3 pin wire terminel(L)	8	Rs. 22.00	Rs. 176.00	
Tip120	4	Rs. 23.00	Rs. 92.00	**Rs. 1,112.00**
LED Blue (5mm)	40	Rs. 3.00	Rs. 120.00	
LED Red (5mm)	40	Rs. 4.00	Rs. 160.00	
Jumper wire 20cm M to M	1	Rs. 250.00	Rs. 250.00	
Jumper wire 20cm M to F	1	Rs. 260.00	Rs. 260.00	
PCF8591 module	2	Rs. 600.00	Rs. 1,200.00	
GL-12 project board Breadboard solderless	1	Rs. 360.00	Rs. 360.00	**Rs. 2,350.00**
PCB (5$ for 5 PCBs printed from china)				
Controller circuit	5	Rs. 360.00	Rs. 1,800.00	
LED Growlight circuit	5	Rs. 360.00	Rs. 1,800.00	**Rs. 3,600.00**
1/2" Plastic solenoid valve 12V DC	2	Rs. 1,485.00	Rs. 2,970.00	
10mm*1yd heat srink tube	3	Rs. 75.00	Rs. 225.00	
5mm*1yd heat shrink tube	2	Rs. 34.00	Rs. 68.00	
Thread brass mist fog nozzle spray sprinkler	2	Rs. 200.00	Rs. 400.00	
Hot melt glue stick	3	Rs. 35.00	Rs. 105.00	
330ohm 1/2W	10	Rs. 1.00	Rs. 10.00	**Rs. 3,778.00**
500W Power Supply	1	Rs. 2,600.00	Rs. 2,600.00	**Rs. 2,600.00**
Wood - 4' x 8"	1	Rs. 280.00	Rs. 280.00	
Wood - 14' x 4"	1	Rs. 420.00	Rs. 420.00	
Labour cost for the wooden box	1	Rs. 2,500.00	Rs. 2,500.00	
1/2" pipe	1	Rs. 240.00	Rs. 240.00	
1/2" ball valves	2	Rs. 385.00	Rs. 770.00	
1/2" caps	6	Rs. 30.00	Rs. 180.00	
1/2" L	2	Rs. 30.00	Rs. 60.00	
1/2" T	7	Rs. 35.00	Rs. 245.00	
Green net	2 m	Rs. 680.00	Rs. 1,360.00	**Rs. 6,055.00**
Total cost				**Rs. 22,255.00**

Table 5.1: Budget

References

Design of a Scalable and Optimized LED Grow-light System Driven by a High Efficiency DC-DC Power Converter (2019). UNIVERSITY OF MINNESOTA. URL: https://conservancy.umn.edu/bitstream/handle/11299/206166/Joshi_umn_0130M_20227.pdf?sequence=1 (visited on 08/13/2022).

DHT11 Humidity Temperature Sensor (2020). Mouser electronics. URL: https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf (visited on 08/13/2022).

How to Interface PCF8591 ADC/DAC Analog Digital Converter Module with Raspberry Pi (2022). Circuit Digest. URL: https://circuitdigest.com/microcontroller-projects/interfacing-pcf8591-adc-dac-module-with-raspberry-pi (visited on 08/13/2022).

TIP120 Darlington NPN Transistor: Pinout, Datasheet, Circuit [Video] (2022). Apogeeweb Semiconductors. URL: https://www.apogeeweb.net/circuitry/TIP120-darlington-npn-transistor.html (visited on 08/13/2022).

Appendix A

Controller

A.1 Controller - 3D View

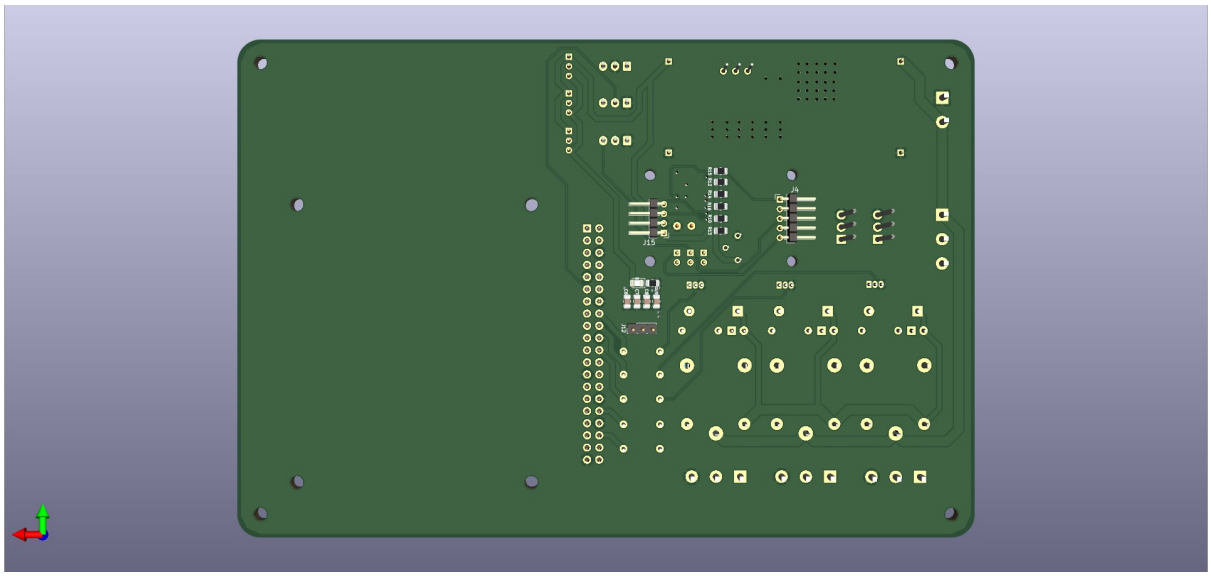


Figure A.1: Front View

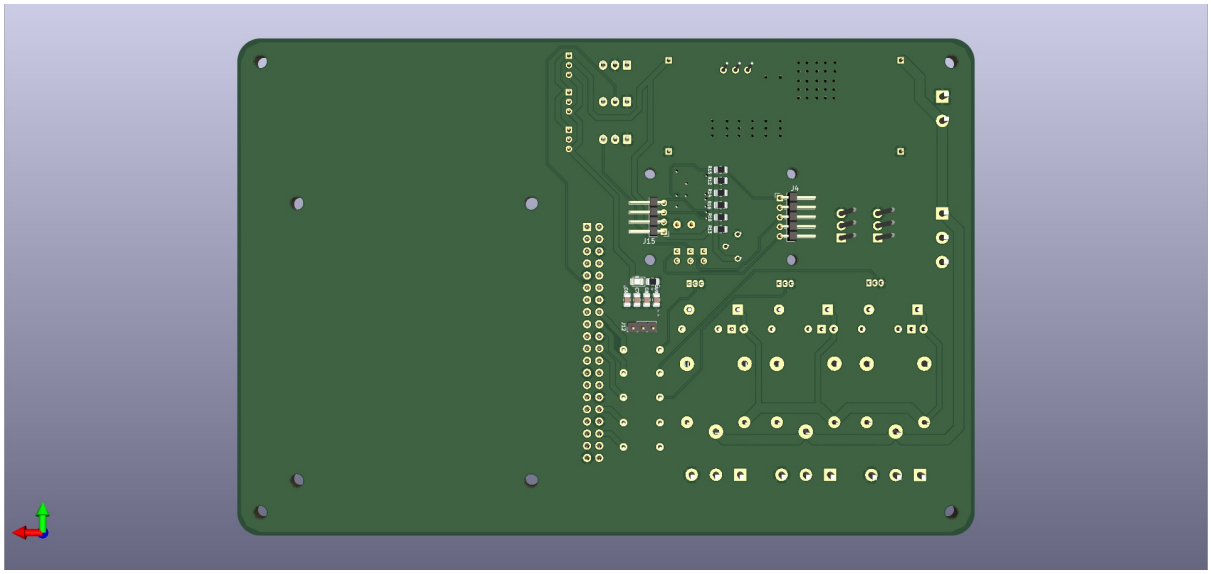


Figure A.2: Rear View

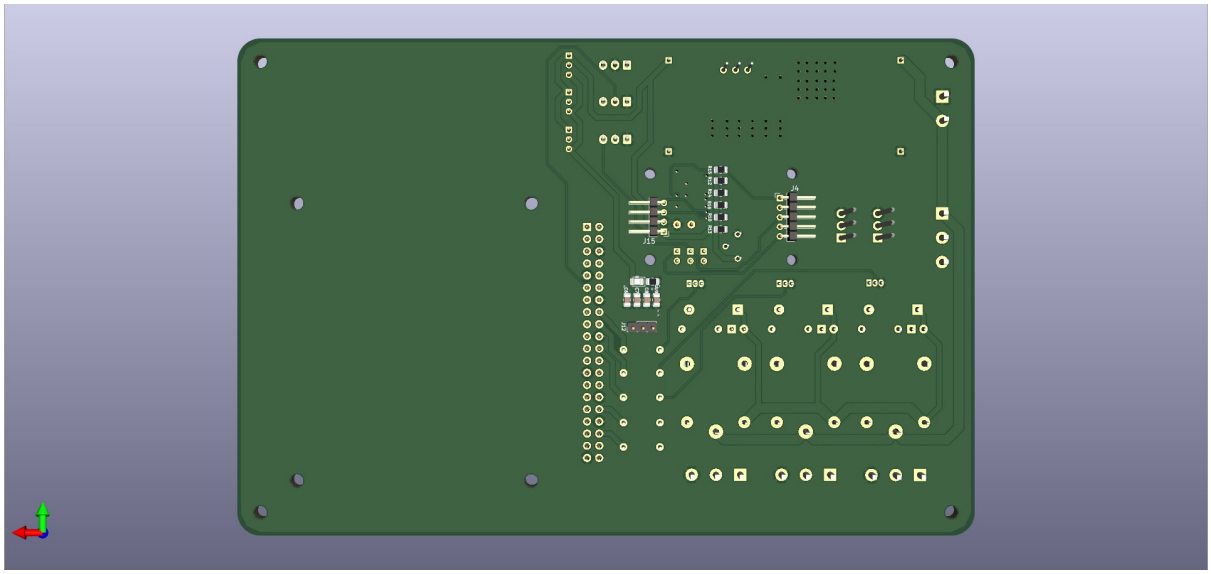


Figure A.3: Left Side View

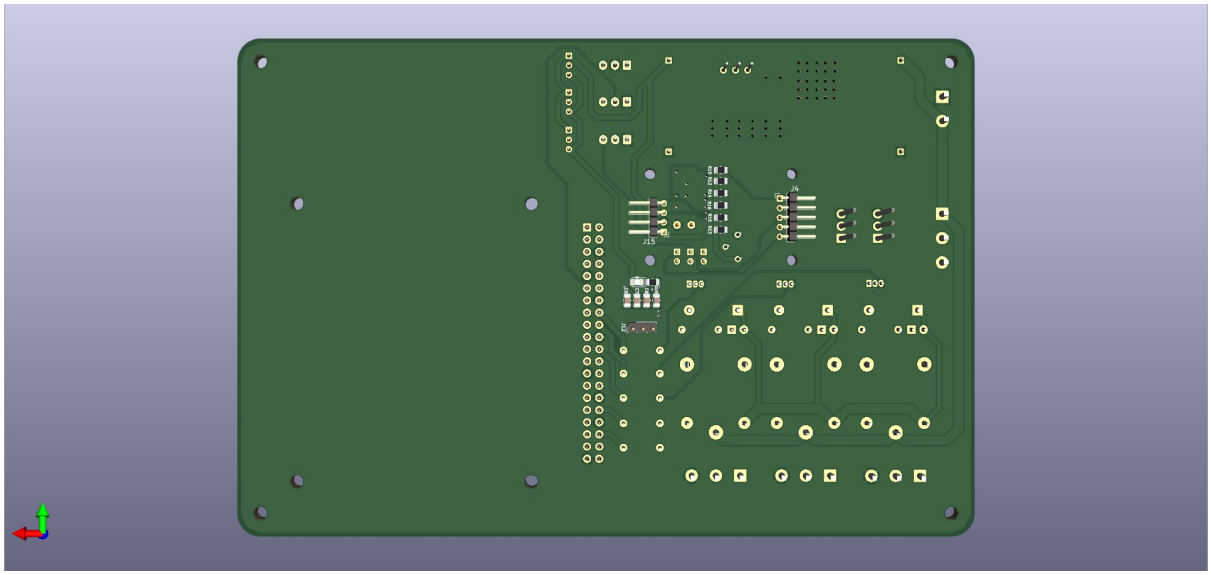
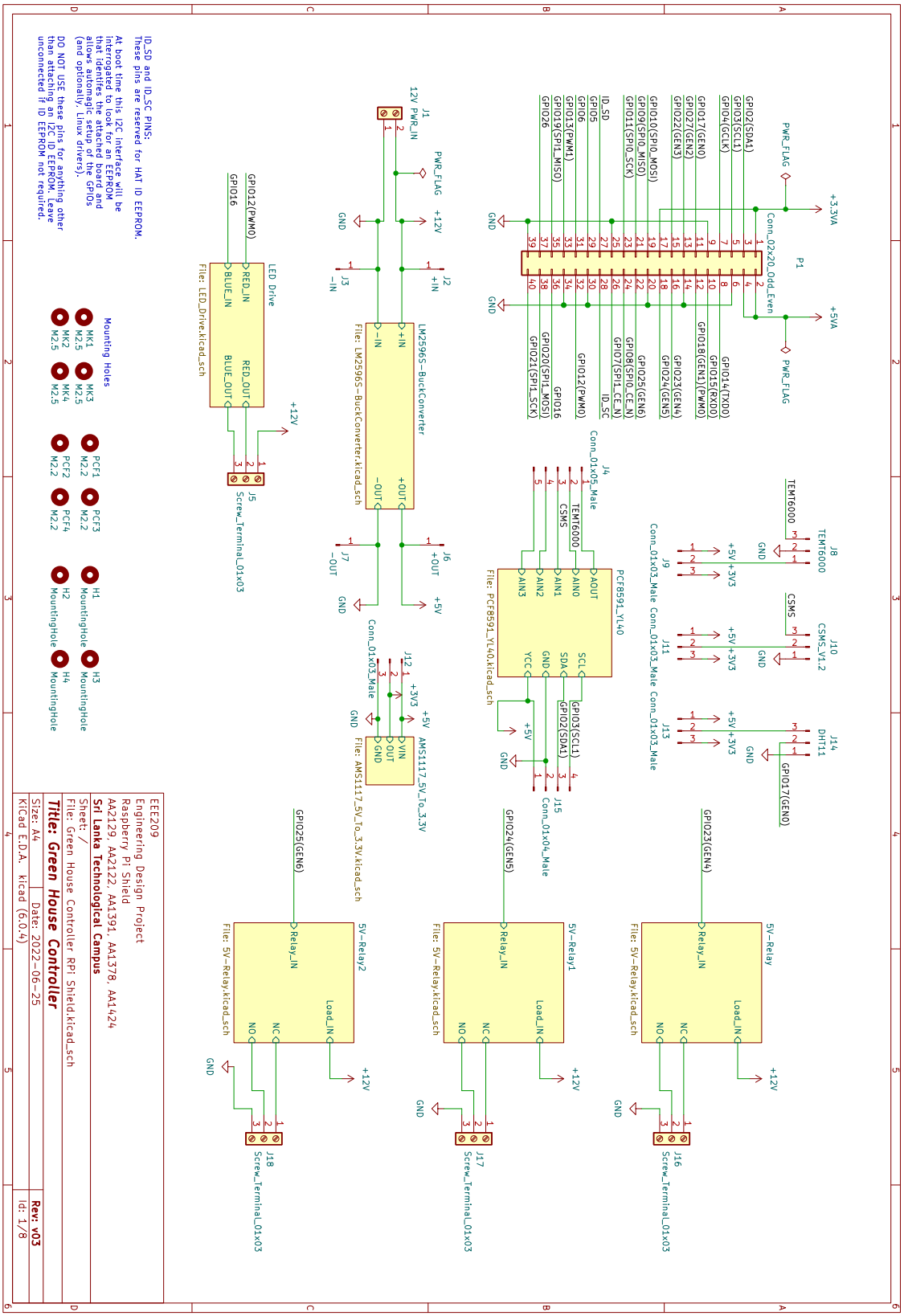
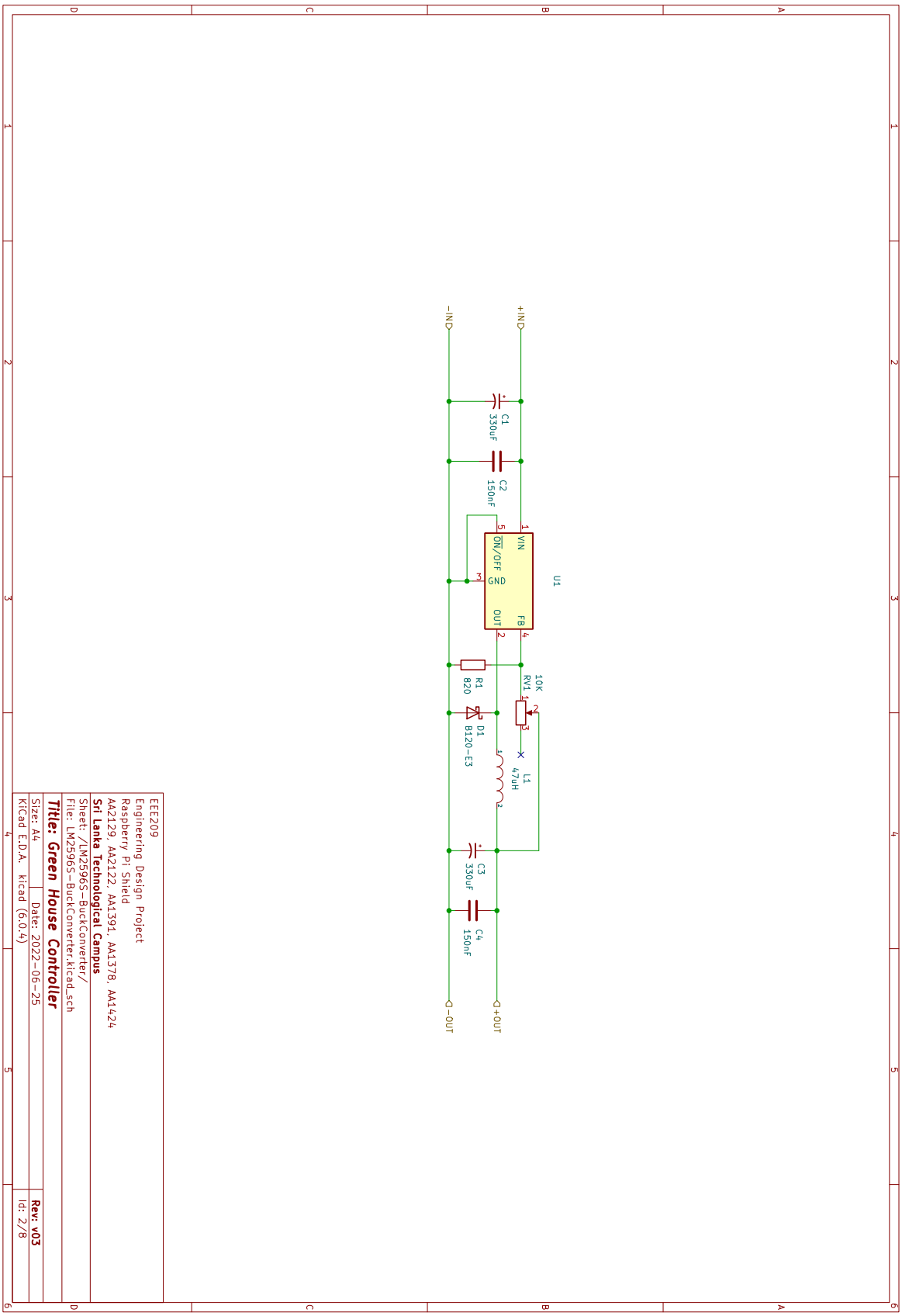
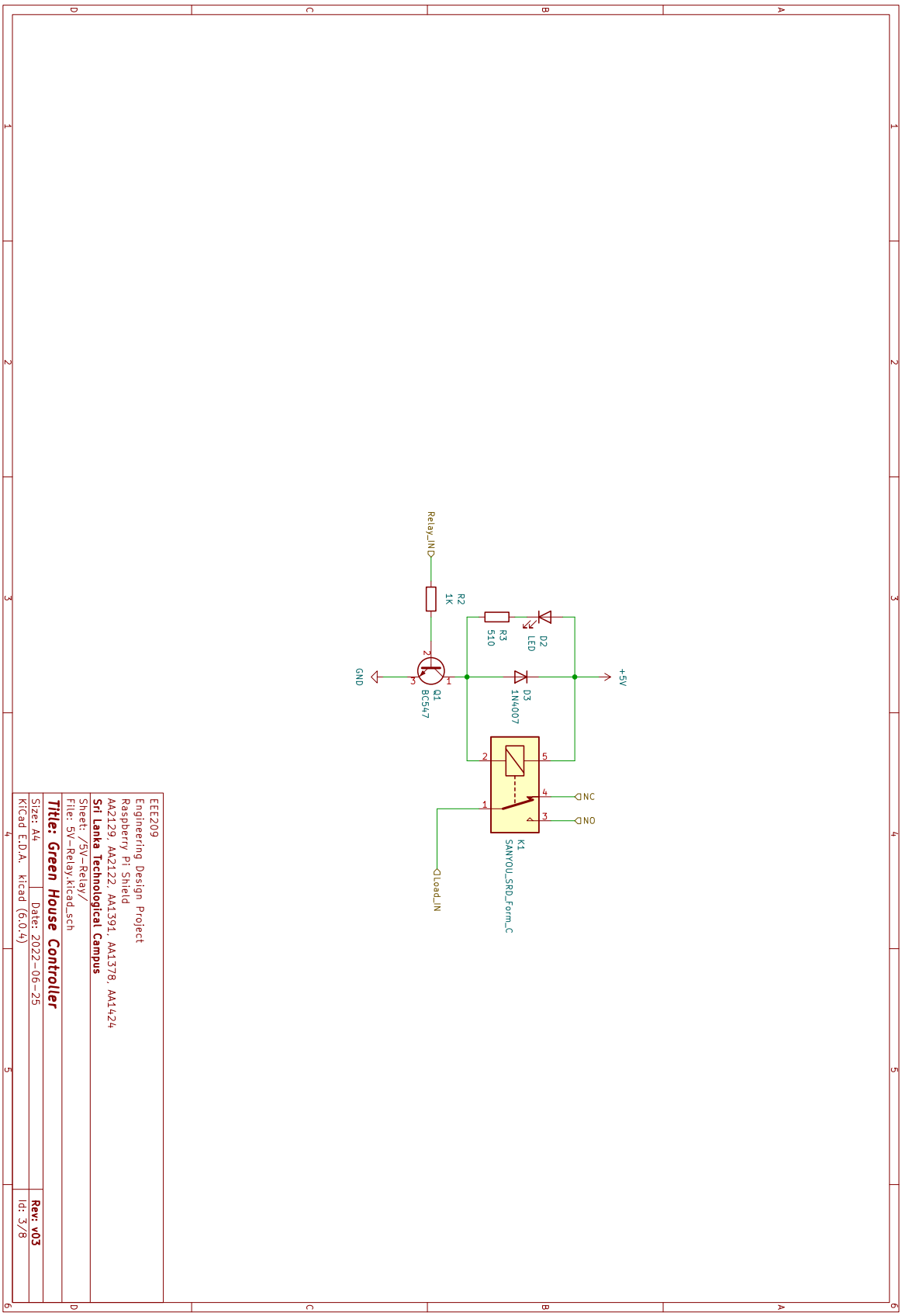


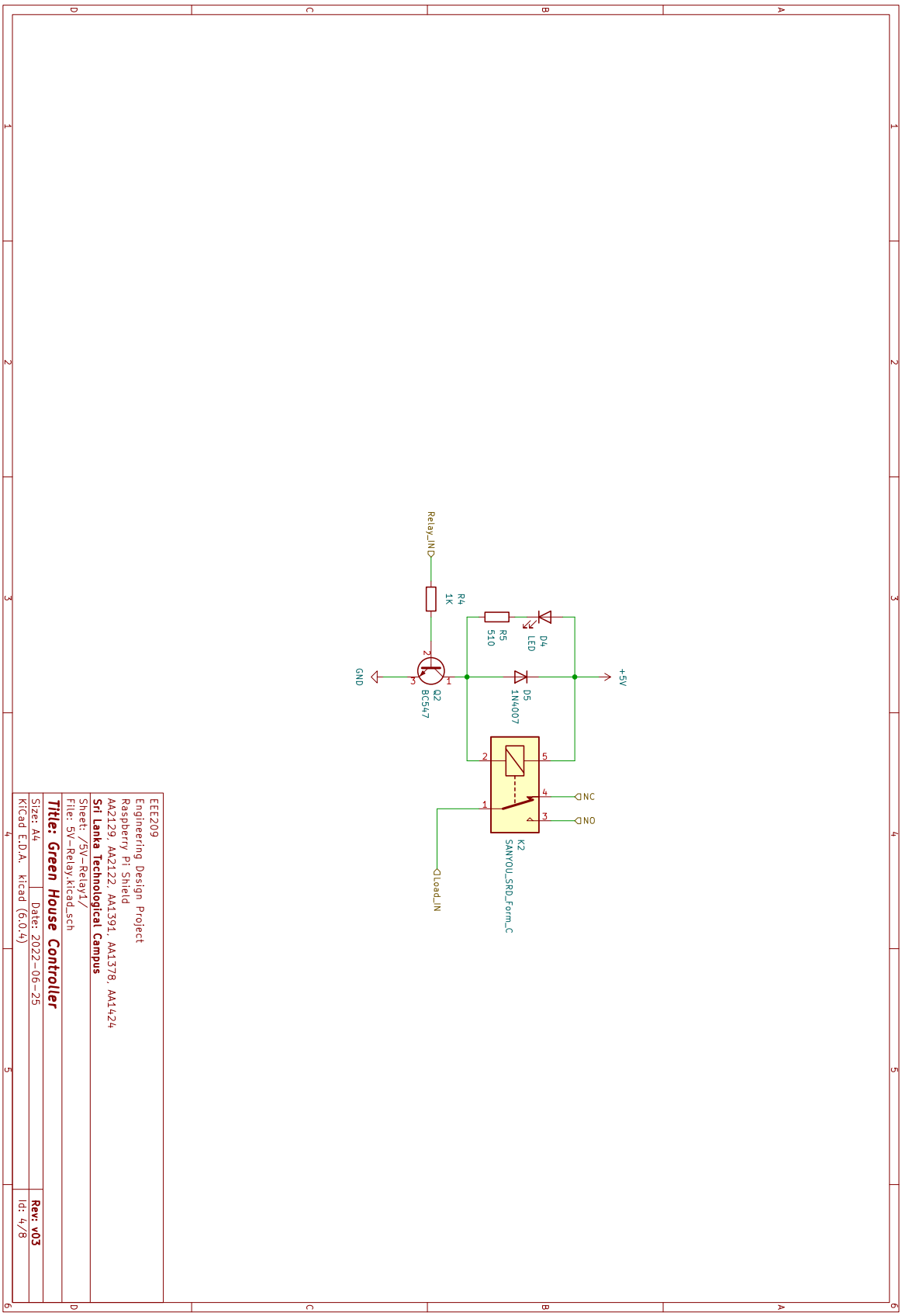
Figure A.4: Right Side View

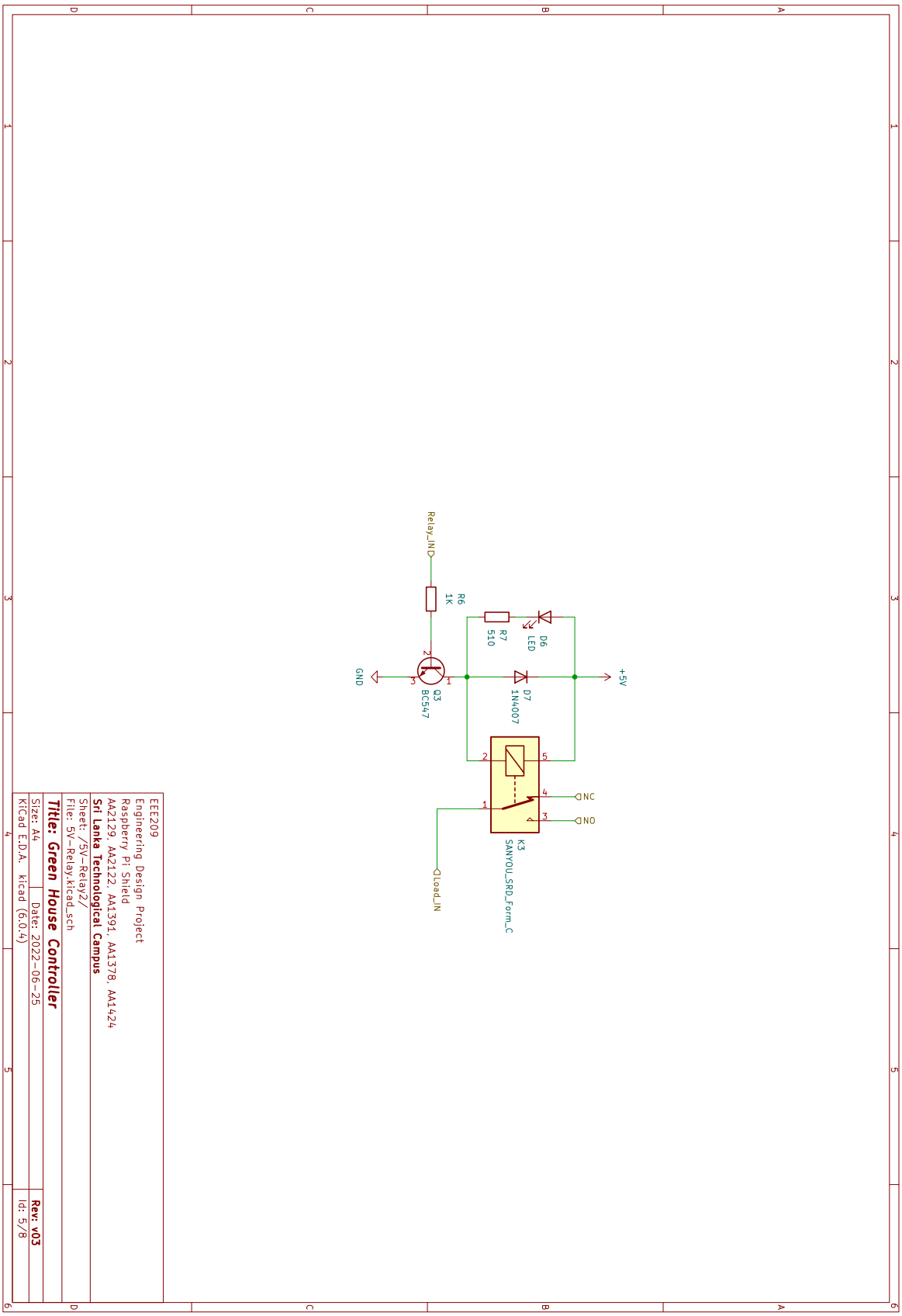
A.2 Controller - Schematic Diagram

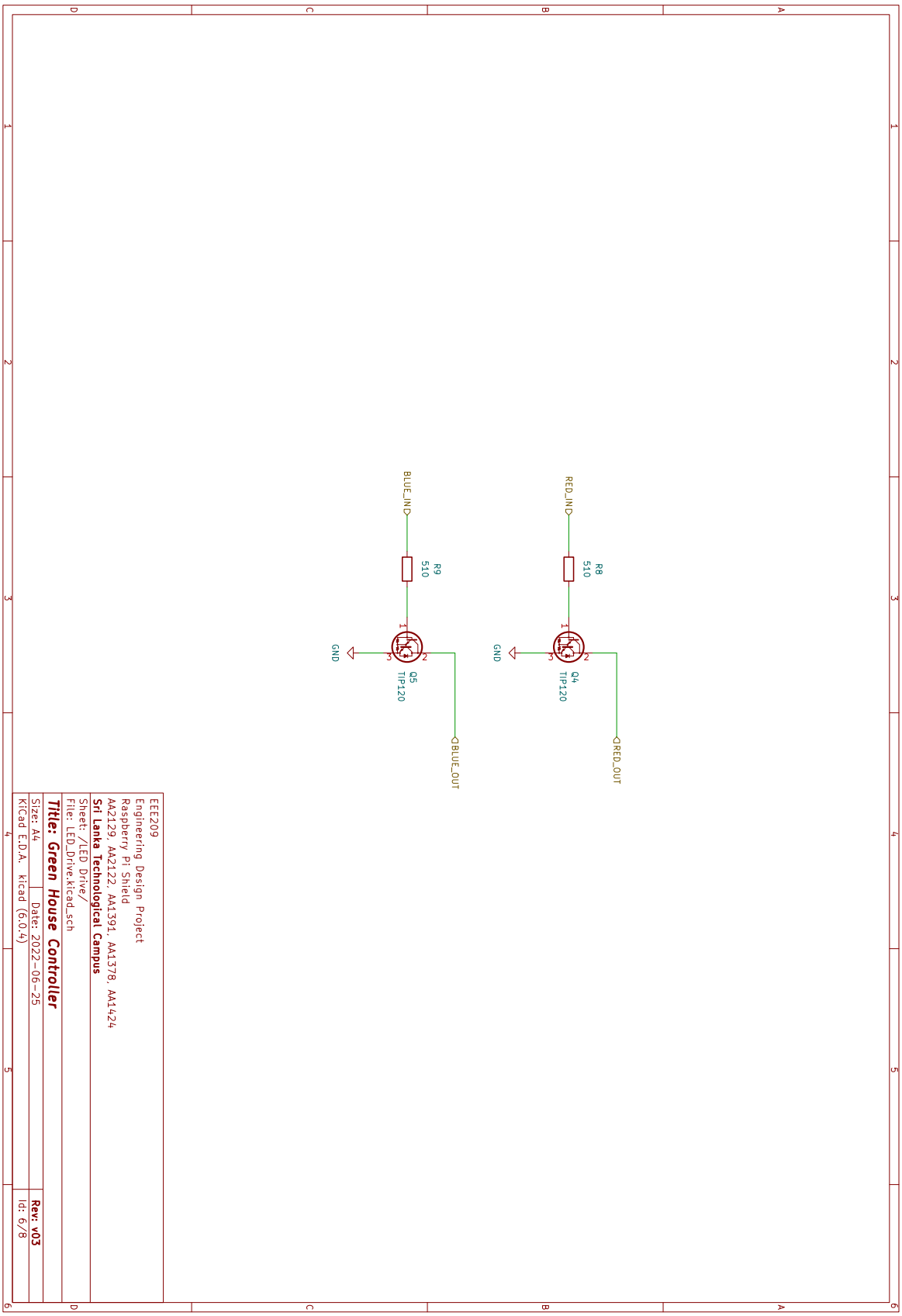


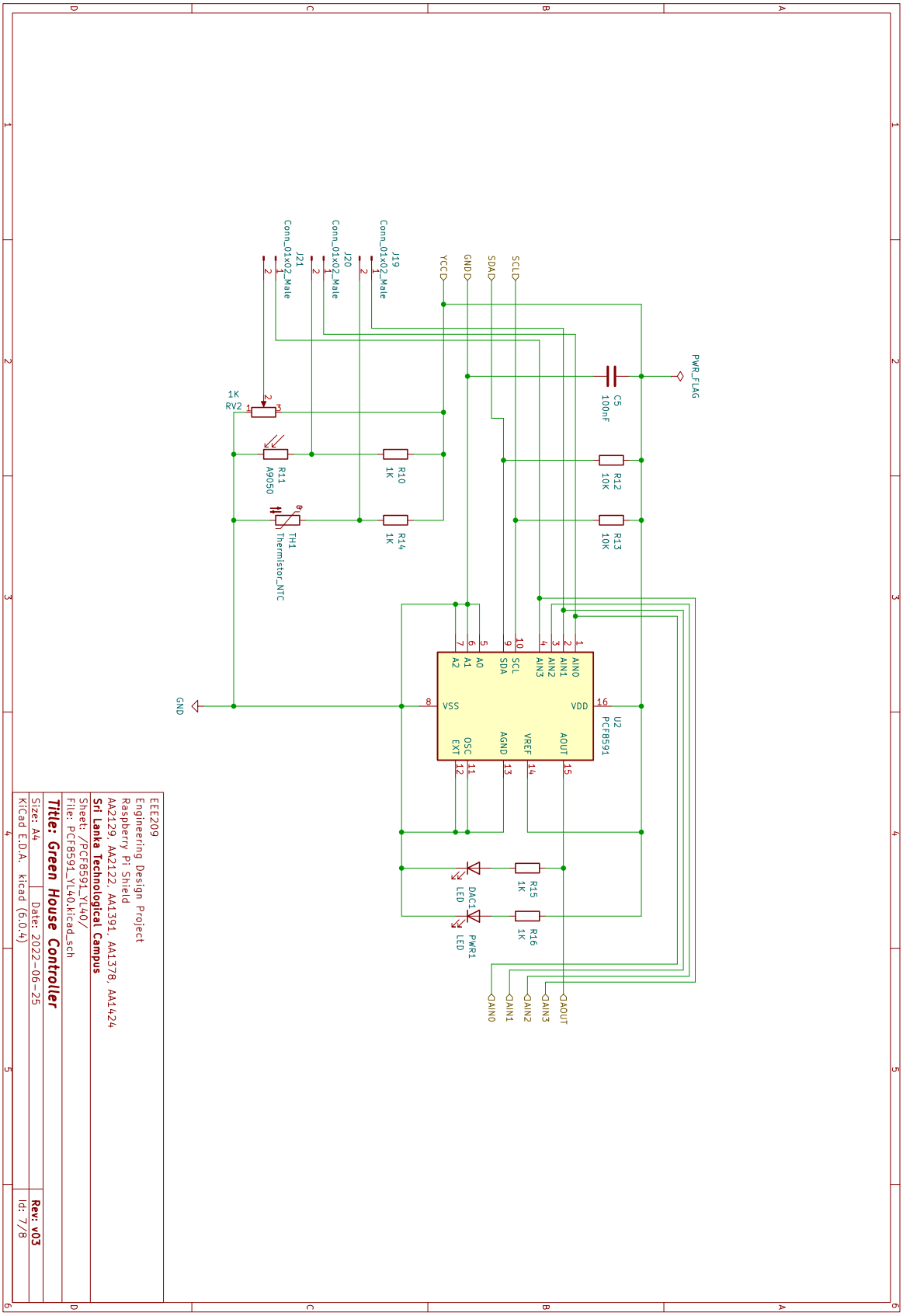

+IN
-IN
C1
330uF
C2
150nF
U1
VIN
ON/OFF
GND
FB
OUT
R1
820
D1
B120-E3
10K
RV1
L1
47uH
C3
330uF
C4
150nF
-OUT
+OUT
EEE209
Engineering Design Project
Raspberry Pi Shield
AA2129, AA2122, AA1391, AA1378, AA1424
Sri Lanka Technological Campus
Sheet: /LM2596S-BuckConverter/
File: LM2596S-BuckConverter.kicad_sch
Title: Green House Controller
Size: A4
Date: 2022-06-25
Rev: v03
KiCad E.D.A. kicad (6.0.4)
Id: 2/8

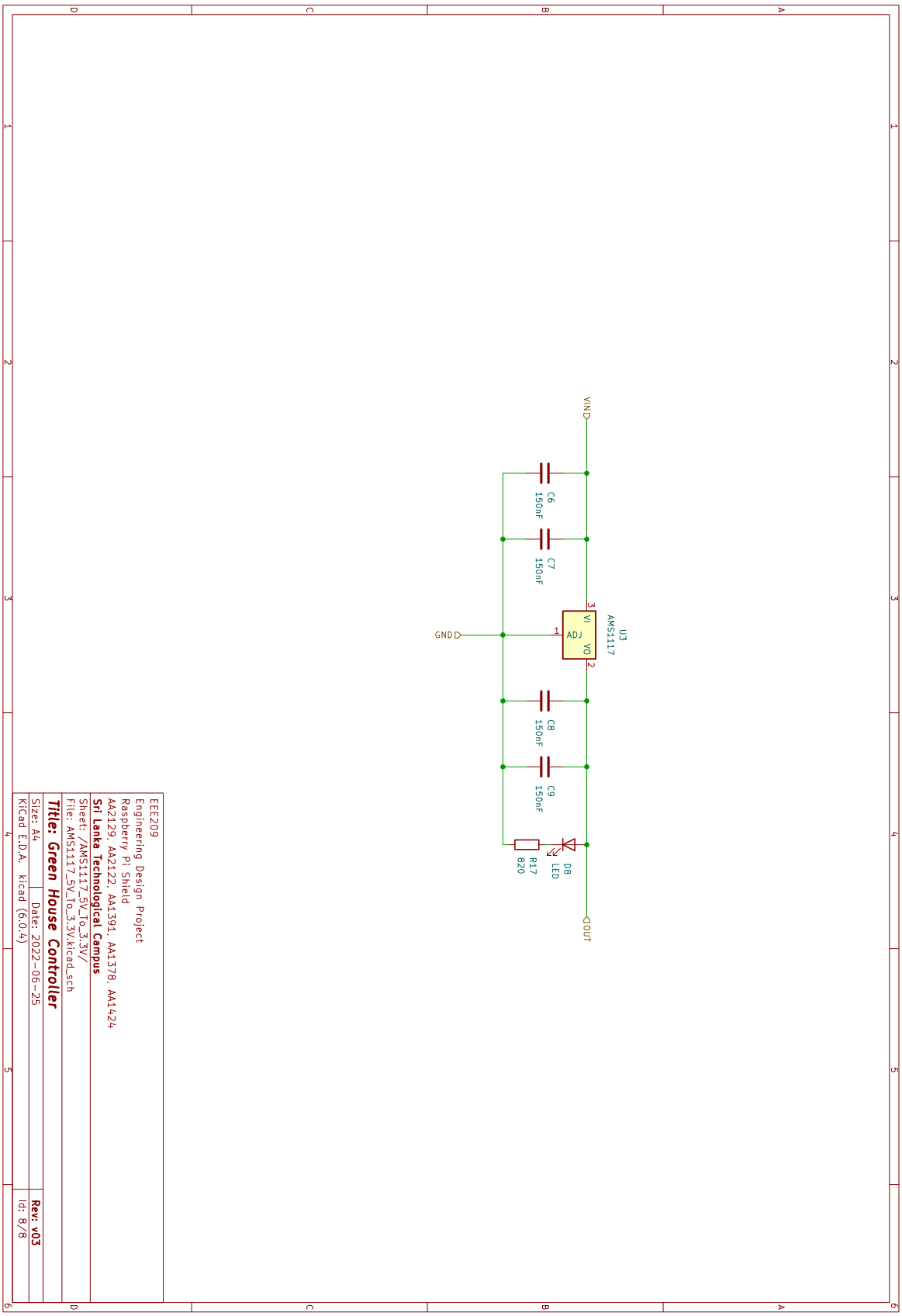

Relay_IN
R2
1K
R3
510
D2
LED
GND
Q1
BC547
D3
1N4007
+5V
NC
NO
K1
SANYOU_SRD_Form_C
Load_IN
EEE209
Engineering Design Project
Raspberry Pi Shield
AA2129, AA2122, AA1391, AA1378, AA1424
Sri Lanka Technological Campus
Sheet: /5V-Relay/
File: 5V-Relay.kicad_sch
Title: Green House Controller
Size: A4
Date: 2022-06-25
KiCad E.D.A. kicad (6.0.4)
Rev: v03
Id: 3/8


Relay_IN
R4
1K
R5
510
D4
LED
Q2
BC547
D5
1N4007
+5V
GND
NC
NO
K2
SANYOU_SRD_Form_C
Load_IN
EEE209
Engineering Design Project
Raspberry Pi Shield
AA2129, AA2122, AA1391, AA1378, AA1424
Sri Lanka Technological Campus
Sheet: /5V-Relay1/
File: 5V-Relay.kicad_sch
Title: Green House Controller
Size: A4
Date: 2022-06-25
KiCad E.D.A. kicad (6.0.4)
Rev: v03
Id: 4/8


Relay_IN
R6
1K
R7
510
D6
LED
GND
Q3
BC547
D7
1N4007
+5V
NC
NO
K3
SANYOU_SRD_Form_C
Load_IN
EEE209
Engineering Design Project
Raspberry Pi Shield
AA2129, AA2122, AA1391, AA1378, AA1424
Sri Lanka Technological Campus
Sheet: /5V-Relay2/
File: 5V-Relay.kicad_sch
Title: Green House Controller
Size: A4
Date: 2022-06-25
KiCad E.D.A. kicad (6.0.4)
Rev: v03
Id: 5/8


BLUE_IN
R9
510
Q5
TIP120
GND
BLUE_OUT
RED_IN
R8
510
Q4
TIP120
GND
RED_OUT
EEE209
Engineering Design Project
Raspberry Pi Shield
AA2129, AA2122, AA1391, AA1378, AA1424
Sri Lanka Technological Campus
Sheet: /LED Drive/
File: LED_Drive.kicad_sch
Title: Green House Controller
Size: A4
Date: 2022-06-25
KiCad E.D.A. kicad (6.0.4)
Rev: v03
Id: 6/8


EEE209
Engineering Design Project
Raspberry Pi Shield
AA2129, AA2122, AA1391, AA1378, AA1424
Sri Lanka Technological Campus
Sheet: /PCF8591_YL40/
File: PCF8591_YL40.kicad_sch
Title: Green House Controller
Size: A4
Date: 2022-06-25
KiCad E.D.A. kicad (6.0.4)
Rev: v03
Id: 7/8
J19
Conn_01x02_Male
J20
Conn_01x02_Male
J21
Conn_01x02_Male
YCC
GND
SDA
SCL
PWR_FLAG
C5
100nF
1K
RV2
R11
A9050
R10
1K
TH1
Thermistor_NTC
R14
1K
R12
10K
R13
10K
U2
PCF8591
A0
A1
A2
SCL
SDA
AIN0
AIN1
AIN2
AIN3
VSS
VDD
OSC
EXT
AGND
VREF
AOUT
DAC1
LED
R15
1K
PWR1
LED
R16
1K
AIN0
AIN1
AIN2
AIN3
AOUT
GND


VIN
C6
150nF
C7
150nF
GND
U3
AMS1117
VI
ADJ
VO
C8
150nF
C9
150nF
R17
820
D8
LED
OUT
EEE209
Engineering Design Project
Raspberry Pi Shield
AA2129, AA2122, AA1391, AA1378, AA1424
Sri Lanka Technological Campus
Sheet: /AMS1117_5V_To_3.3V/
File: AMS1117_5V_To_3.3V.kicad_sch
Title: Green House Controller
Size: A4
Date: 2022-06-25
KiCad E.D.A. kicad (6.0.4)
Rev: v03
Id: 8/8

A.3 Controller - PCB Layouts

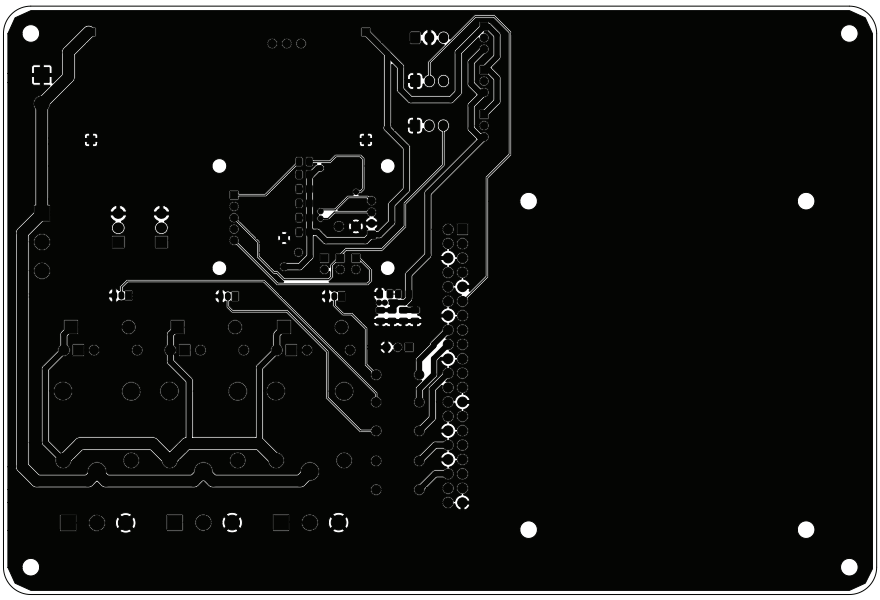


Figure A.5: Rear Copper Plate

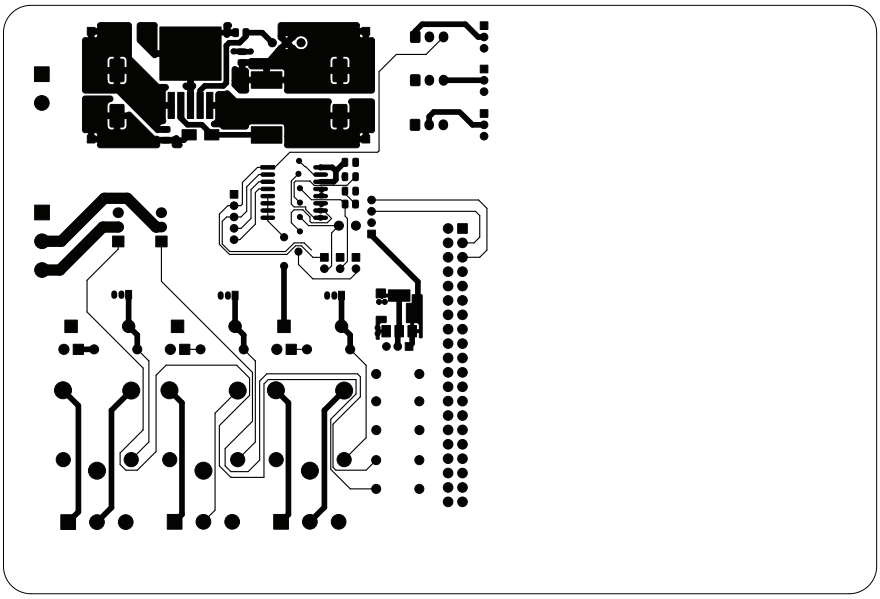


Figure A.6: Front Copper Plate

Appendix B

LED Grow Light

B.1 LED Grow Light - 3D View

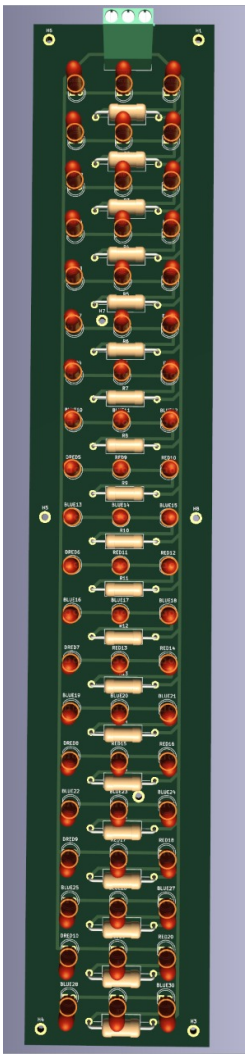
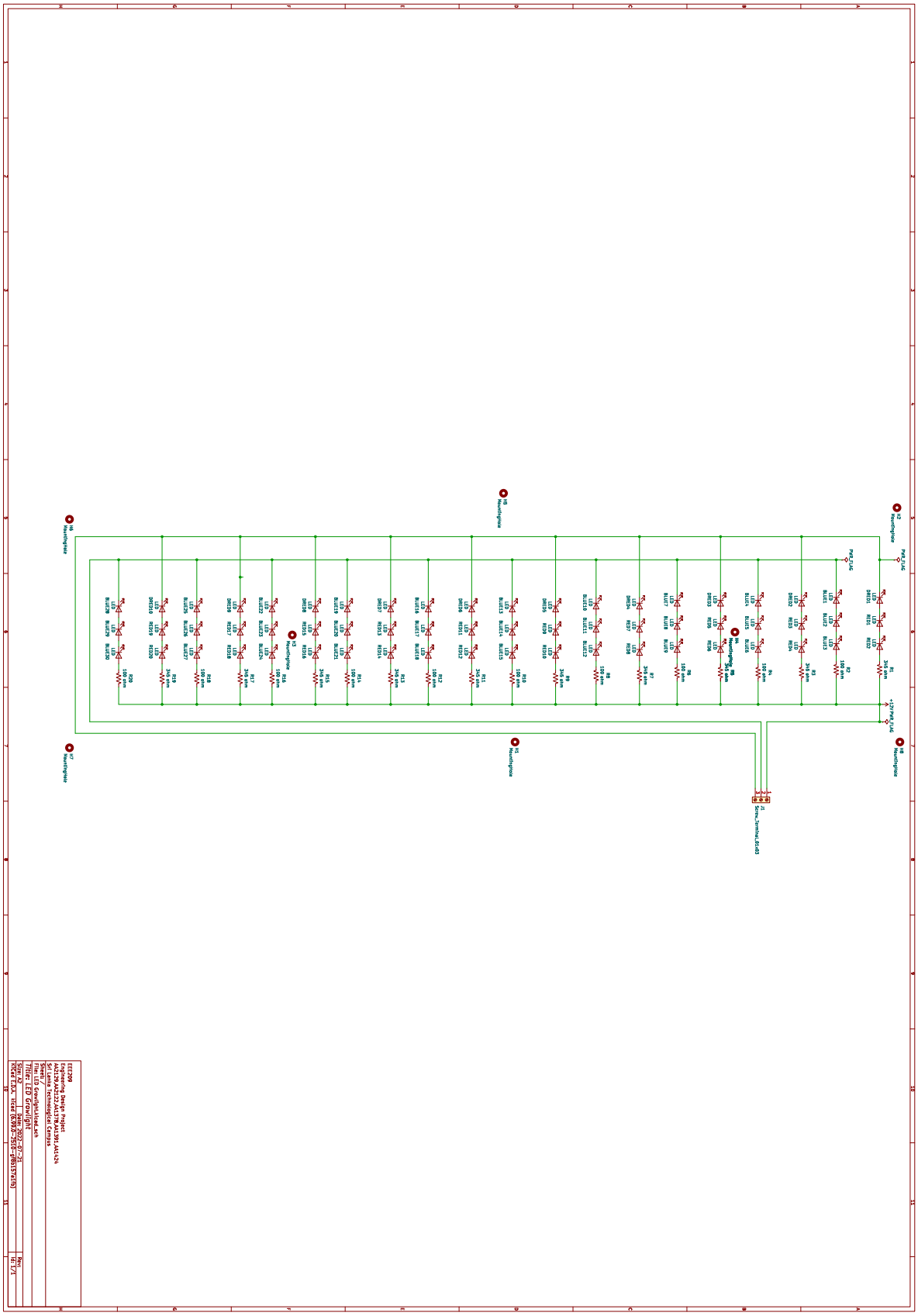


Figure B.1: 3D View LED Grow Light

B.2 LED Grow Light - Schematic Diagram



B.3 LED Grow Light - PCB layout

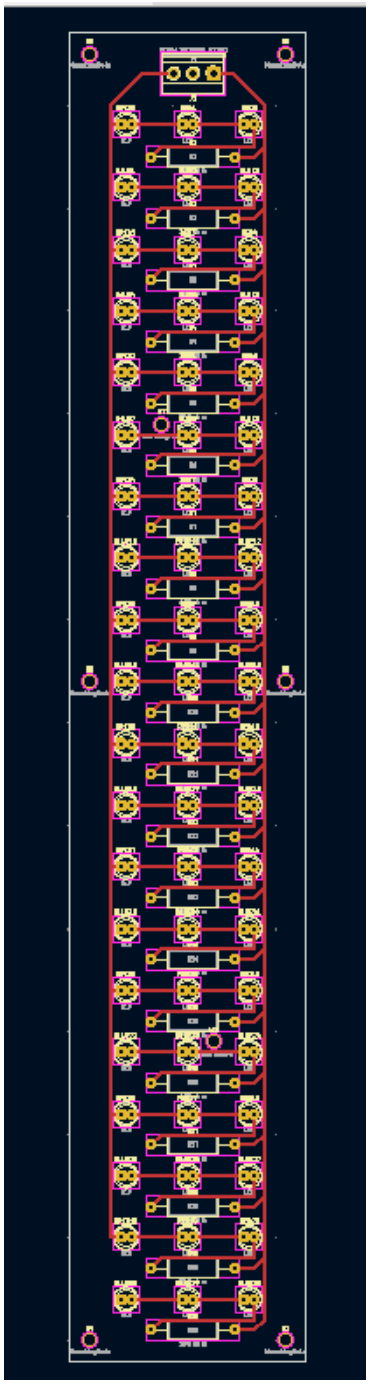


Figure B.2: Light PCB

Appendix C

Python Script

```
import time
import pyrebase
import Adafruit_DHT
import spidev
import RPi.GPIO as GPIO
import smbus
import csv
from datetime import datetime

# configures firebase

firebaseConfig = {
    "apiKey": "AIzaSyA1cZhgLy_rE6I23zvKhg4Gy3LpfJJJ6tk",
    "authDomain": "greenhouseautomation-8af0f.firebaseapp.com",
    "databaseURL": "https://greenhouseautomation-8af0f-default-rtdb.firebaseio.com",
    "storageBucket": "greenhouseautomation-8af0f.appspot.com"
}

firebase = pyrebase.initialize_app(firebaseConfig)
db = firebase.database()

# configures the smbus module for the analog to digital convertor

bus = smbus.SMBus(1)
address = 0x48

# configures output pins and initializes them
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

redLED = 23
blueLED = 24

GPIO.setup(17, GPIO.OUT)
GPIO.output(17, False)
GPIO.setup(27, GPIO.OUT)
GPIO.output(27, False)
GPIO.setup(22, GPIO.OUT)
GPIO.output(22, False)
GPIO.setup(redLED, GPIO.OUT)
```

```
GPIO.setup(blueLED, GPIO.OUT)

red_pwm = GPIO.PWM(redLED, 1000)
blue_pwm = GPIO.PWM(blueLED, 1000)


# read function used for the analog to digital convertor
def read(control):
    write = bus.write_byte(address, control)
    read = bus.read_byte(address)
    return read


# function to return temperature and humidity values from sensor
def temp_humid_sensor():
    DHT_SENSOR = Adafruit_DHT.DHT11
    DHT_PIN = 4

    humidity, temperature = Adafruit_DHT.read_retry(DHT_SENSOR, DHT_PIN)
    if humidity is not None and temperature is not None:
        print("Temp={0:0.1f}C Humidity={1:0.1f}%".format(temperature, humidity))
    else:
        print("DHT sensor failure.")
        temperature = 29
        humidity = 91
    return temperature, humidity


# function to return light and moisture values from sensor through ADC
def light_moisture_sensor():
    unused1 = read(0x40)
    moisture_raw = read(0x41)
    unused2 = read(0x42)
    light_raw = read(0x43)

    moisture = (moisture_raw / float(255) * 100)
    moisture = round(moisture, 2)

    light = (light_raw / float(255) * 1000)
    light = round(light, 2)

    return light, moisture


# function to update the firebase with the latest sensor values
def update_database(humidity, light, moisture, temperature):
    db.child("IOTGreenhouse").update({"humidity": humidity})
    db.child("IOTGreenhouse").update({"luminosity": light})
    db.child("IOTGreenhouse").update({"moisture": moisture})
    db.child("IOTGreenhouse").update({"temperature": temperature})


# function to open or close fan
def fan(state):
    try:
        GPIO.output(17, state)
    except:
        print("error")
        pass
```

```
# function to start or stop sprinkling
def sprinkle(state):
    try:
        GPIO.output(27, state)
    except:
        print("error")
        pass


# function to start or stop dripping
def drip(state):
    try:
        GPIO.output(22, state)
    except:
        print("error")
        pass


# function to filter quotes from string and then convert to int
def convert_int(string_unfiltered):
    string_filtered = string_unfiltered.replace('"', '')
    integer = int(string_filtered)
    return integer


# function for saving data to a local csv file
def save_to_csv(temp, humid, light, moist):
    now = datetime.now()
    current_day = now.strftime("%d/%m/%Y")
    current_time = now.strftime("%H:%M:%S")

    with open('data.csv', 'a', newline='') as file:
        fieldnames = ['Date', 'Time', 'Temperature', 'Humidity', 'Light', 'Moisture']
        writer = csv.DictWriter(file, fieldnames=fieldnames)
        # writer.writeheader()
        writer.writerow(
            {'Date': current_day, 'Time': current_time, 'Temperature': temp, 'Humidity': humid, 'Light': light,
             'Moisture': moist})


# main code starts here


try:
    # starts the red and blue LEDs
    red_pwm.start(50)
    blue_pwm.start(50)
    while True:
        # gets new data from sensors and uploads them to firbase and save locally
        temp, humid = temp_humid_sensor()
        light_level, moist = light_moisture_sensor()

        save_to_csv(temp, humid, light_level, moist)
        update_database(humid, light_level, moist, temp)

        # adjusts dutycycle of red and blue PWM according to instructions
        dutycycle = convert_int(db.child("IOTGreenhouse").child("led intensity").get().val())

        if dutycycle<50:
            reddutycycle = 100
            bluedutycycle = 25
```

```
        elif dutycycle == 50:
            reddutycycle = 100
            bluedutycycle = 100
        else:
            reddutycycle = 25
            bluedutycycle = 100

        red_pwm.ChangeDutyCycle(reddutycycle)
        blue_pwm.ChangeDutyCycle(bluedutycycle)

        # checks and switches the mode of operation
        automatic = db.child("IOTGreenhouse").child("automatic").get().val()

        if automatic == 'true':
            # code for automatic mode starts here
            # retrieves the acceptable parameters from firebase database
            temp_max = convert_int(db.child("IOTGreenhouse").child("temp max").get().val())
            temp_min = convert_int(db.child("IOTGreenhouse").child("temp min").get().val())
            humid_max = convert_int(db.child("IOTGreenhouse").child("humid max").get().val())
            humid_min = convert_int(db.child("IOTGreenhouse").child("humid min").get().val())
            moist_max = convert_int(db.child("IOTGreenhouse").child("moist max").get().val())
            moist_min = convert_int(db.child("IOTGreenhouse").child("moist min").get().val())

            # opens and closes relays depending on conditions
            if temp > temp_max:
                fan(True)
            elif temp < temp_min:
                fan(False)

            if humid > humid_max:
                sprinkle(True)
            elif humid < humid_min:
                sprinkle(False)

            if moist < moist_min:
                drip(True)
            elif moist > moist_max:
                drip(False)
        else:
            # code for manual mode starts here
            # retrieves instructions for relays from firebase
            turn_on_dripping = db.child("IOTGreenhouse").child("turn on dripping").get().val()
            turn_on_sprinkling = db.child("IOTGreenhouse").child("turn on sprinkling").get().val()
            turn_on_fan = db.child("IOTGreenhouse").child("turn on fan").get().val()

            # opens or closes relays according to instructions
            if turn_on_dripping == 'true':
                drip(True)
            else:
                drip(False)
            if turn_on_sprinkling == 'true':
                sprinkle(True)
            else:
                sprinkle(False)
            if turn_on_fan == 'true':
                fan(True)
            else:
                fan(False)

except KeyboardInterrupt:
    # breaks the loop and moves on to the cleanup of outputs
```

```
    print("Interrupted by user")

finally:
    # cleans up outputs and terminates the lighting
    red_pwm.stop()
    blue_pwm.stop()
    GPIO.cleanup()
```

Appendix D

Kodular Blocks

D.1 Login Page Blocks

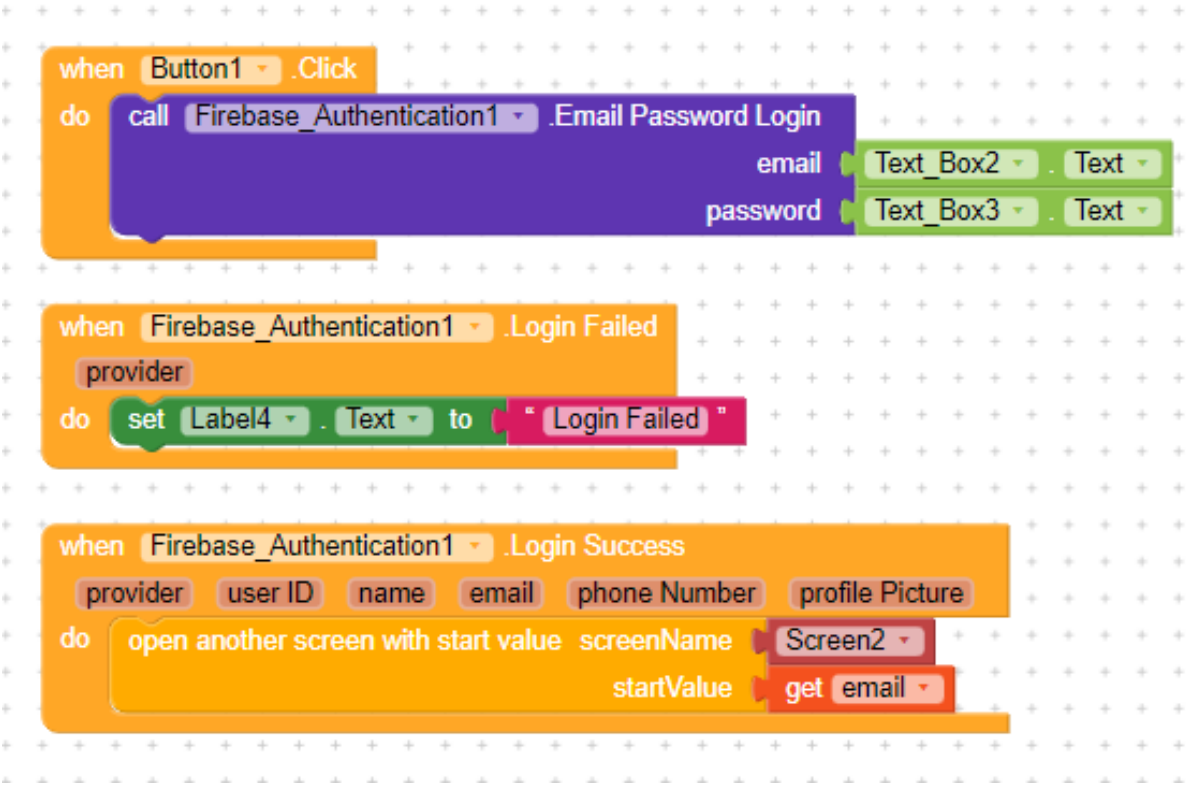


Figure D.1: Login page Block

D.2 Auto Page Blocks

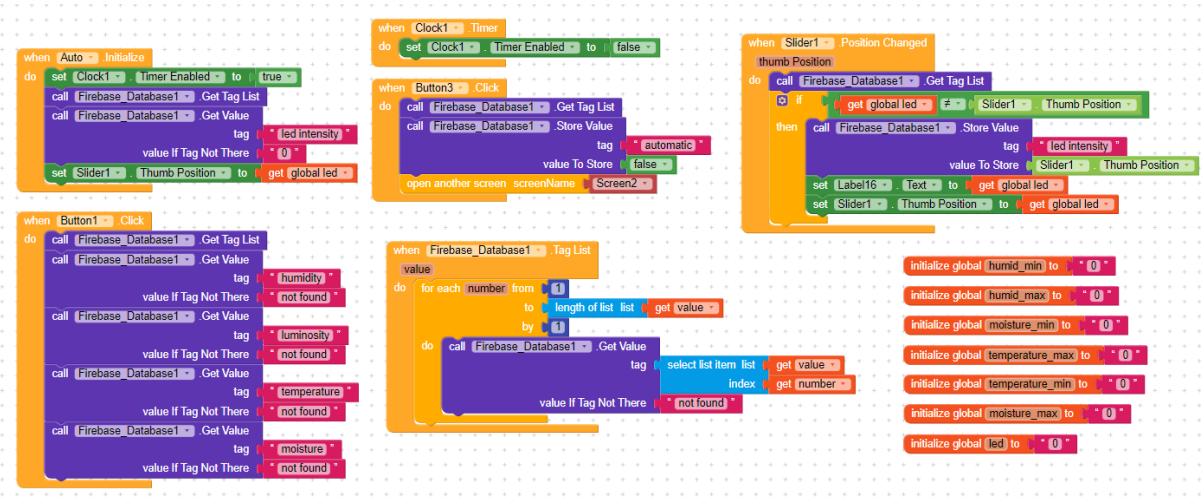


Figure D.2: Auto Page Block 1

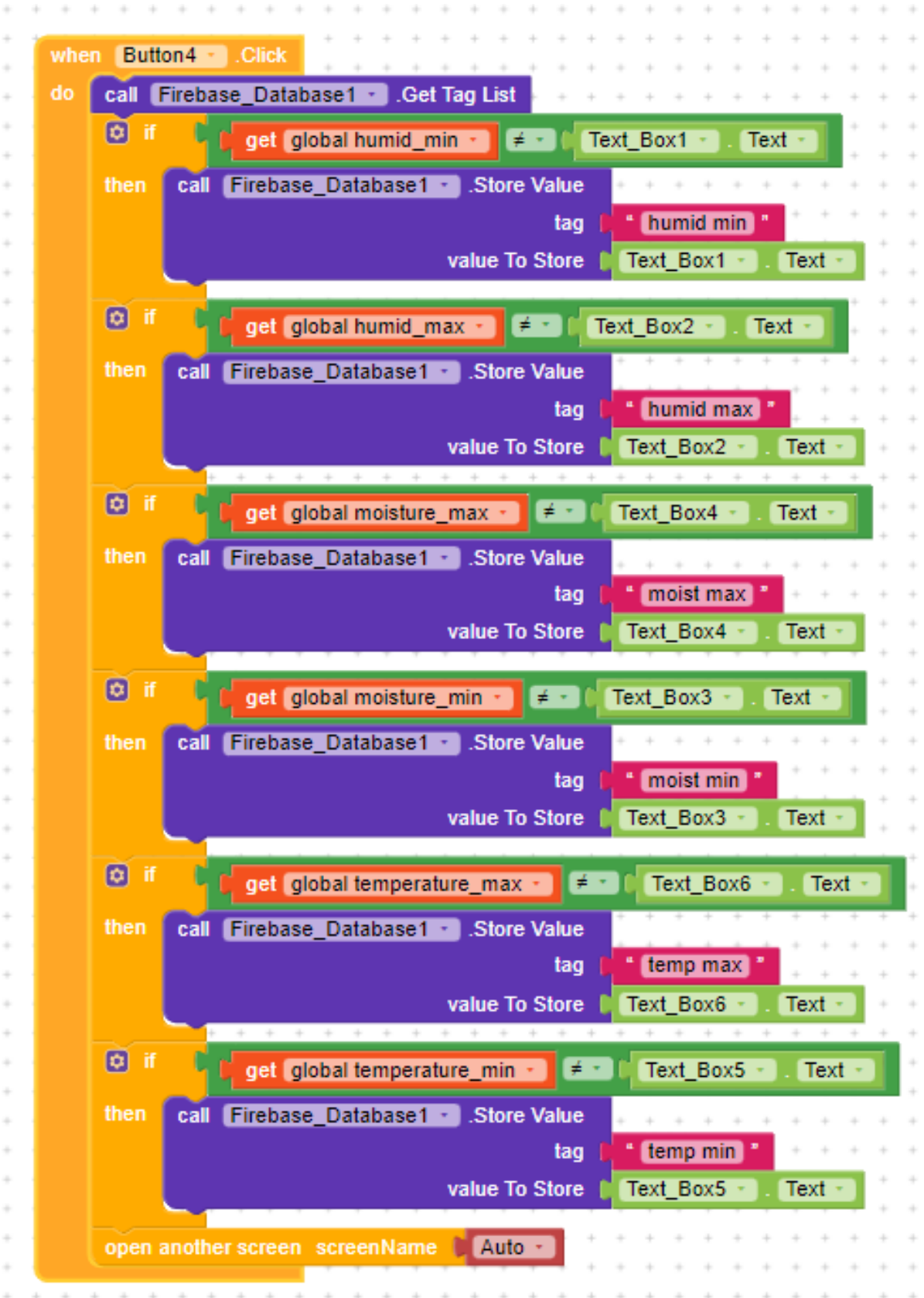


Figure D.3: Auto Page Block 2


when Firebase_Database1 .Got Value
tag value
do if get tag = " automatic "
then if get value = false
then open another screen screenName Screen2
if get tag = " humidity "
then set Label8 . Text to get value
if get tag = " luminosity "
then set Label8 . Text to get value
if get tag = " temperature "
then set Label7 . Text to get value
if get tag = " moisture "
then set Label9 . Text to get value
if get tag = " humid max "
then set Text_Box2 . Text to get value
set global humid_max to get value
if get tag = " humid min "
then set Text_Box1 . Text to get value
set global humid_min to get value
if get tag = " moist max "
then set Text_Box4 . Text to get value
set global moisture_max to get value
if get tag = " moist min "
then set Text_Box3 . Text to get value
set global moisture_min to get value
if get tag = " temp max "
then set Text_Box6 . Text to get value
set global temperature_max to get value
if get tag = " temp min "
then set Text_Box5 . Text to get value
set global temperature_min to get value
if get tag = " led intensity "
then if Slider1 . Thumb Position ≠ get value
then set Slider1 . Thumb Position to get value
set Label16 . Text to get value
set global led to get value

D.3 Manual Page Blocks

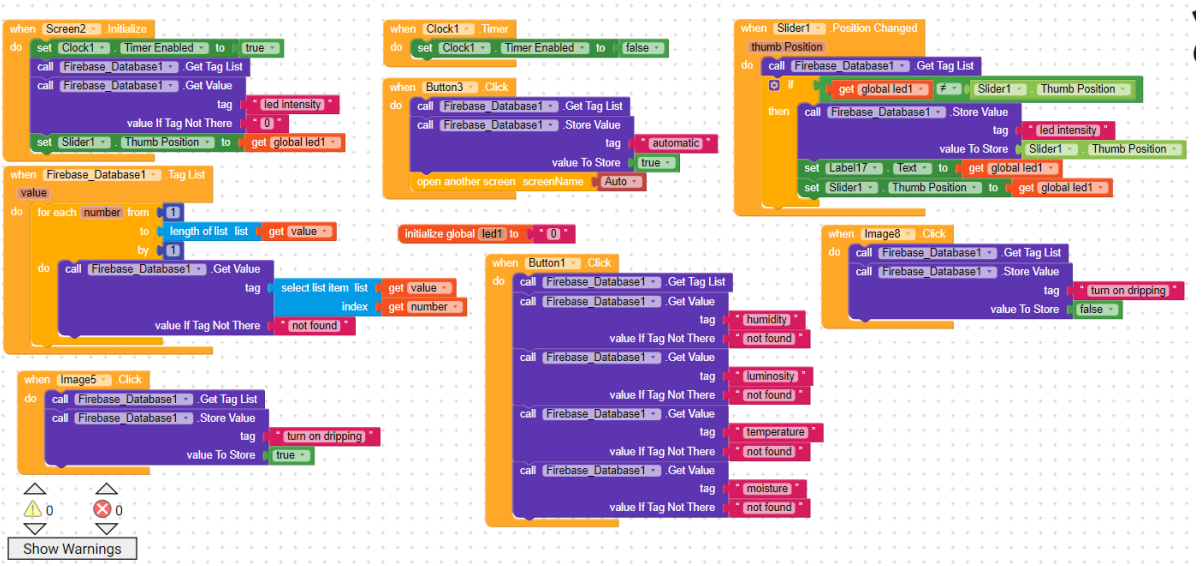


Figure D.5: Manual Page Block 1

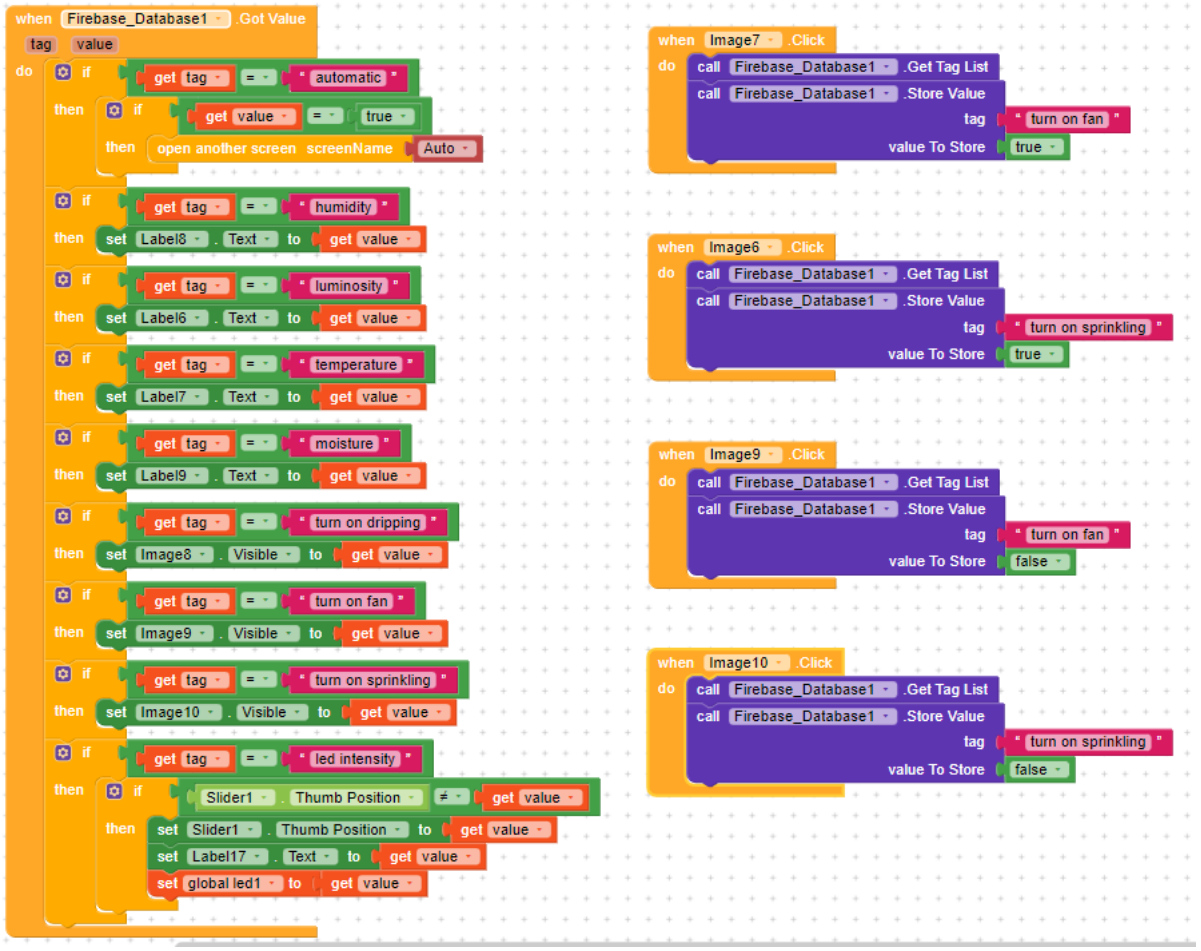


Figure D.6: Manual Page Block 2